AWS CLI CheatSheet

install AWS CLI

```
sudo apt-get install -y python-dev python-pip
sudo pip install awscli
aws --version
aws configure
```

Cloudtrail - Logging and Auditing

```
# list all trails
aws cloudtrail describe-trails

# list all S3 buckets
aws s3 ls

# create a new trail
aws cloudtrail create-subscription \
    --name awslog \
    --s3-new-bucket awslog2016

# list the names of all trails
aws cloudtrail describe-trails --output text | cut -f 8

# get the status of a trail
aws cloudtrail get-trail-status \
    --name awslog

# delete a trail
aws cloudtrail delete-trail \
    --name awslog

# delete the S3 bucket of a trail
aws s3 rb s3://awslog2016 --force

# add tags to a trail, up to 10 tags
aws cloudtrail add-tags \
    --resource-id awslog \
    --tags-list "Key=log-type,Value=all"

# list the tags of a trail
aws cloudtrail list-tags \
    --resource-id-list

# remove a tag from a trail
aws cloudtrail remove-tags \
    --resource-id awslog \
    --tags-list "Key=log-type,Value=all"
```

IAM

Users

```
# list all user's info
aws iam list-users

# list all user's usernames
aws iam list-users --output text | cut -f 6

# list current user's info
aws iam get-user

# list current user's access keys
aws iam list-access-keys

# crate new user
aws iam create-user \
    --user-name aws-admin2

# create multiple new users, from a file
allUsers=$(cat ./user-names.txt)
for userName in $allUsers; do
    aws iam create-user \
        --user-name $userName
done

# list all users
aws iam list-users --no-paginate

# get a specific user's info
aws iam get-user \
    --user-name aws-admin2

# delete one user
aws iam delete-user \
    --user-name aws-admin2

# delete all users
# allUsers=$(aws iam list-users --output text | cut -f 6);
allUsers=$(cat ./user-names.txt)
for userName in $allUsers; do
    aws iam delete-user \
        --user-name $userName
done
```

Password policy

```
# list policy
aws iam get-account-password-policy

# set policy
aws iam update-account-password-policy \
        --minimum-password-length 12 \
        --require-symbols \
        --require-numbers \
        --require-uppercase-characters \
        --require-lowercase-characters \
```

```
        --allow-users-to-change-password

# delete policy
aws iam delete-account-password-policy
```

Access Keys

```
# list all access keys
aws iam list-access-keys

# list access keys of a specific user
aws iam list-access-keys \
    --user-name aws-admin2

# create a new access key
aws iam create-access-key \
    --user-name aws-admin2 \
    --output text | tee aws-admin2.txt

# list last access time of an access key
aws iam get-access-key-last-used \
    --access-key-id AKIAINA6AJZY4EXAMPLE

# deactivate an acccss key
aws iam update-access-key \
    --access-key-id AKIAI44QH8DHBEXAMPLE \
    --status Inactive \
    --user-name aws-admin2

# delete an access key
aws iam delete-access-key \
    --access-key-id AKIAI44QH8DHBEXAMPLE \
    --user-name aws-admin2
```

Groups, Policies, Managed Policies

```
# list all groups
aws iam list-groups

# create a group
aws iam create-group --group-name FullAdmins

# delete a group
aws iam delete-group \
    --group-name FullAdmins

# list all policies
aws iam list-policies

# get a specific policy
aws iam get-policy \
    --policy-arn <value>

# list all users, groups, and roles, for a given policy
aws iam list-entities-for-policy \
    --policy-arn <value>

# list policies, for a given group
aws iam list-attached-group-policies \
```

```
    --group-name FullAdmins

# add a policy to a group
aws iam attach-group-policy \
    --group-name FullAdmins \
    --policy-arn arn:aws:iam::aws:policy/AdministratorAccess

# add a user to a group
aws iam add-user-to-group \
    --group-name FullAdmins \
    --user-name aws-admin2

# list users, for a given group
aws iam get-group \
    --group-name FullAdmins

# list groups, for a given user
aws iam list-groups-for-user \
    --user-name aws-admin2

# remove a user from a group
aws iam remove-user-from-group \
    --group-name FullAdmins \
    --user-name aws-admin2

# remove a policy from a group
aws iam detach-group-policy \
    --group-name FullAdmins \
    --policy-arn arn:aws:iam::aws:policy/AdministratorAccess

# delete a group
aws iam delete-group \
    --group-name FullAdmins
```

S3

```
# list existing S3 buckets
aws s3 ls

# create a bucket name, using the current date timestamp
bucket_name=test_$(date "+%Y-%m-%d_%H-%M-%S")
echo $bucket_name

# create a public facing bucket
aws s3api create-bucket --acl "public-read-write" --bucket $bucket_name

# verify bucket was created
aws s3 ls | grep $bucket_name

# check for public facing s3 buckets (should show the bucket name you
created)

aws s3api list-buckets --query 'Buckets[*].[Name]' --output text | xargs -I
{} bash -c 'if [[ $(aws s3api get-bucket-acl --bucket {} --query
'"'"'Grants[?Grantee.URI==`http://acs.amazonaws.com/groups/global/AllUsers`
&& Permission==`READ`]'"'"' --output text) ]]; then echo {} ; fi'

# check for public facing s3 buckets, updated them to be private
```

```
aws s3api list-buckets --query 'Buckets[*].[Name]' --output text | xargs -I
{} bash -c 'if [[ $(aws s3api get-bucket-acl --bucket {} --query
'"'"'Grants[?Grantee.URI==`http://acs.amazonaws.com/groups/global/AllUsers`
&& Permission==`READ`]'"'"' --output text) ]]; then aws s3api put-bucket-
acl --acl "private" --bucket {} ; fi'

# check for public facing s3 buckets (should be empty)

aws s3api list-buckets --query 'Buckets[*].[Name]' --output text | xargs -I
{} bash -c 'if [[ $(aws s3api get-bucket-acl --bucket {} --query
'"'"'Grants[?Grantee.URI==`http://acs.amazonaws.com/groups/global/AllUsers`
&& Permission==`READ`]'"'"' --output text) ]]; then echo {} ; fi'
```

EC2

keypairs

```
# list all keypairs
aws ec2 describe-key-pairs

# create a keypair
aws ec2 create-key-pair \
    --key-name <value> --output text

# create a new local private / public keypair, using RSA 4096-bit
ssh-keygen -t rsa -b 4096

# import an existing keypair
aws ec2 import-key-pair \
    --key-name keyname_test \
    --public-key-material file:///home/apollo/id_rsa.pub

# delete a keypair
aws ec2 delete-key-pair \
    --key-name <value>
```

Security Groups

```
# list all security groups
aws ec2 describe-security-groups

# create a security group
aws ec2 create-security-group \
    --vpc-id vpc-1a2b3c4d \
    --group-name web-access \
    --description "web access"

# list details about a securty group
aws ec2 describe-security-groups \
    --group-id sg-0000000

# open port 80, for everyone
aws ec2 authorize-security-group-ingress \
    --group-id sg-0000000 \
    --protocol tcp \
    --port 80 \
```

```
    --cidr 0.0.0.0/24

# get my public ip
my_ip=$(dig +short myip.opendns.com @resolver1.opendns.com);
echo $my_ip

# open port 22, just for my ip
aws ec2 authorize-security-group-ingress \
    --group-id sg-0000000 \
    --protocol tcp \
    --port 80 \
    --cidr $my_ip/24

# remove a firewall rule from a group
aws ec2 revoke-security-group-ingress \
    --group-id sg-0000000 \
    --protocol tcp \
    --port 80 \
    --cidr 0.0.0.0/24

# delete a security group
aws ec2 delete-security-group \
    --group-id sg-00000000
```

Images

```
# list all private AMI's, ImageId and Name tags
aws ec2 describe-images --filter "Name=is-public,Values=false" \
    --query 'Images[].[ImageId, Name]' \
    --output text | sort -k2

# delete an AMI, by ImageId
aws ec2 deregister-image --image-id ami-00000000
```

Instances

```
# list all instances (running, and not running)

aws ec2 describe-instances


# list all instances running
aws ec2 describe-instances --filters Name=instance-state-
name,Values=running

# create a new instance
aws ec2 run-instances \
    --image-id ami-f0e7d19a \
    --instance-type t2.micro \
    --security-group-ids sg-00000000 \
    --dry-run

# stop an instance
instances.html
aws ec2 terminate-instances \
    --instance-ids <instance_id>
```

```
# list status of all instances
aws ec2 describe-instance-status

# list status of a specific instance
aws ec2 describe-instance-status \
    --instance-ids <instance_id>

# list all running instance, Name tag and Public IP Address
aws ec2 describe-instances \
  --filters Name=instance-state-name,Values=running \
  --query 'Reservations[].Instances[].[PublicIpAddress,
Tags[?Key==`Name`].Value | [0] ]' \
  --output text | sort -k2
```

Tags

```
# list the tags of an instance
aws ec2 describe-tags

# add a tag to an instance
aws ec2 create-tags \
    --resources "ami-1a2b3c4d" \
    --tags Key=name,Value=debian

# delete a tag on an instance
aws ec2 delete-tags \
    --resources "ami-1a2b3c4d" \
    --tags Key=Name,Value=
```

Cloudwatch

Log Groups

create a group

```
aws logs create-log-group \
        --log-group-name "DefaultGroup"
```

list all log groups

```
aws logs describe-log-groups

aws logs describe-log-groups \
        --log-group-name-prefix "Default"
```

delete a group

```
aws logs delete-log-group \
        --log-group-name "DefaultGroup"
```

Log Streams

```
# Log group names can be between 1 and 512 characters long. Allowed
# characters include a-z, A-Z, 0-9, '_' (underscore), '-' (hyphen),
```

```
# '/' (forward slash), and '.' (period).

# create a log stream
aws logs create-log-stream \
        --log-group-name "DefaultGroup" \
        --log-stream-name "syslog"

# list details on a log stream
aws logs describe-log-streams \
        --log-group-name "syslog"

aws logs describe-log-streams \
        --log-stream-name-prefix "syslog"

# delete a log stream
aws logs delete-log-stream \
        --log-group-name "DefaultGroup" \
        --log-stream-name "Default Stream"
```