AWSCLI COMMAND

The AWS Command Line Interface (CLI) is a unified tool to manage your AWS services

With just one tool to download and configure, you can control multiple AWS services from the command line and automate them through scripts.

Developing and performing AWS tasks against AWS can be done in several ways.

* Using the AWS CLI on our local computer
* Using the AWS CLI on our EC2 instance
* Using the AWS SDK on local or EC2 instance

CLI Installation

Install CLI on windows using **AWSCLI.MSI**

Install CLI on Linux Machine

* yum install -y python
* python --version
* yum install wget
* wget https://pypi.python.org/packages/source/s/setuptools/setuptools-7.0.tar.gz
* tar xvf setuptools-7.0.tar.gz
* cd setuptools-7.0
* python setup.py install
* wget https://bootstrap.pypa.io/get-pip.py
* python get-pip.py
* pip install awscli
* aws –-version
* AWS will store these credentials and configuration details in two

separate files named

* ~/. aws/credentials and ~/. aws/config, respectively
* We could run ‘AWS Configure’ on EC2 just like we did (and all it works).
* But. It’s SUPER INSECURE.
* NEVER NEVER EVER PUT YOUR PERSONAL CREDENTIALS ON EC2.
* Your PERSONAL credentials are PERSONAL and only belongs on your
* PERSONAL computer.
* IF the EC2 is compromised, so is your personal account.
* If the EC2 is shared, other people may perform AWS actions while
* impersonating you.
* For EC2, there is a better way… its called AWS IAM ROLE

**The Right Way**

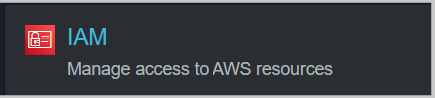
* IAM Roles can be attached to EC2 instances
* IAM Roles can come with a policy authorizing exactly what the EC2
* instance should be able to do.
* This is the best practice on AWS, you should do it 100%

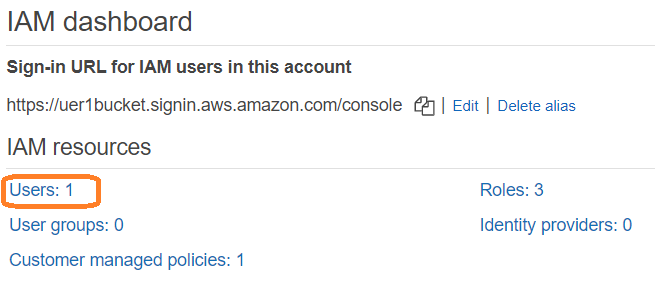
**AWS EC2 Instance Metadata**

* The URL curl <http://169.254.169.254/latest/meta-dat>
* You can retrieve the IAM Role name from the meta-data, but you
* CANNOT retrieve the IAM policy.
* Meta-data = Info about EC2 instance
* User-data = launch script of the EC2 instance.

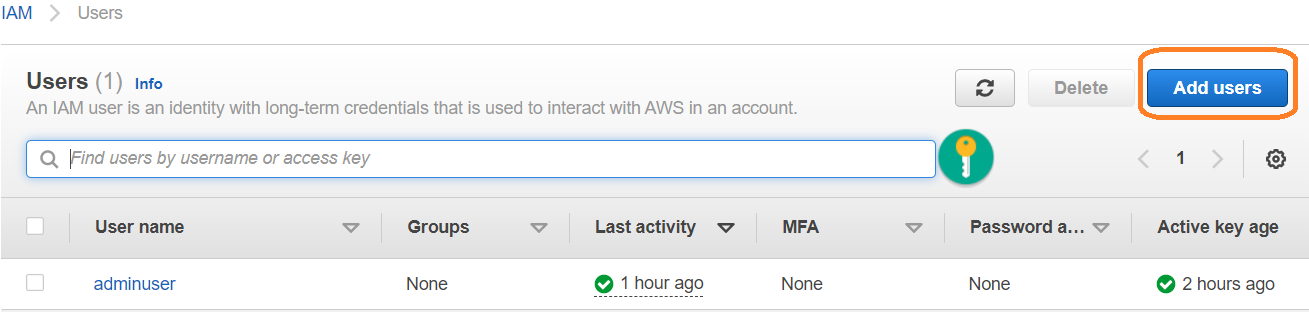
**CREATE USER:**

Go to search bar and enter “IAM”

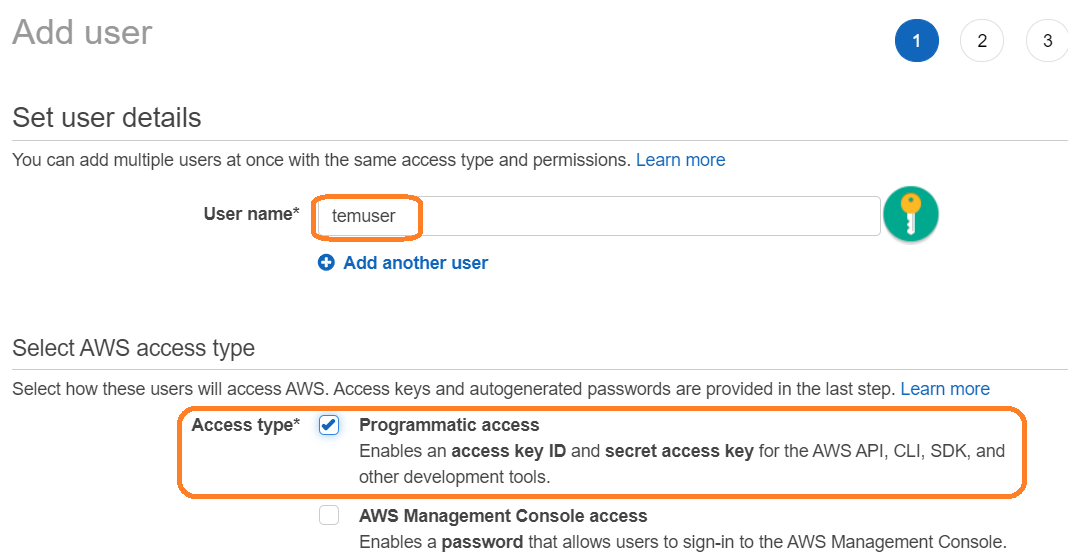




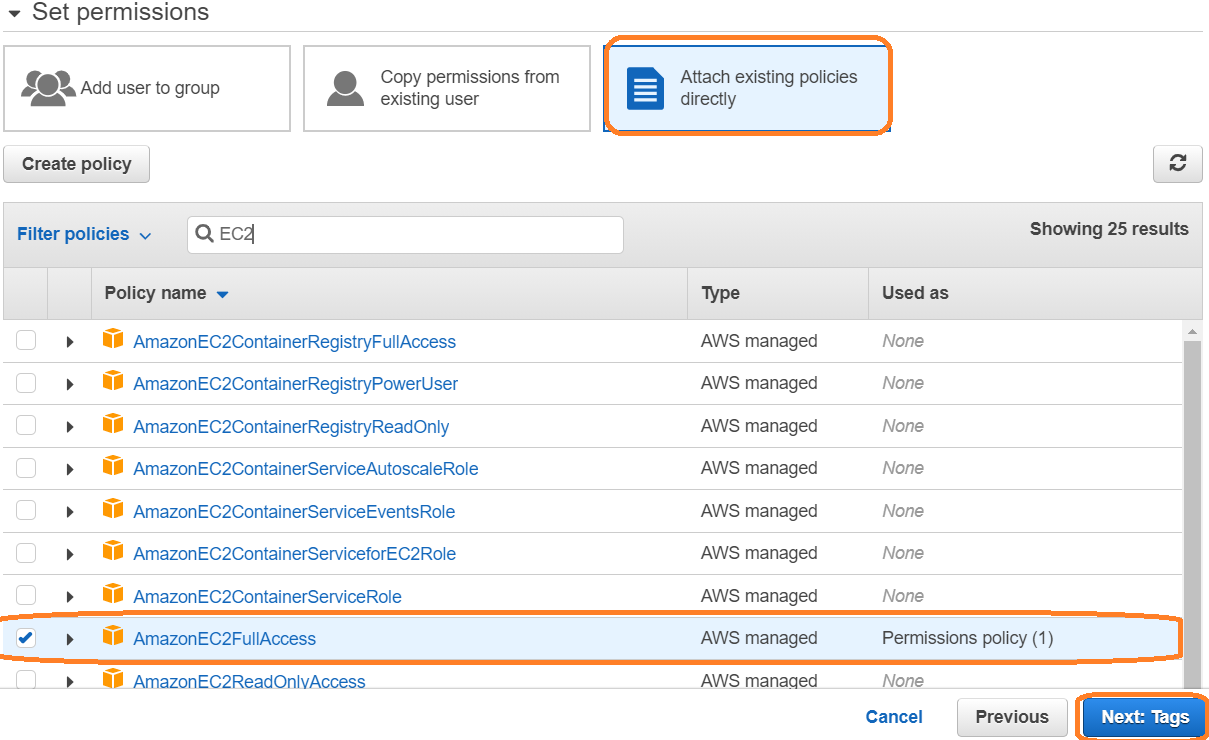
**Click on “Users”**

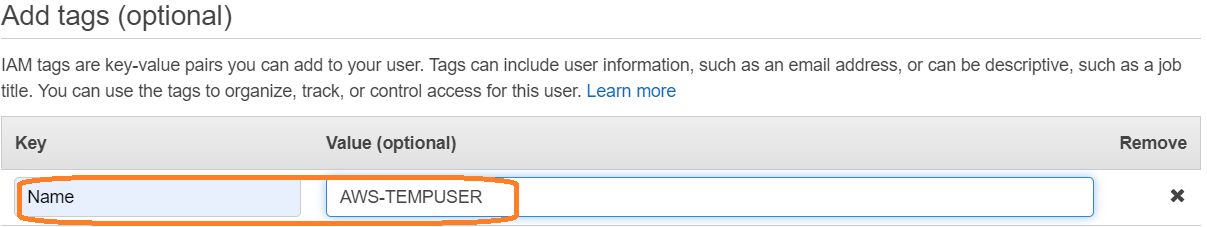
****

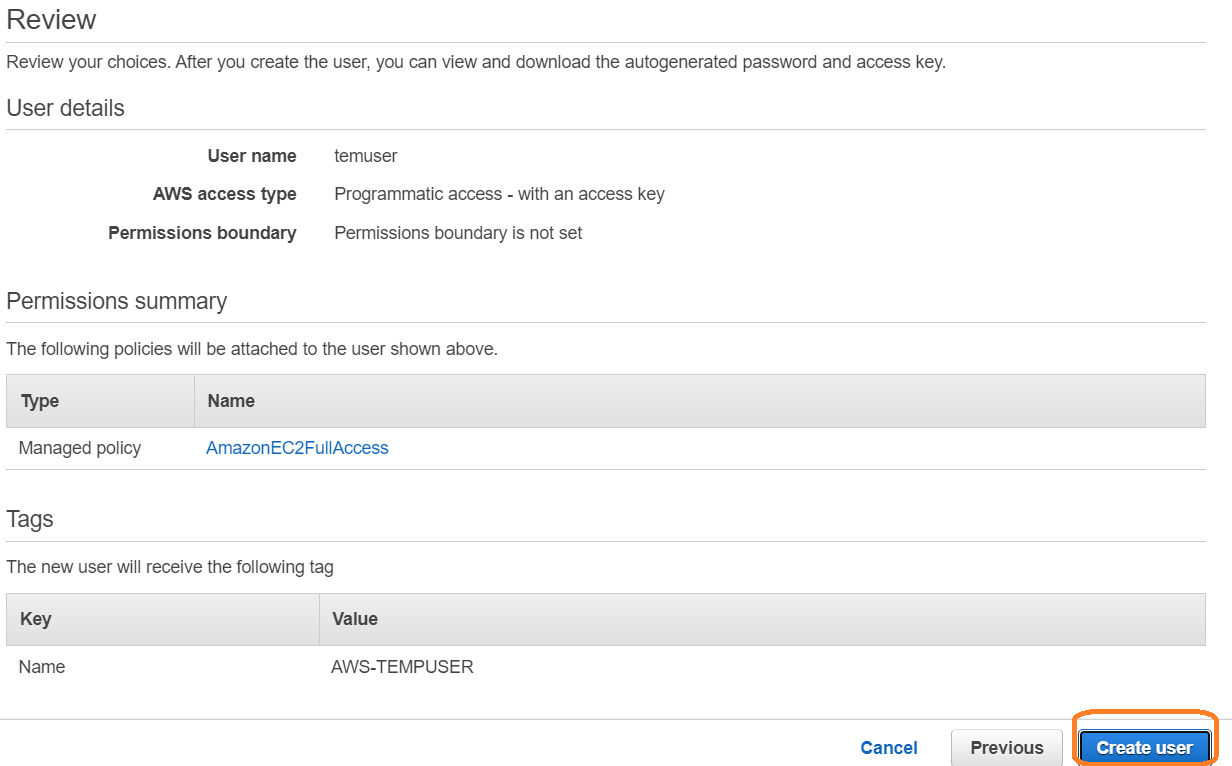
**Click on “Add user” button**

****

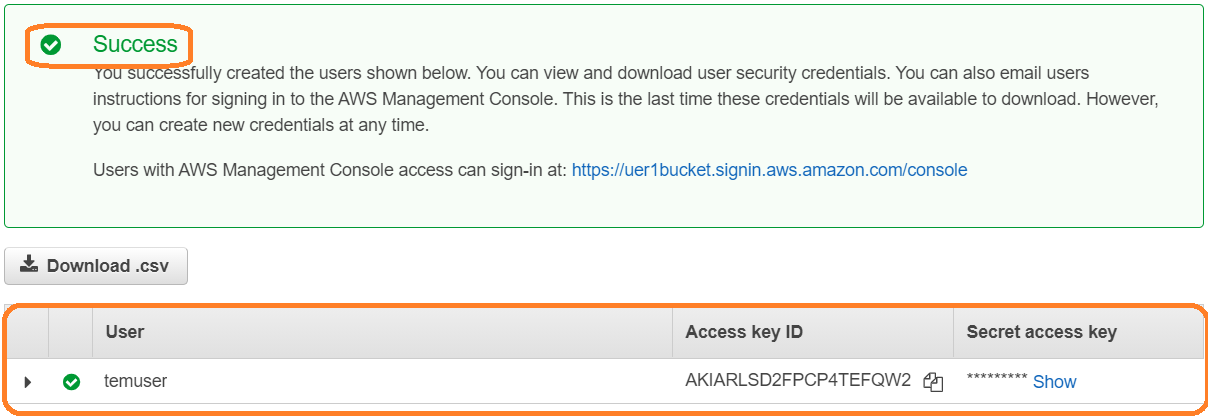
**Click on “Next:Permissions”**

****

****

****

Click “**Create User**”



**Note:**

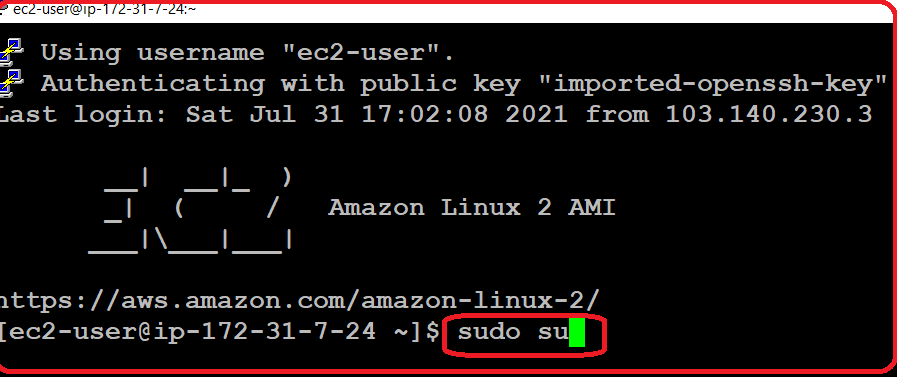
We need to copy Access key ID and Secret access key in notepad for future reference

**Access key ID**: AKIARLSD2FPCP4TEFQW2

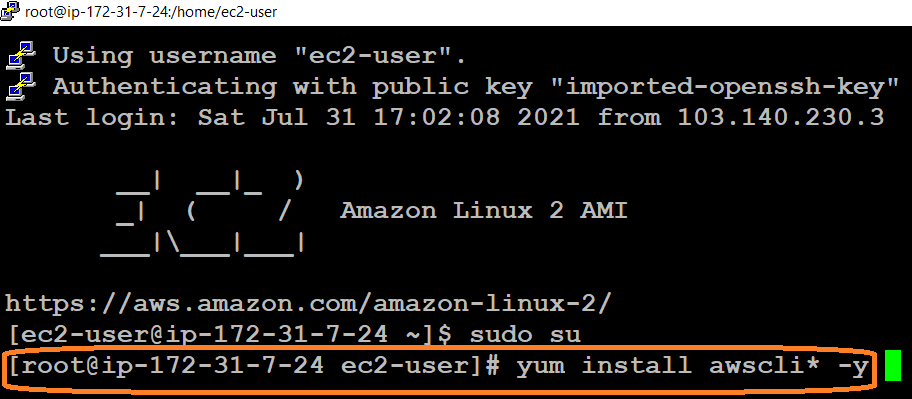
**Secret access key**: rw2a8RMxHwuGBQUC7PZrnVGh2mJVoZJgk/z0VhzY

We need to CREATE EC2 instance and login

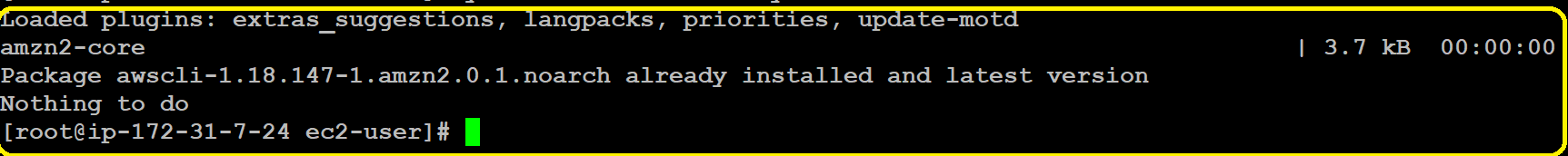
Login as a root user:



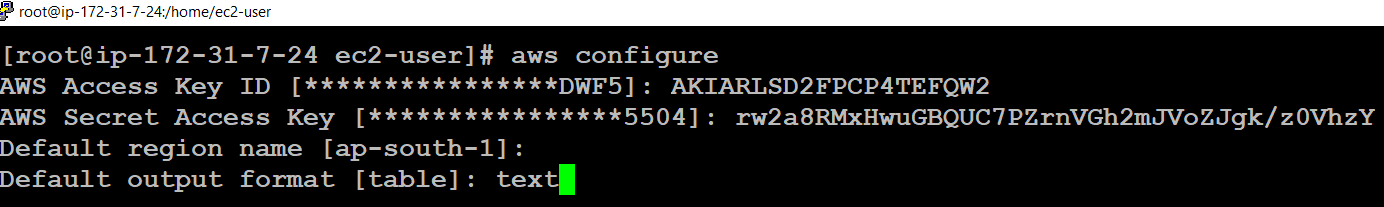
Verify **awscli** install or not



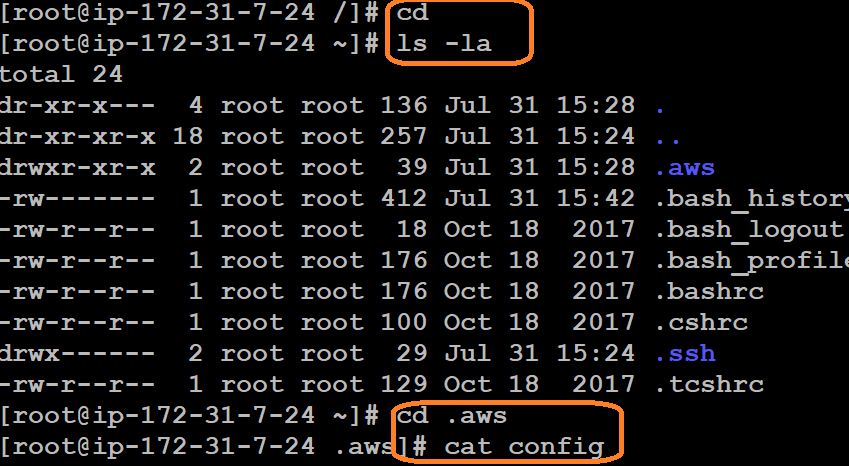
We will get below message if already installed

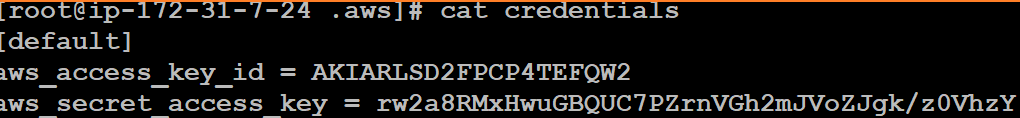


We need login in to user (i.e temuser) using below command



**aws ec2 describe-instances**



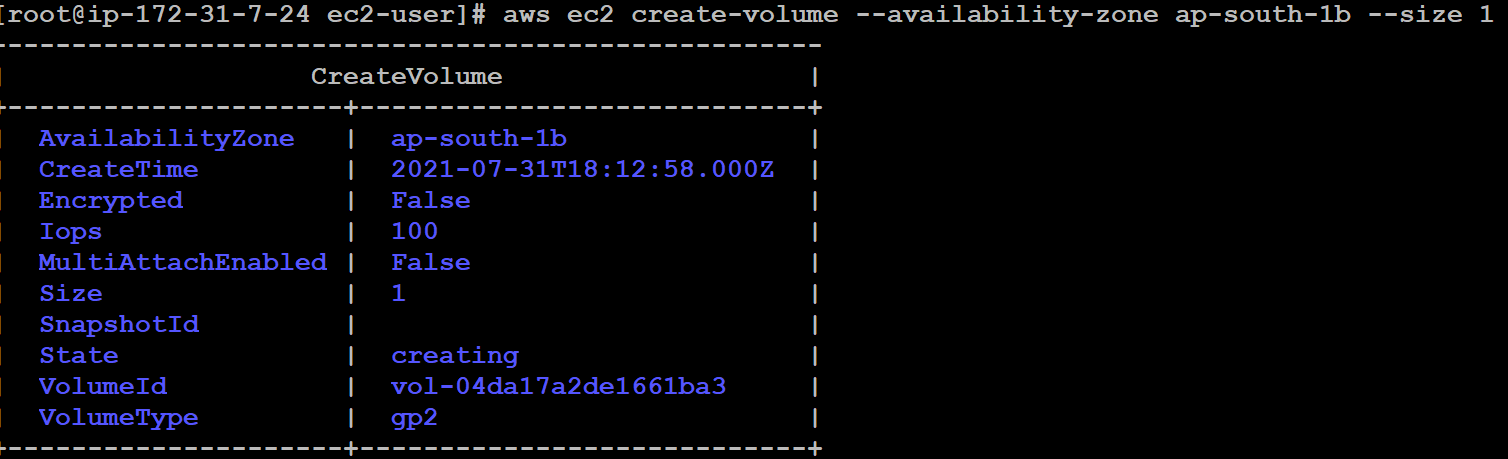


**CREATE-VOLUME:**

**aws ec2 create-volume --availability-zone ap-south-1b**

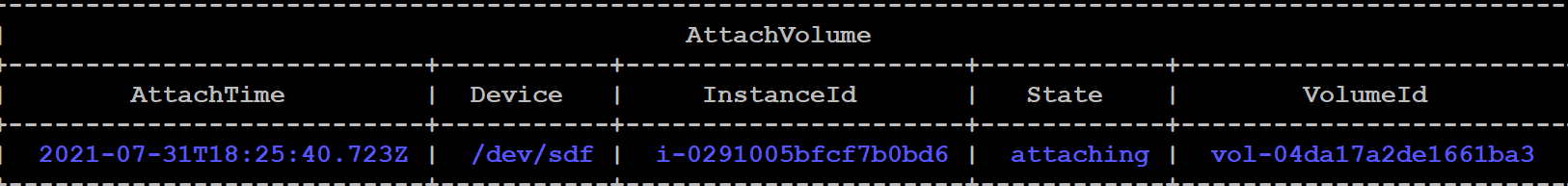
**Exception:**

**An error occurred (MissingParameter) when calling the CreateVolume operation: The request must contain the parameter size/snapshot**

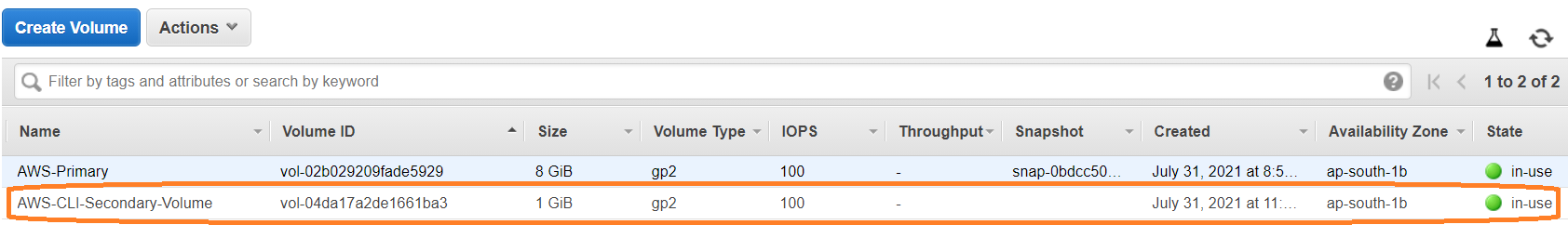
****

**Attach Volume to instance:**

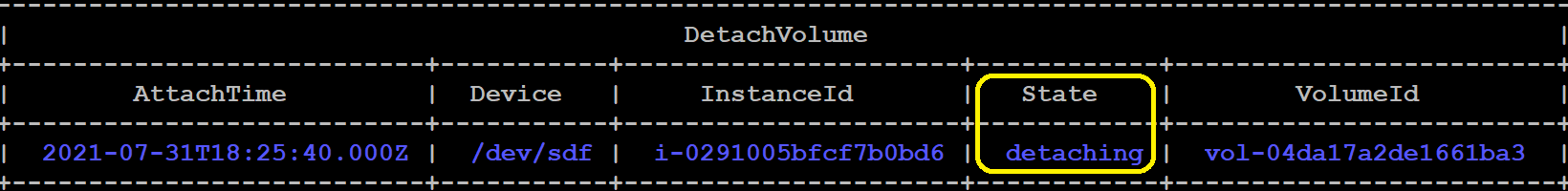
**aws ec2 attach-volume --volume-id vol-04da17a2de1661ba3 --instance-id i-0291005bfcf7b0bd6 --device /dev/sdf**

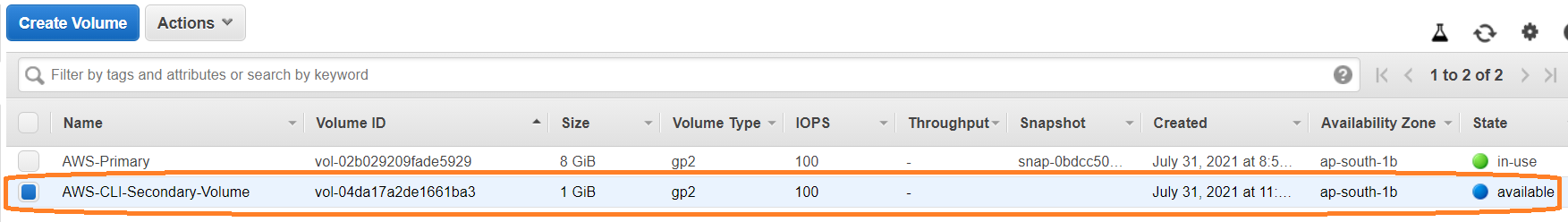


Volume has been attached with **in-use** status



**aws ec2 detach-volume --volume-id vol-04da17a2de1661ba3 --instance-id i-0291005bfcf7b0bd6**





Volume has been attached with **available** status

**aws ec2 delete-volume --volume-id vol-04da17a2de1661ba3**

IMAGE

* **aws ec2 create-image --instance-id <instance-id> --name "<image-name>" --description "<description>"**
* **aws ec2 deregister-image --image-id <image-id>**
* **aws ec2 delete-snapshot --snapshot-id <snapshot-id>**
* **curl** <http://169.254.169.254/latest/meta-data/>

S3 COMMANDS

* **aws s3 mb s3://mybucket**
* **aws s3 mb s3://mybucket --region us-west-1**
* **aws s3 mv test.txt s3://mybucket/test2.txt**
* **aws s3 mv s3://mybucket/test.txt s3://mybucket/test2.txt**
* **aws s3 mv s3://mybucket/test.txt test2.txt**
* **aws s3 mv s3://mybucket/test.txt s3://mybucket2/ (retain orginal name)**
* **aws s3 mv s3://mybucket . --recursive**
* **aws s3 mv myDir s3://mybucket/ --recursive --exclude "\*.jpg"**
* **aws s3 mv s3://mybucket/ s3://mybucket2/ --recursive --exclude "mybucket/another/\*"**
* **aws s3 ls**
* **aws s3 ls s3://mybucket**
* **aws s3 ls s3://mybucket --recursive**
* **aws s3 ls s3://mybucket --recursive --human-readable --summarize**

**Copying a local file to S3**

* aws s3 cp test.txt s3://mybucket/test2.txt

**Copying an S3 object to a local file**

* aws s3 cp s3://myreya/text.txt text.txt

**Copying an S3 object from one bucket to another**

* aws s3 cp s3://mybucket/test.txt s3://mybucket2/

**Recursively copying S3 objects to a local directory**

* aws s3 cp s3://mybucket . --recursive

**Recursively copying local files to S3**

* aws s3 cp myDir s3://myreya/ --recursive --exclude "\*.jpg"

**Copying a file from S3 to S3**

* aws s3 cp s3://mybucket/test.txt s3://mybucket/test2.txt

**To create a pre-signed URL with the default one hour lifetime that links to an object in an S3 bucket**

* aws s3 presign s3://awsexamplebucket/test2.txt

**To create a pre-signed URL with a custom lifetime that links to an object in an S3 bucket**

* aws s3 presign s3://awsexamplebucket/test2.txt --expires-in 604800

**rm**

* aws s3 rm s3://mybucket/test2.txt
* aws s3 rm s3://mybucket --recursive
* aws s3 rm s3://mybucket/ --recursive --exclude "\*.jpg"

**Remove buckets**

* aws s3 rb s3://mybucket
* aws s3 rb s3://mybucket –force

**Sync**

* aws s3 sync s3://mybucket s3://mybucket2
* aws s3 sync s3://mybucket
* aws s3 sync s3://mybucket .
* aws s3 sync . s3://mybucket --delete (delete additional file in s3 if that is not in local)
* **aws s3 website s3://my-bucket/ --index-document index.html --error-document error.html**
* (https://docs.aws.amazon.com/cli/latest/reference/s3/sync.html)

LOG

* aws logs create-log-stream --log-group-name my-logs --log-stream-name 20150601

SNS

* aws sns create-topic \ --name my-topic
* aws sns delete-topic \ --topic-arn "arn:aws:sns:us-west-2:123456789012:my-topic"
* aws sns list-topics

Subscription

* aws sns **subscribe** \

--topic-arn arn:aws:sns:us-west-2:123456789012:my-topic \

--protocol email \

--notification-endpoint [my-email@example.com](mailto:my-email@example.com)

* aws sns **unsubscribe** \

--subscription-arn arn:aws:sns:us-west-2:0123456789012:my- topic:8a21d249-4329-4871-acc6-7be709c6ea7f

* aws sns **confirm-subscription** \

--topic-arn arn:aws:sns:us-west-2:123456789012:my-topic \

--token 2336412f37fb687f5d51e6e241d7700ae02f7124d8268910b858cb4db727ceeb2474bb937929d3bdd7ce5d0cce19325d036bc858d3c217426bcafa9c501a2cace93b83f1dd3797627467553dc438a8c974119496fc3eff026eaa5d14472ded6f9a5c43aec62d83ef5f49109da7176391

To create an HTTP load balancer

* aws elb **create-load-balancer** --load-balancer-name <<load-balancer-name>> --listeners "Protocol=HTTP,LoadBalancerPort=80,InstanceProtocol=HTTP,InstancePort=80" --subnets subnet-15aaab61 --security-groups <<sg-name>>