Author: Madhurima Rawat

Setting Up and Configuring Cloud Networking

This experiment covers configuring and managing cloud networking services, including setting up Virtual Private Clouds (VPCs) and subnets. It utilizes AWS CLI and LocalStack to simulate cloud environments. This guide includes the required LocalStack version, AWS setup instructions, command execution, and detailed output breakdowns.

1. Creating a Virtual Private Cloud (VPC)

Command:

```
aws ec2 create-vpc --cidr-block 10.0.0.0/16
--endpoint-url=%AWS_ENDPOINT_URL%
```

Explanation:

- aws ec2 create-vpc → Creates a new Virtual Private Cloud (VPC).
- --cidr-block 10.0.0.0/16 \rightarrow Defines the IP address range for the VPC.
- --endpoint-url=%AWS_ENDPOINT_URL% → Uses a custom AWS endpoint (LocalStack or another AWS service).

I	CreateVpc	 I
+		+
П	Vpc	$ \cdot $
+	+	+
CidrBlock	10.0.0.0/16	
DhcpOptionsId	default	
InstanceTenancy	default	
OwnerId	00000000000	
State	pending	
VpcId	vpc-e57c31c086cb3ba0c	
+	+	+
Cid	rBlockAssociationSet	
+		-+
AssociationId	vpc-cidr-assoc-f48b6c421c5fe1c29	
CidrBlock	10.0.0.0/16	
+		-+
	CidrBlockState	

- CidrBlock \rightarrow The assigned IP range (10.0.0.0/16) for the VPC.
- DhcpOptionsId → The default DHCP options for automatic IP allocation.
- InstanceTenancy → default means EC2 instances can be shared on the hardware.
- State → pending, indicating the VPC is being created.
- VpcId → The unique identifier (vpc-e57c31c086cb3ba0c) for the VPC.
- CidrBlockAssociationSet → Shows the association of CIDR blocks with the VPC.
- CidrBlockState → associated, confirming the CIDR block is successfully linked.

2. Creating a Subnet

Command:

```
aws ec2 create-subnet --vpc-id vpc-e57c31c086cb3ba0c
--cidr-block 10.0.1.0/24 --endpoint-url=%AWS_ENDPOINT_URL%
```

Explanation:

- aws ec2 create-subnet → Creates a subnet inside a VPC.
- --vpc-id vpc-e57c31c086cb3ba0c → Associates the subnet with the created VPC.
- --cidr-block 10.0.1.0/24 → Defines a smaller IP range within the VPC.
- --endpoint-url=%AWS_ENDPOINT_URL% → Uses a local AWS environment (e.g., LocalStack).

```
DefaultForAz
                              False
|| Ipv6Native
                            | False
| MapPublicIpOnLaunch
                           False
| OwnerId
                            000000000000
| | State
                            pending
                           arn:aws:ec2:us-east-1:00000000000:subnet/subnet-13df13c5c1
| | SubnetArn
| | SubnetId
                               subnet-13df13c5c1296a641
|| VpcId
                            vpc-e57c31c086cb3ba0c
```

- AssignIpv6AddressOnCreation → False , indicating IPv6 is not enabled.
- AvailabilityZone → The subnet is located in us-east-1a.
- CidrBlock \rightarrow The subnet uses 10.0.1.0/24, which is a smaller range than the VPC.
- MapPublicIpOnLaunch → False, meaning instances launched in this subnet will not automatically get a public IP.
- SubnetId → subnet-13df13c5c1296a641 , the unique identifier for the subnet.
- $VpcId \rightarrow Confirms$ the subnet belongs to Vpc-e57c31c086cb3ba0c.
- State → pending, indicating the subnet is still being created.

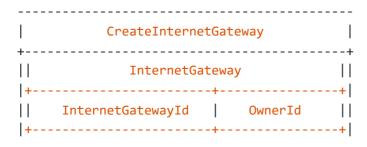
3. Creating an Internet Gateway

Command:

aws ec2 create-internet-gateway --endpoint-url=%AWS ENDPOINT URL%

Explanation:

- aws ec2 create-internet-gateway → Creates an Internet Gateway (IGW) to enable public internet
 access for the VPC.
- --endpoint-url=%AWS_ENDPOINT_URL% → Uses a custom AWS endpoint.



```
|| igw-095e7ce5a8f8472d1 | 000000000000 ||
|+------
```

- InternetGatewayId \rightarrow igw-095e7ce5a8f8472d1 , the unique identifier for the created Internet Gateway.
- OwnerId → The AWS account ID (0000000000000).

4. Attaching the Internet Gateway to the VPC

Command:

```
aws ec2 attach-internet-gateway --internet-gateway-id
igw-095e7ce5a8f8472d1 --vpc-id vpc-e57c31c086cb3ba0c
--endpoint-url=%AWS_ENDPOINT_URL%
```

Explanation:

- aws ec2 attach-internet-gateway → Links the Internet Gateway (IGW) to the specified VPC.
- --internet-gateway-id igw-095e7ce5a8f8472d1 → Specifies the Internet Gateway to attach.
- --vpc-id vpc-e57c31c086cb3ba0c → Identifies the VPC to attach the IGW.
- --endpoint-url=%AWS_ENDPOINT_URL% → Uses a custom AWS endpoint.

Output:

(No explicit output, indicating the command executed successfully.)

Breakdown of Output:

- If no errors occur, the IGW is successfully attached to the VPC.
- The VPC now has potential internet access, but a route table must be updated to enable it.

5. Creating a Route Table

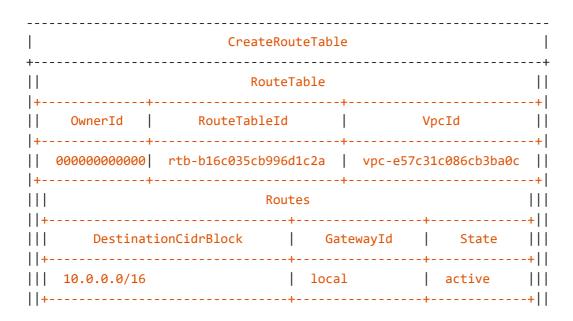
Command:

```
aws ec2 create-route-table --vpc-id
```

Explanation:

- aws ec2 create-route-table \rightarrow Creates a new Route Table for network traffic management.
- --vpc-id vpc-e57c31c086cb3ba0c → Specifies the VPC where the Route Table will be created.
- --endpoint-url=%AWS_ENDPOINT_URL% → Uses a custom AWS endpoint.

Output:



Breakdown of Output:

- RouteTableId \rightarrow The unique ID for the route table (rtb-b16c035cb996d1c2a).
- VpcId → Confirms this route table is associated with vpc-e57c31c086cb3ba0c.
- DestinationCidrBlock → 10.0.0.0/16 shows the VPC's internal network range.
- GatewayId → local means this route table only has local routes initially.
- State → active, confirming the route table is ready.

6. Creating a Route for Internet Access

Command:

```
aws ec2 create-route --route-table-id
rtb-b16c035cb996d1c2a --destination-cidr-block 0.0.0.0/0
```

Explanation:

- aws ec2 create-route → Adds a new route to an existing Route Table.
- --route-table-id rtb-b16c035cb996d1c2a → Specifies the Route Table to modify.
- --destination-cidr-block 0.0.0.0/0 → Routes all external traffic to the IGW (enabling internet access).
- --gateway-id igw-095e7ce5a8f8472d1 → Specifies the IGW as the exit point for internet-bound traffic.
- --endpoint-url=%AWS_ENDPOINT_URL% → Uses a custom AWS endpoint.

Output:

```
| CreateRoute | +----+ | Return | True | +----+
```

Breakdown of Output:

- Return → True, confirming the route was successfully created.
- Now, all internet-bound traffic (0.0.0.0/0) will be routed through the IGW.

7. Associating the Route Table with the Subnet

Command:

```
aws ec2 associate-route-table --route-table-id rtb-b16c035cb996d1c2a --subnet-id subnet-13df13c5c1296a641 --endpoint-url=%AWS_ENDPOINT_URL%
```

Explanation:

- aws ec2 associate-route-table → Links the Route Table to a specific subnet.
- --route-table-id rtb-b16c035cb996d1c2a \rightarrow Specifies the Route Table being associated.
- --subnet-id subnet-13df13c5c1296a641 → Identifies the subnet to attach to the Route Table.
- --endpoint-url=%AWS_ENDPOINT_URL% → Uses a custom AWS endpoint.

 AssociationId → rtbassoc-7cb7be591b1bd9da6 confirms the Route Table is successfully linked to the subnet.

8. Describing VPCs

Command:

aws ec2 describe-vpcs --endpoint-url=%AWS_ENDPOINT_URL%

Explanation:

- aws ec2 describe-vpcs → Lists all VPCs in the environment.
- --endpoint-url=%AWS_ENDPOINT_URL% → Uses a custom AWS endpoint.

1	DescribeVpcs	 I
4	Describevpes	ا ــــــــــــــــــــــــــــــــــــ
H	Vpcs	II.
CidrBlock IsDefault VpcId	172.31.0.0/16 True vpc-71f925ef3211ce7cf	
 Cid	rBlockAssociationSet	Hİ
CidrBlock	vpc-cidr-assoc-3d970697469f6cd5- 172.31.0.0/16	III
iiii	CidrBlockState	Ш
1111 000.00	associated	+
П	Vpcs	11

+	+			
CidrBlock	10.0.0.0/16			
IsDefault	False			
VpcId	vpc-e57c31c086cb3ba0c			
+	+			
CidrBlockAssociationSet				
++				
AssociationId	vpc-cidr-assoc-f48b6c421c5fe1c29			
CidrBlock	10.0.0.0/16			
+	+			
	CidrBlockState			
++				
State	associated			
+	++			

- Two VPCs are listed:
 - \circ vpc-71f925ef3211ce7cf \rightarrow The default VPC (172.31.0.0/16).
 - \circ vpc-e57c31c086cb3ba0c \rightarrow The custom VPC (10.0.0.0/16).
- IsDefault → Indicates whether a VPC is the default one.
- State → associated , confirming that the CIDR blocks are active.

9. Describing Subnets

Command:

aws ec2 describe-subnets --endpoint-url=%AWS_ENDPOINT_URL%

Explanation:

- aws ec2 describe-subnets → Lists all subnets available in the VPCs.
- --endpoint-url=%AWS_ENDPOINT_URL% → Uses a custom AWS endpoint.

	MapPublicIpOnLaunch	True	
	SubnetId	subnet-0e07399449d53791e	
\prod_{i}	VpcId	vpc-71f925ef3211ce7cf	
İI	Subnets		
	CidrBlock	10.0.1.0/24	
	DefaultForAz	False	
	MapPublicIpOnLaunch	False	
	SubnetId	subnet-13df13c5c1296a641	
	VpcId	vpc-e57c31c086cb3ba0c	
14			

- The list contains multiple subnets associated with different VPCs:
 - \circ subnet-0e07399449d53791e \rightarrow A default subnet inside vpc-71f925ef3211ce7cf .
 - subnet-13df13c5c1296a641 → A custom subnet inside vpc-e57c31c086cb3ba0c
- CidrBlock → Defines the range of IPs allocated to the subnet.
- MapPublicIpOnLaunch \rightarrow True means instances in the subnet get a public IP by default.