

**AWS Cheat Sheet**

about

# Create a Security Group & Adding Inbound rules

**# aws ec2 create-security-group --group-name** <security grp Name> **--description** "<Description>"

**# curl https://checkip.amazonaws.com**

**# aws ec2 authorize-security-group-ingress --group-id** <security group Id> **--protocol tcp --port** <port Number> **--cidr** <ip address>

**# aws ec2 authorize-security-group-ingress --group-id** <security grp Id>**--protocol tcp --port** 22-8000 **--cidr** 0.0.0.0/0

**# aws ec2 describe-security-groups --group-names** <security grp Name>

Delete your security group: **# aws ec2 delete-security-group --group-name** <security grp Name>

# Create a key pair

**aws ec2 create-key-pair --key-**name <keypair-Name> **--query 'KeyMaterial'** **--output text >** <keypair-Name.pem>

*Delete a key pair:*

**aws ec2 delete-key-pair --key-name** <keypair-Name>

# Launch Instance

**# aws ec2 run-instances --image-id** <ami-Id> **--count 1 --instance-type** <type> **--key-name** <keypair-Name> **--security-groups** <security grp Name>

Add Tags:

**# *aws ec2 create-tags --resources*** *<Instance-Id>****--tags Key=Name,****Value****=****<value>*

Terminate your Instance:

***# aws ec2 terminate-instances --instance-ids*** *<Instance-Id>*

# Create a snapshot

**# aws ec2 create-snapshot --volume-id** <volume Id> **--description 'Prod backup' --**

**tag-specifications 'ResourceType=**snapshot**,Tags=[{Key=**Name**,Value=**<value>**},{Key=**Database**,Value=**Mysql**}]'**

List Snapshots:

***# aws ec2 describe-snapshots --owner-ids self --query "Snapshots[\*].{ID:SnapshotId,StartTime:StartTime,VolumeId:VolumeId}" --output table***

Delete Snapshot: ***# aws ec2 delete-snapshot --snapshot-id*** <snapshot Id>

# Create EBS Volume

**# aws ec2 create-volume --volume-type** <volume type> **--size** <size in number> **--encrypted{optional} --availability-zone** <zone>

**# aws ec2 create-volume --snapshot-id** <snapshot Id> **--availability-zone** <Av-Zone>

Add Tags:

**# aws ec2 create-tags --resources** <volume-id> **--tags Key=Name,**Value**=**<value>

Identify the Volume ID and Instance ID:

***aws ec2 describe-volumes --query "Volumes[\*].{ID:VolumeId,State:State,Instance:Attachments[0].InstanceId}" --output table***

Attach the Volume:

**aws ec2 attach-volume --volume-id** <volume-id> **--instance-id** <Instance-Id> **--device** /dev/sdf

Detach the Volume: **# aws ec2 detach-volume --volume-id** <volume-id>

Delete a Volume:  **# aws ec2 delete-volume --volume-id** <volume-id>

# Create *Launch Template*

**#** **aws ec2 create-launch-template --launch-template-name** WebTemplate **--version-description "Version 1" --launch-template-data '{"ImageId":"**ami-04c41671613c335d1**","InstanceType":"**t2.micro**","KeyName":"AhmedKey", "SecurityGroupIds":["**sg-00b2af2e5b584c148**"], "TagSpecifications":[{"ResourceType":"instance","Tags":[{"Key":"**Name**","Value": "**MyLaunchTemplateInstance**"}]}]}'**

Launch Ec2 from Template:  **# aws ec2 run-instances --launch-template LaunchTemplateName =** WebTemplate **,Version =1 --tag-specifications 'ResourceType=instance ,Tags= [{Key=Name,Value =**MyNewInstanceName**}]'**

Delete Launch Template: **# aws ec2 delete-launch-template --launch-template-id** < template id> **--region** <region>

# Create an AMI

**# aws ec2 create-image --instance-id** i-0a4193c5f4f8a834a **--name** "WebImg" **--description** "An ami of Ec2 instance for website on centos 7"

# Mount an EBS Volume

Identify the Attached Volume: lsblk

Create a Filesystem on the Volume: sudo mkfs -t ext4 /dev/xvdf

Create a Mount Point: sudo mkdir /mnt/new\_volume

Mount the Volume: **sudo mount /dev/xvdf /mnt/new\_volume**

Update /etc/fstab for Automatic Mounting: **sudo bash -c 'echo "/dev/xvdf /mnt/new\_volume ext4 defaults,nofail 0 2" >> /etc/fstab'**

Verify the Mount: df -h