DEV323

Introduction to the AWS CLI

Leo Zhadanovsky
Principal Solutions Architect
AWS

November 29, 2017

AWS re:Invent

aws

Agenda

- Basics
    - Installation
    - Configuration
    - Syntax
- Foundations
    - Profiles
    - Environment Variables
    - Roles
- Advanced
    - Querying & Filtering
    - MFA
    - S3 Commands

AWS Command Line Interface

Unified tool to manage your AWS Services

Basics

- Installing the CLI
- Setting up Credentials
- Configuration Files
- Syntax
- Help

Install the CLI

- Requirements
    - Python 2 version 2.6.5+ or Python 3 version 3.3+
    - OS: Windows, Linux, MacOS, or Unix
    - Installation Options
        - pip
        - Bundled Installer
        - MSI
    - virtualenv recommended when using pip

Install the CLI

```
$ pip install awscli
```

Update the CLI

```
$ pip install awscli --upgrade
```

Check version of the CLI

```
$ aws --version
```

Uninstall the CLI

```
$ pip uninstall awscli
```

Setting up Credentials

```
$ aws configure
AWS Access Key ID [None]:AKIAIOSFODNN7EXAMPLE
AWS Secret Access Key [None]:wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
Default region name [None]:us-east-1
Default output format [None]:json
```

Configuration files

~/.aws/credentials

- Supported by all AWS SDKs
- Contains credentials

~/.aws/config

- Some settings used only by CLI
- Can contain credentials

Configuration files

~/.aws/credentials

```
[prod]
aws_access_key_id = foo
aws_secret_access_key = bar
```

~/.aws/config

```
[profile prod]
aws_access_key_id = foo
aws_secret_access_key = bar
region = us-east-1
emr =
  service_role = EMR_DefaultRole
  instance_profile = EMR_EC2_DefaultRole
  log_uri = s3://myBucket/logs
  enable_debugging = True
  key_name = myKeyName
  key_pair_file = /home/myUser/myKeyName.pem
output = json
```

Syntax

Syntax

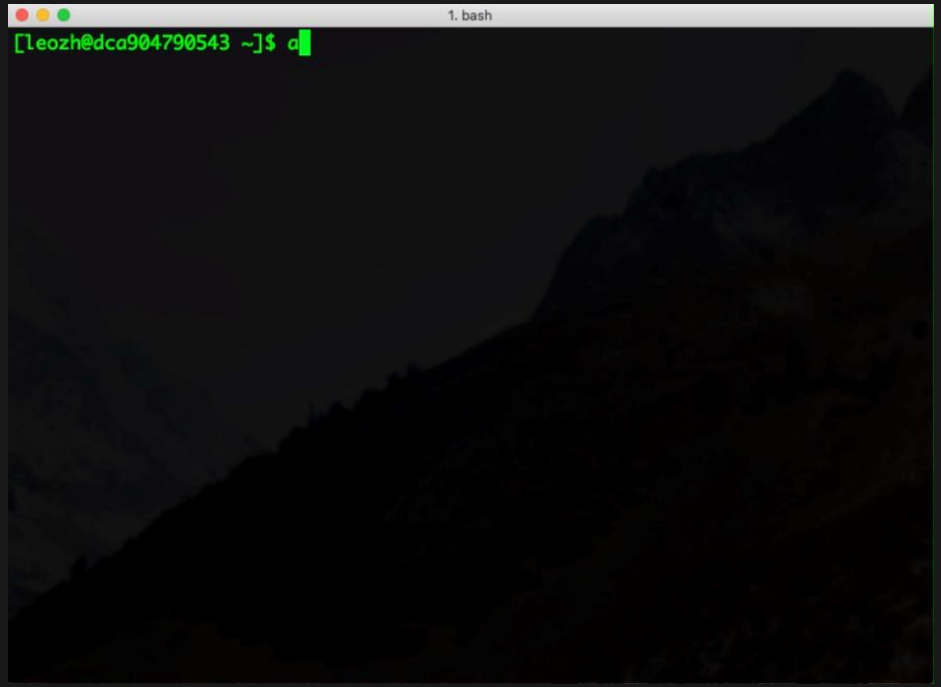
$ aws iam list-access-keys

service (command) operation (subcommand)

Help

```
$ aws ec2 describe-instances help
```

Help



Foundations

- Configuration Settings and Precedence
- Named Profiles
- Environment Variables
- Command Line Options
- Roles
- Output Types

- Tab Completion
- aws-shell
- Bastion Hosts

Configuration Settings and Precedence

1. **Command line options** – region, output format and profile can be specified as command options to override default settings.
2. **Environment variables** – AWS_ACCESS_KEY_ID, AWS_SECRET_ACCESS_KEY, and AWS_SESSION_TOKEN.
3. **The AWS credentials file** – located at ~/.aws/credentials on Linux, macOS, or Unix, or at C:\Users*USERNAME* \.aws\credentials on Windows. This file can contain multiple named profiles in addition to a default profile.
4. **The CLI configuration file** – typically located at ~/.aws/config on Linux, macOS, or Unix, or at C:\Users*USERNAME* \.aws\config on Windows. This file can contain a default profile, named profiles, and CLI specific configuration parameters for each.
5. **Container credentials** – provided by Amazon EC2 Container Service on container instances when you assign a role to your task.
6. **Instance profile credentials** – these credentials can be used on EC2 instances with an assigned instance role, and are delivered through the Amazon EC2 metadata service.

Named Profiles

~/.aws/credentials

```
[default]
aws_access_key_id=AKIAIOSFODNN7EXAMPLE
aws_secret_access_key=wJalrXUtnFEMI/K7MDENG/bPxRfiCYE
XAMPLEKEY

[user2]
aws_access_key_id=AKIAI44QH8DHBEXAMPLE
aws_secret_access_key=je7MtGbClwBF/2Zp9Utk/h3yCo8nvbE
XAMPLEKEY
```

~/.aws/config

```
[default]
region=us-east-1
output=json

[profile user2]
region=us-west-2
output=text
```

Using Profiles

```
$ aws ec2 describe-instances --profile user2
```

Environment Variables

- AWS_ACCESS_KEY_ID – AWS access key.
- AWS_SECRET_ACCESS_KEY – AWS secret key. Access and secret key variables override credentials stored in credential and config files.
- AWS_SESSION_TOKEN – Specify a session token if you are using temporary security credentials.
- AWS_DEFAULT_REGION – AWS region. This variable overrides the default region of the in-use profile, if set.
- AWS_DEFAULT_OUTPUT – Change the AWS CLI's output formatting to json, text, or table.
- AWS_PROFILE – name of the CLI profile to use. This can be the name of a profile stored in a credential or config file, or default to use the default profile.
- AWS_CA_BUNDLE – Specify the path to a certificate bundle to use for HTTPS certificate validation.
- AWS_SHARED_CREDENTIALS_FILE – Change the location of the file that the AWS CLI uses to store access keys.
- AWS_CONFIG_FILE – Change the location of the file that the AWS CLI uses to store configuration profiles.

Using Environment Variables

Linux, macOS, or Unix

```
$ export AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
$ export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
$ export AWS_DEFAULT_REGION=us-west-2
```

Windows

```
> set  AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
> set AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
> set AWS_DEFAULT_REGION=us-west-2
```

Command Line Options

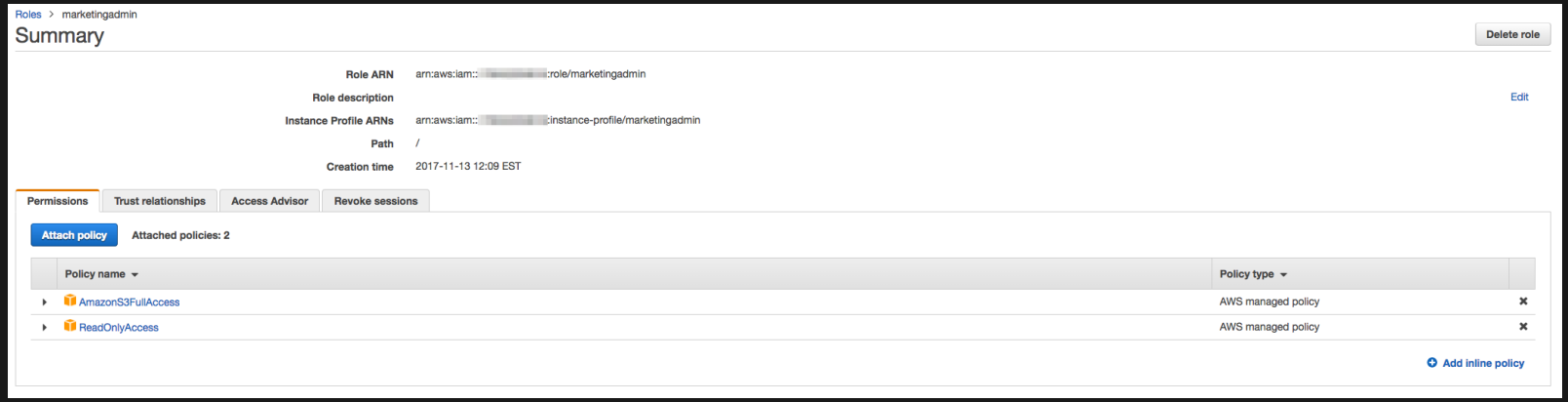
- **--profile** – name of a profile to use, or "default" to use the default profile.
- **--region** – AWS region to call.
- **--output** – output format.
- **--endpoint-url** – The endpoint to make the call against. The endpoint can be the address of a proxy or an endpoint URL for the in-use AWS region. Specifying an endpoint is not required for normal use as the AWS CLI determines which endpoint to call based on the in-use region.

* The above options override the corresponding profile settings for a single operation.

Roles

- Set of permissions granted to a trusted entity
- Assumed by **IAM users**, applications or AWS services like EC2
- Use case:
  - Cross-services
  - Temporary access
  - Cross-account
  - Federation
- Benefits
  - Security: no sharing of secrets
  - Control: revoke access anytime

Roles



~/.aws/config

```
[profilemarketingadmin]
role_arn = arn:aws:iam::123456789012:role/marketingadmin
source_profile = default
```

Output Types

json

```
{
   "Places": [
       {
            "City": "Seattle",

      "State": "WA"
      },
      {
            "City": "Las Vegas",
            "State": "NV"
      }
   ]
}
```

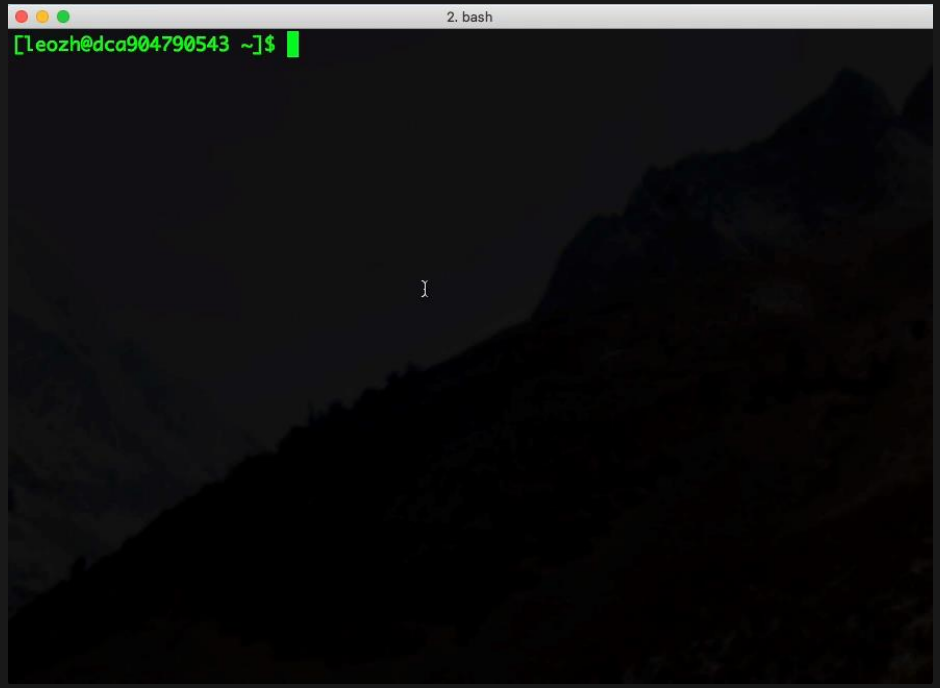
text

```
PLACES  Seattle    WA
PLACES  Las Vegas  NV
```

table

```
----------------------
|     SomeOperationName  |
+------------------------+
||         Places       ||
|+-------------+---------+|
|| City        | State   ||
|+-------------+---------+|
|| Seattle     | WA      ||
|| Las Vegas   | NV      ||
|+-------------+---------+|
```

Tab Completion



Tab Completion Setup

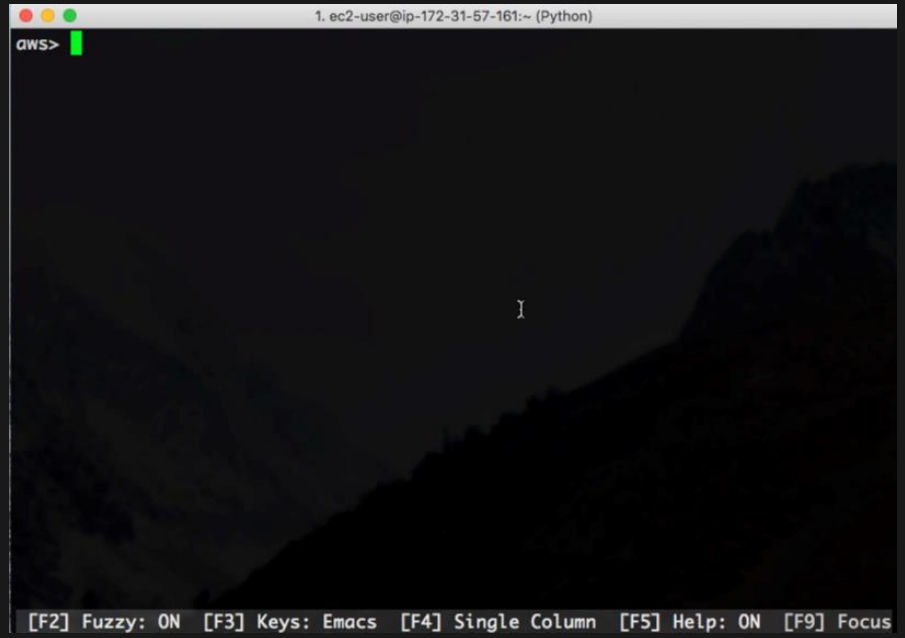
```
$ which aws_completer
/usr/local/bin/aws_completer

$ complete -C '/usr/local/bin/aws_completer' aws
```

* assuming bash

AWS Shell

An integrated shell for working with the AWS CLI

Get it here:
https://github.com/awslabs/aws-shell



Bastion Hosts

Demo

Advanced

- MFA
- Querying & Filtering
- CLI Aliases
- JMESPath Terminal
- Generate CLI Skeleton

MFA

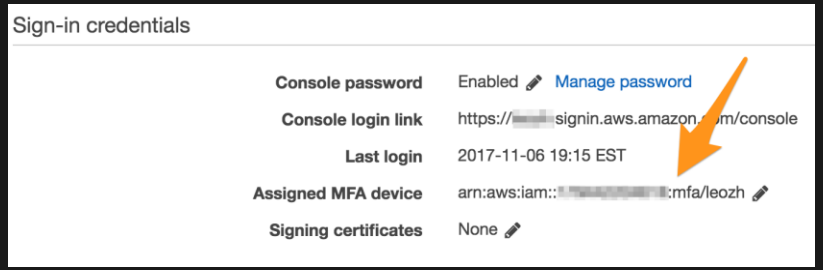
- Require MFA for role assumption:

```
{
 "Version": "2012-10-17",
 "Statement": [
   {
     "Sid": "",
     "Effect": "Allow",
     "Principal": { "AWS": "arn:aws:iam::179442201234:user/leozh" },
     "Action": "sts:AssumeRole",
     "Condition": { "Bool": { "aws:MultiFactorAuthPresent": true } }
   }
 ]
}
```

- ~/.aws/config

```
[profile billing]
role_arn = arn:aws:iam::179442201234:role/billing
source_profile = default
mfa_serial = arn:aws:iam::179442201234:mfa/leozh
```

Where do I find my MFA ARN?



or

```
$ aws iam list-mfa-devices --user-name leozh
{
    "MFADevices": [
        {
            "UserName": "leozh",
            "SerialNumber": "arn:aws:iam::179442201234:mfa/leozh",
            "EnableDate": "2016-09-15T00:27:33Z"
        }
    ]
}
```

MFA

Demo

Querying & Filtering

Filtering

- `--filter`
- Server side
- Only available on certain subcommands (such as many list-* and describe-*)
- Use this if you are expecting a large (1000 or >) amount of items as a return
- Use in conjunction with --page-size on large item sets

Querying

- `--query`
- Client side
- Available on all subcommands
- JMESPath query language

Filtering

```
aws ec2 describe-instances --filter Name=instance-type,Values=t2.micro
{
   "Reservations": [
       {
           "Instances": [
               {
                   "Monitoring": {
                       "State": "disabled"
                   },
                   "PublicDnsName": "ec2-34-228-84-177.compute-1.amazonaws.com",
                   "State": {
                       "Code": 16,
                       "Name": "running"
                   },
                   "EbsOptimized": false,
                   "LaunchTime": "2017-07-18T23:14:07.000Z",
…
…
            },
```

CLI Aliases

https://github.com/awslabs/awscli-aliases

```
$ aws whoami
{
   "Account": "179442201234”,
   "UserId": "AIDAIXV2V6G4AEXAMPLE”,
   "Arn": "arn:aws:iam::179442201234:user/leozh"
}

$ aws amazon-linux-amis
ami-6057e21a amzn-ami-hvm-2017.09.1.20171103-x86_64-gp2 Amazon Linux AMI 2017.09.1.20171103 x86_64 HVM GP2
ami-8c1be5f6 amzn-ami-hvm-2017.09.0.20170930-x86_64-gp2 Amazon Linux AMI 2017.09.0.20170930 x86_64 HVM GP2
ami-5e8c9625 amzn-ami-hvm-2017.09.rc-0.20170913-x86_64-gp2 Amazon Linux AMI 2017.09.rc-0.20170913 x86_64 HVM
GP2
ami-4fffc834 amzn-ami-hvm-2017.03.1.20170812-x86_64-gp2 Amazon Linux AMI 2017.03.1.20170812 x86_64 HVM GP2
…
```

Querying

```
$ aws iam list-users
{
   "Users": [
       {
           "UserName": "leozh",
           "PasswordLastUsed": "2017-11-07T00:15:03Z",
           "CreateDate": "2013-08-04T17:24:01Z",
           "UserId": "AIDAIXV2V6G4AAAV5UXYZ”,
           "Path": "/",
           "Arn": "arn:aws:iam::179442201234:user/leozh"
       },
       {
           "UserName": “billing”,
           "PasswordLastUsed": "2015-03-17T18:47:44Z",
           "CreateDate": "2015-03-17T03:31:45Z",
           "UserId": "AIDAIFVT4OFCCN6B5QABC”,
           "Path": "/",
           "Arn": "arn:aws:iam::1794422041234:user/billing”
       }
```

Querying

```
$ aws iam list-users --query 'Users[*].[UserName, CreateDate, PasswordLastUsed, Arn]'
[
    [
        "leozh",
        "2013-08-04T17:24:01Z",
        "2017-11-07T00:15:03Z",
        "arn:aws:iam::179442201234:user/leozh"
    ],
    [
        “Billing”,
        "2015-03-17T03:31:45Z",
        "2015-03-17T18:47:44Z",
        "arn:aws:iam::179442201234:user/billing”
    ]
]
```

Querying

```
$ aws ec2 describe-instances --query Reservations[*].Instances[*].State.Name
[
    [
        "stopped"
    ],
    [
        "running"
    ],
    [
        "running"
    ]
]
```

JMESPath Terminal

Demo

Querying

JMESPath

Learn more here: http://jmespath.org/tutorial.html

JMESPath Terminal

https://github.com/jmespath/jmespath.terminal

Generate CLI Skeleton

```
$ aws ec2 run-instances --generate-cli-skeleton > ec2runinst.json
{
   "DryRun": true,
   "ImageId": "",
   "MinCount": 0,
   "MaxCount": 0,
   "KeyName": "",
   "SecurityGroups": [
       ""
   ],
   "SecurityGroupIds": [
       ""
   ],
   "UserData": "",
   "InstanceType": "",
…
}
```

Generate CLI Skeleton

edit ec2runinst.json

```
{
   "DryRun": false,
   "ImageId": "ami-dfc39aef",
   "KeyName": "mykey",
   "SecurityGroups": [
       "my-sg"
   ],
   "InstanceType": "t2.micro",
   "Monitoring": {
       "Enabled": true
   }
}
```

Generate CLI Skeleton

```
$ aws ec2 run-instances --cli-input-json file://ec2runinst.json
{
   "OwnerId": "123456789012",
   "ReservationId": "r-d94a2b1",
   "Groups": [],
   "Instances": [
...
```

Generate CLI Skeleton

Demo

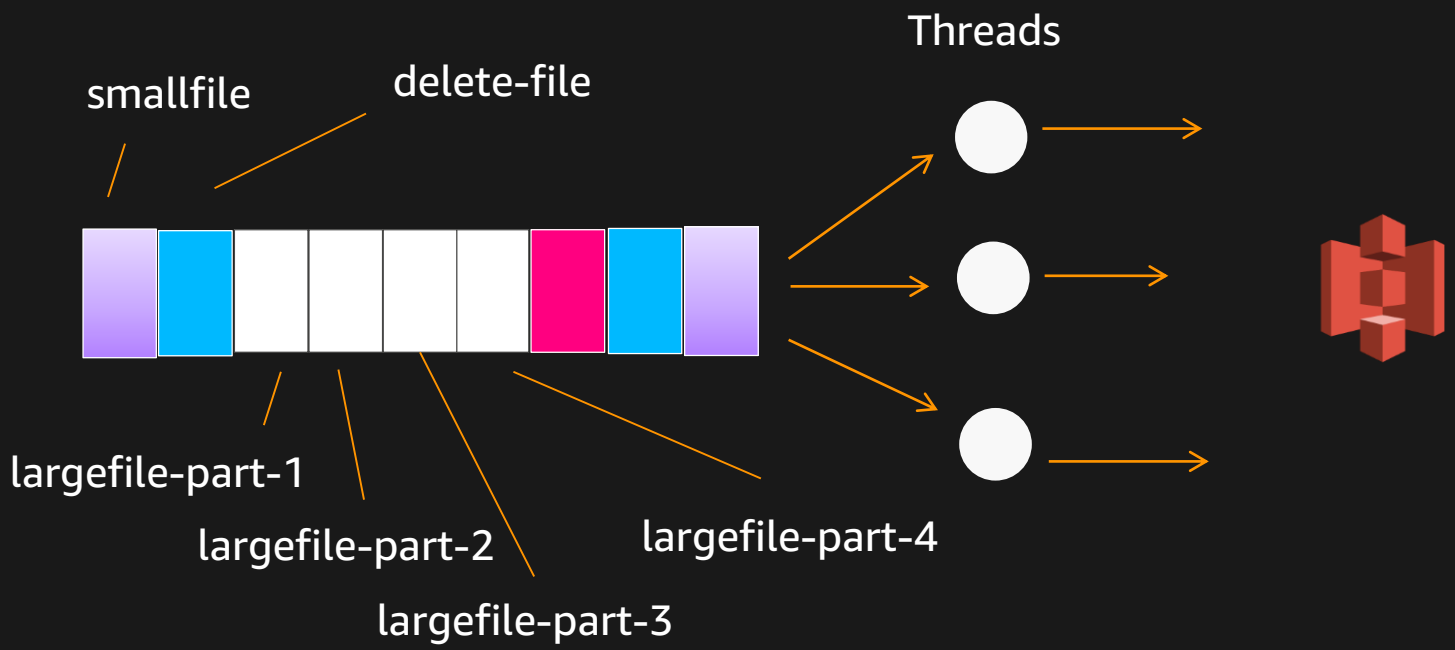
S3 Commands

- Copy
- Sync

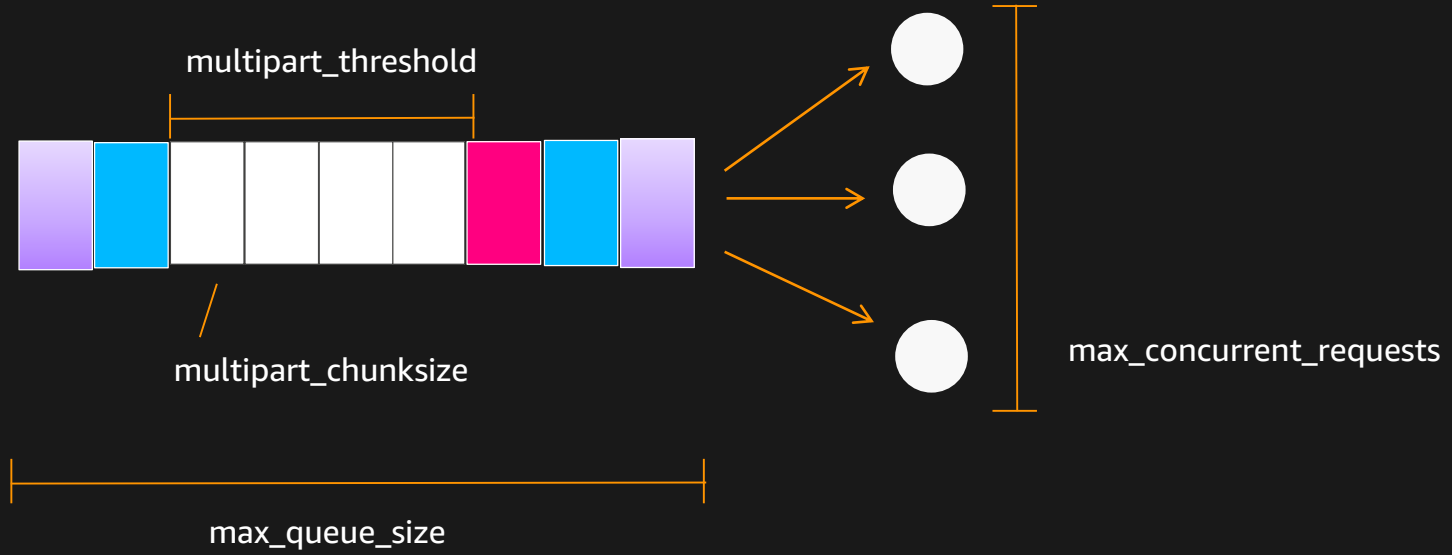
```
aws s3 cp local s3://bucket/key
aws s3 sync . s3://bucket/dir
```

```
aws s3 cp s3://src-bucket/key s3://dest-bucket/key
aws s3 sync s3://src-bucket s3://dest/bucket
```

aws s3 sync . s3://bucket/dir



```
$ aws configure set default.s3.max_concurrent_requests 20
$ aws configure set default.s3.max_queue_size 10000
$ aws configure set default.s3.multipart_threshold 64MB
$ aws configure set default.s3.multipart_chunksize 16MB
```



S3 Commands

http://docs.aws.amazon.com/cli/latest/topic/s3-config.html

```
$ aws help s3-config
```

Recap

- Many ways to install on multiple platforms
- Use tab completion, help commands and aws-shell to your advantage
- Take advantage of Roles and MFA
- Powerful parsing of CLI command returns with querying, filtering & jmespath
- Generate CLI Skeleton makes scripting with the CLI easier
- Use S3 commands for data migration / backups

Learn More

- Go to or watch on YouTube
    - DEV307 - AWS CLI: 2017 and Beyond
- Go to
    - https://aws.amazon.com/cli/
- Connect with other AWS developers on the CLI community forum
- Find examples and more in the User Guide

AWS re:Invent

THANK YOU!

AWS re:Invent

aws