AWS EC2 CLI Cheat sheet

BlackPerl

AWS EC2

AWS CLI commands to create EC2

```
aws ec2 run-instances --image-id <AMI_ID> --instance-type <INSTANCE_TYPE> --key-name <KEY_PAIR_NAME> --subnet-id
<SUBNET_ID> --security-group-ids <SECURITY_GROUP_ID>

        #Replace the placeholders:
                <AMI_ID>: The ID of the Amazon Machine Image you want to use.
                <INSTANCE_TYPE>: The EC2 instance type you want to launch.
                <KEY_PAIR_NAME>: The name of the EC2 key pair you want to use for SSH/RDP access.
                <SUBNET_ID>: The ID of the subnet where you want to launch the instance.
                <SECURITY_GROUP_ID>: The ID of the security group you want to assign to the instance.
```

AWS CLI commands to create VPC (Virtual Private Cloud)

```
1. aws ec2 create-vpc --cidr-block <CIDR_BLOCK>
Replace <CIDR_BLOCK> with the desired IP range in CIDR notation.
2. aws ec2 create-tags --resources <VPC_ID> --tags Key=Name,Value=MyVPC
        #Replace <VPC_ID> with the VPC ID obtained from the previous step.
3. aws ec2 modify-vpc-attribute --vpc-id <VPC_ID> --enable-dns-support "{\"Value\":true}"
4. aws ec2 modify-vpc-attribute --vpc-id <VPC_ID> --enable-dns-hostnames "{\"Value\":true}"
5. aws ec2 create-subnet --vpc-id <VPC_ID> --cidr-block <SUBNET_CIDR_BLOCK> --availability-zone <AVAILABILITY_ZONE>
        #Replace <VPC_ID>, <SUBNET_CIDR_BLOCK>, and <AVAILABILITY_ZONE> with appropriate values.
6. aws ec2 create-internet-gateway
7. aws ec2 attach-internet-gateway --internet-gateway-id <IGW_ID> --vpc-id <VPC_ID>
8. aws ec2 create-route-table --vpc-id <VPC_ID>
9. aws ec2 associate-route-table --route-table-id <ROUTE_TABLE_ID> --subnet-id <SUBNET_ID>
10. aws ec2 create-route --route-table-id <ROUTE_TABLE_ID> --destination-cidr-block 0.0.0.0/0 --gateway-id <IGW_ID>
```

AWS CLI commands to create SG (Security Group)

```
1. aws ec2 create-security-group --group-name <GROUP_NAME> --description "<DESCRIPTION>" --vpc-id <VPC_ID>
        #Replace <VPC_ID> with the ID of the VPC where you want to create the security group.
2. aws ec2 authorize-security-group-ingress --group-id <GROUP_ID> --protocol <PROTOCOL> --port <PORT> --source-group
<SOURCE_GROUP_ID>
        #Use the authorize-security-group-ingress command to add inbound rules to the security group. Replace <GROUP_ID>
with the security group ID and customize the rule parameters as needed.
3. aws ec2 authorize-security-group-egress --group-id <GROUP_ID> --protocol <PROTOCOL> --port <PORT> --destination-group
<DESTINATION_GROUP_ID>
        #authorize-security-group-egress command to add outbound rules to the security group.
4. aws ec2 revoke-security-group-ingress --group-id <GROUP_ID> --protocol <PROTOCOL> --port <PORT> --source-group
<SOURCE_GROUP_ID>
aws ec2 revoke-security-group-egress --group-id <GROUP_ID> --protocol <PROTOCOL> --port <PORT> --destination-group
<DESTINATION_GROUP_ID>
        #use the revoke-security-group-ingress and revoke-security-group-egress commands respectively to update or modify
rules.
```

AWS CLI commands to create NACL (Network Access Control List)

```
1. aws ec2 create-network-acl --vpc-id <VPC_ID>
        #Replace <VPC_ID> with the ID of the VPC in which you want to create the NACL.
2. # Add inbound rule (example: allow SSH traffic)
aws ec2 create-network-acl-entry --network-acl-id <NACL_ID> --rule-number <RULE_NUMBER> --protocol <PROTOCOL> --rule-action
<ALLOW/DENY> --cidr-block <CIDR> --port-range From=<PORT>,To=<PORT>

# Add outbound rule (example: allow all outbound traffic)
aws ec2 create-network-acl-entry --network-acl-id <NACL_ID> --rule-number <RULE_NUMBER> --egress --protocol <PROTOCOL> --
rule-action <ALLOW/DENY> --cidr-block <CIDR>
        #Replace <NACL_ID> with the ID of the NACL you created.
3. aws ec2 associate-network-acl --network-acl-id <NACL_ID> --subnet-id <SUBNET_ID>
        #Replace <SUBNET_ID> with the ID of the subnet.
4. aws ec2 replace-network-acl-association --association-id <ASSOCIATION_ID> --network-acl-id <NACL_ID>
```

AWS CLI commands to add inbound and outbound rules to an existing Security Group (SG)

```
1. #Add Inbound Rule
aws ec2 authorize-security-group-ingress --group-id <GROUP_ID> --protocol <PROTOCOL> --port <PORT> --source-cidr <CIDR>
2. #Add Outbound Rule
aws ec2 authorize-security-group-egress --group-id <GROUP_ID> --protocol <PROTOCOL> --port <PORT> --destination-cidr <CIDR>
        **Note**
        ➢ Replace placeholders (<GROUP_ID>, <PROTOCOL>, <PORT>, <CIDR>) with actual values.
        ➢ <PROTOCOL>: Specify the protocol (e.g., tcp, udp, icmp, etc.).
        ➢ <PORT>: Specify the port number or range (e.g., 22 or 80-443).
        ➢ <CIDR>: Specify the CIDR block to allow traffic from or to.
```

AWS CLI commands to add inbound and outbound rules to an existing Network Access Control List (NACL)

```
1. #Add Inbound Rule
aws ec2 create-network-acl-entry --network-acl-id <NACL_ID> --rule-number <RULE_NUMBER> --protocol <PROTOCOL> --rule-action
<ALLOW/DENY> --cidr-block <CIDR_BLOCK> --port-range From=<FROM_PORT>,To=<TO_PORT>
2. #Add Outbound Rule
aws ec2 create-network-acl-entry --network-acl-id <NACL_ID> --rule-number <RULE_NUMBER> --egress --protocol <PROTOCOL> --
rule-action <ALLOW/DENY> --cidr-block <CIDR_BLOCK>
#Include the " –egress" flag to indicate an outbound rule.
        **Note**
        ➢ Replace placeholders (<NACL_ID>, <RULE_NUMBER>, <PROTOCOL>, <FROM_PORT>, <TO_PORT>, <CIDR_BLOCK>) with actual values.
        ➢ <PROTOCOL>: Specify the protocol (e.g., 6 for TCP, 17 for UDP, etc.).
        ➢ <FROM_PORT> and <TO_PORT>: Specify the port range for the rule.
        ➢ <CIDR_BLOCK>: Specify the CIDR block to allow traffic from/to.
```

To associate a Security Group (SG) with an existing EC2 instance using AWS CLI commands

```
1. aws ec2 modify-instance-attribute --instance-id <INSTANCE_ID> --groups <SG_ID_1> <SG_ID_2> ...
        #Replace <INSTANCE_ID> with the EC2 instance ID and <SG_ID_1>, <SG_ID_2>, etc. with the Security Group IDs you want to
associate.
```

To create a snapshot of an Amazon Elastic Block Store (EBS) volume attached to an EC2 instance using AWS CLI commands

```
1. #Identify the ID of the EBS volume for which you want to create a snapshot. You can find this information either in the AWS
Management Console or by using the "describe-volumes" AWS CLI command.
2. aws ec2 create-snapshot --volume-id <VOLUME_ID> --description "<SNAPSHOT_DESCRIPTION>"
        #Replace <VOLUME_ID> with the EBS volume ID and <SNAPSHOT_DESCRIPTION> with a description for the snapshot.
```

AWS CLI commands for managing EC2 instances

```
1. #Get information about one or more instances.
aws ec2 describe-instances [--instance-ids <INSTANCE_ID> ...]
2. #Start a stopped instance.
aws ec2 start-instances --instance-ids <INSTANCE_ID> ...
3. #Stop a running instance.
aws ec2 stop-instances --instance-ids <INSTANCE_ID> ...
4. #Terminate an instance (permanently delete).
aws ec2 terminate-instances --instance-ids <INSTANCE_ID> ...
5. # create a new instance
aws ec2 run-instances \
  --image-id ami-f0e7d19a \
  --instance-type t2.micro \
  --security-group-ids sg-00000000 \
  --dry-run
6. #Reboot a running instance.
aws ec2 reboot-instances --instance-ids <INSTANCE_ID> ...
7. #Modify various instance attributes, such as instance type, security groups, and IAM role.
aws ec2 modify-instance-attribute --instance-id <INSTANCE_ID> --<ATTRIBUTE_NAME> <ATTRIBUTE_VALUE>
```

AWS CLI commands for managing tags on EC2 instances

```
1. #Add one or more tags to an AWS resource.
aws ec2 create-tags --resources <RESOURCE_ID> --tags Key=<KEY>,Value=<VALUE> ...
2. #Retrieve tags associated with a specific AWS resource.
aws ec2 describe-tags --filters "Name=resource-id,Values=<RESOURCE_ID>"
3. #Remove specific tags from an AWS resource.
 aws ec2 delete-tags --resources <RESOURCE_ID> --tags Key=<KEY> ...
4. #List Tags on Resources.
aws ec2 describe-tags
5. #Modify Tags on a Resource
aws ec2 create-tags --resources <RESOURCE_ID> --tags Key=<KEY>,Value=<NEW_VALUE> ...
```

Access more here: