AWS-CLI Cheatsheet

Table of content

Installation

Ref: https://docs.aws.amazon.com/es_es/cli/latest/userguide/cli-chap-welcome.html

Function	Command
Install awscli	`pip3 install awscli --upgrade --user`
Configuring awscli	`aws configure`

EC2

EC2-UTIL: List all instances

```
aws ec2 describe-instances
```

EC2-UTIL: List specific fields of all instances

```
aws ec2 describe-instances \
    --query "Reservations[].Instances[].[InstanceId, PublicIpAddress, Tags[?Key=='Name']|[0].Value]"
```

EC2-UTIL: List all instances of a product

```
aws ec2 describe-instances \ --filter "Name=tag:Name,Values=latch*" \ --query "Reservations[].Instances[].[InstanceId, PublicIpAddress, Tags[?Key=='Name']|[0].Value]"
```

EC2-UTIL: List all stopped instances

```
aws ec2 describe-instances \
    --filters Name=instance-state-name,Values=stopped
```

EC2-UTIL: List all stopped instances with ElasticIP

aws ec2 describe-instances \ --query "Reservations[*].Instances[*].PublicIpAddress" \ --filters Name=instance-state-name,Values=stopped ```

EC2-UTIL: List all snapshots in the date specified

```
aws ec2 describe-snapshots \
    --filters Name=start-time,Values=2019-01-05*
```

EC2-SEC: List all snapshots without encryption

aws ec2 describe-snapshots \ --filters "Name=encrypted,Values=false"

EC2-SEC: List SecurityGroups with SSH open to Internet

aws ec2 describe-security-groups \ --filters Name=ip-permission.from-port,Values=22 Name=ip-permission.to-port,Values=22 Name=ip-permission.cidr,Values='0.0.0.0/0' \ --query 'SecurityGroups[*].{Name:GroupName}' \ --output table

---

IAM

IAM-UTIL: List certificates

aws iam list-server-certificates

IAM-UTIL: List policies

aws iam list-policies

IAM-UTIL: List policies attached to a group

aws iam list-attached-group-policies \ --group-name ec2-Users

IAM-UTIL: List users of a group

aws iam get-group \ --group-name ec2-users \ --query "Users[]"

IAM-UTIL: List groups of a user

aws iam list-groups-for-user \ --user-name aws-admin2

IAM-SEC: Access Keys Rotation

aws iam list-access-keys \ --user-name aws-admin2 \ --query 'AccessKeyMetadata[?Status=="Active"].[CreateDate]'

IAM-SEC: User with MFA enabled

if [[$(aws iam list-mfa-devices --user-name root --output text)]]; then echo "MFA Enabled"; else echo "MFA Disabled";fi

```
---

## S3

#### S3-UTIL: List buckets
```

aws s3 ls

```
#### S3-UTIL: List bucket objects
```

aws s3api list-objects \ --bucket pre-cdo-web-resources \ --query 'Contents[].{Key: Key, Size: Size}' \ --output text

```
#### S3-SEC: Bucket with public READ access
```

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'

```
#### S3-SEC: Bucket with public WRITE access
```

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== `WRITE`]'"'"' –output text)]]; then echo {} ; fi'

```
#### S3-SEC: Bucket with public FULL_CONTROL access
```

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'

```
---

## RDS

#### RDS-UTIL: List databases
```

aws rds describe-db-instances \ --query 'DBInstances[].DBInstanceIdentifier'

```
#### RDS-SEC: List Databases without DeletionProtection enabled
```

aws rds describe-db-instances \ --query 'DBInstances[*].[DBInstanceIdentifier]' \ *--output text \ | xargs -I {} bash -c 'if [[$(aws rds describe-db-instances --db-instance-identifier {} --query '"'"'DBInstances[*]*.DeletionProtection'"'"' --output text) == False]]; then echo {} ; fi'

```
#### RDS-SEC: List Public Databases
```

aws rds describe-db-instances \ --query 'DBInstances[?PubliclyAccessible=="true"].[DBInstanceIdentifier,Endpoint.Address]' ```
```
```