MODULE 7 · STUDY NOTES

Networking

Building logically isolated networks with Amazon VPC — subnets, route tables, gateways, connectivity options, layered security, and how to troubleshoot them

Networking is where a systems operator earns their keep: almost every connectivity problem in AWS traces back to how a virtual private cloud is wired. This module builds that mental model from the ground up. A cloud network is, at heart, a private IP address space you can deploy resources into — and in AWS that space is an Amazon VPC, a logically isolated virtual network you provision and control. You start with the components inside a VPC (subnets, route tables, the implicit router, gateways, elastic network interfaces) and the addressing rules that govern them (CIDR blocks, reserved addresses, public vs. private subnets). You then work outward through the connectivity options that link a VPC to the internet, to other VPCs, to on-premises data centers, and to AWS services without ever touching the public internet. Finally you layer on defense — route tables, network ACLs, security groups, and bastion hosts — and close with a disciplined, layer-by-layer approach to troubleshooting when traffic will not flow.

LEARNING OBJECTIVES

- Explain the foundational role of an Amazon VPC in AWS Cloud networking
- Identify the networking components inside a VPC and their purpose
- Differentiate the options for VPC connectivity
- Describe the layered network defense model inside a VPC
- List the steps to troubleshoot common VPC network issues
- Configure a VPC

7.1 AWS Cloud networking and the VPC foundation

In its most basic form, a cloud-based network is a **private IP address space** into which you deploy computing resources. In AWS, that space is provided by an **Amazon Virtual Private Cloud (Amazon VPC)** — a virtual network in your own logically isolated area of the AWS Cloud. Inside it you deploy resources such as **Amazon EC2** and **Amazon RDS** instances, and you define how — and whether — the network connects to the internet, to a corporate data center, or to a second VPC. As a systems operator you control these components, so understanding how to configure and relate them is essential.

Formally, an Amazon VPC is a virtual network provisioned in a logically isolated section of the AWS Cloud. It **supports logical separation with subnets**, **offers fine-grained security**, and **supports an optional hardware VPN**. You create a VPC with the Amazon VPC service, and in doing so you define its **IP address range, subnets, and route tables**; you can optionally add network gateways or a hardware VPN to connect it securely to on-premises networks.

KEY TERMS

Subnet
A logical network segment within the VPC, inside a single Availability Zone. Public if attached (through its route table) to an internet gateway, private otherwise. A subnet is required to deploy an instance.

Security group
A set of stateful "firewall" rules that allow or block inbound and outbound traffic for an instance; if none is specified at launch, the instance joins the VPC's default security group.

Primary network interface
The default elastic network interface (a virtual NIC) that connects an instance to the network and cannot be detached from the instance.

Router
The implicit component that routes all traffic within the VPC.

Internet gateway
A VPC component that allows communication between instances in the VPC and the internet.

Virtual private gateway (VGW)
The component defined on the AWS side of a VPN connection, which provides a secure, encrypted tunnel between two endpoints.

Customer gateway
A physical device or software application defined on the client side of a VPN connection.

> CORE VPC CHARACTERISTICS
>
> - A VPC can **span multiple Availability Zones** within an AWS Region.
> - Every VPC has an **implicit router** that routes all traffic in the VPC.
> - Every VPC has a **default route table** whose base rule sends all traffic destined for the VPC's own CIDR range through the **local** route.
> - You may connect it to endpoints in your topology — the internet, another VPC, or your data center — but that connectivity is something you deliberately configure.

7.2 IP addressing: CIDR, ranges, and reserved addresses

When you create a VPC you must specify its allowable IPv4 range as a **CIDR (Classless Inter-Domain Routing) block** — the VPC's **primary CIDR block**. After creation you can optionally add **up to four secondary CIDR blocks**. A VPC's CIDR must be between **`/16` and `/28` inclusive**, so a VPC can contain from **16 to 65,536 IP addresses**. As a best practice, restrict the range to the private addresses defined by **RFC 1918** — draw from `10.0.0.0/8`, `172.16.0.0/12`, or `192.168.0.0/16` — and avoid ranges that overlap other VPCs, since overlap would block you from later connecting them over a hardware VPN. You may also optionally assign an **IPv6 CIDR block** provided by AWS; it always uses a fixed **`/56`** prefix whose range and size you cannot choose. In a multi-CIDR VPC, the DNS server address lives in the **primary** CIDR.

READING CIDR NOTATION

A CIDR block is written `x.x.x.x/n`. The `x.x.x.x` is an IPv4 address — a 32-bit number shown as four bytes (each 0–255) — logically split into a **network prefix** and a **host identifier**. The `/n` gives the length in bits of the network prefix, counted from the left. For IPv4, n can be 0–32, but in a VPC it is restricted to **16–28**. The larger the value of n, the smaller the range and the fewer usable host addresses. Example: `aws ec2 create-vpc --cidr-block 10.0.0.0/16` creates a VPC with 65,536 addresses.

TABLE 7.1 CIDR block examples

CIDR BLOCK	WHAT IT MEANS	ADDRESS RANGE
10.50.1.0/24	24-bit prefix, 8 host bits	10.50.1.0 – 10.50.1.255
10.50.1.0/27	27-bit prefix, 5 host bits	10.50.1.0 – 10.50.1.31
10.50.1.132/32	Full 32-bit prefix — one host	10.50.1.132 (single address)
0.0.0.0/0	No prefix bits fixed	0.0.0.0 – 255.255.255.255 (all addresses)

TABLE 7.2 The first four and last address of every subnet are reserved (example 10.0.0.0/24)

ADDRESS	RATIONALE
10.0.0.0	Network address
10.0.0.1	Reserved by AWS for the VPC router
10.0.0.2	The Amazon DNS server — always the base of the range plus two; also reserved at the base of every subnet
10.0.0.3	Reserved by AWS for future use
10.0.0.255	Network broadcast address (AWS does not support broadcast in a VPC)

7.3 Subnets, route tables, and the implicit router

Subnets further segment a VPC's address range and give you logical groupings — for example, separating EC2 instances from database instances, dividing public from private resources, or splitting resources by team. You can create **up to 200 subnets per VPC**. For IPv4 the minimum subnet size is a **`/28`** (16 addresses); for IPv6 the size is fixed at **`/64`**, and only one IPv6 CIDR block can be allocated to a subnet.

SUBNET CHARACTERISTICS

- A subnet exists in **one — and only one — Availability Zone**.
- A subnet's CIDR block must be a **subset** of its VPC's address range.
- Subnet CIDR blocks within a VPC **must not overlap** one another.
- Traffic to and from every subnet flows through the VPC's **implicit router**.

After you create a subnet you can set whether new EC2 instances launched into it **automatically receive a public IP address**. Only instances launched into a **default VPC** get a public IP by default; instances in a **custom VPC** need the subnet's *Modify auto-assign IP settings* enabled or the setting overridden at launch. Crucially, an auto-assigned **public IP is not static** — unlike an **Elastic IP** (a static IPv4 address reachable from the internet), it is released when the instance is stopped or terminated.

The implicit router uses a **route table** to decide the valid destinations out of a subnet and the target used to reach each. A route table is a set of routes, and each route is a **destination and a target** — read as "any traffic to *destination* should be routed via *target*." A target might be an instance ID, an elastic network interface, an **internet gateway**, or a **virtual private gateway**. When a VPC is created, a **default (main) route table** is created with the local route, and every new subnet is automatically associated with it. To route a subnet's traffic differently, you create a **custom route table** and associate the subnet with it instead.

TABLE 7.3 Reading route-table entries

DESTINATION	TARGET	EFFECT
10.0.0.0/16	local	Traffic within the VPC range stays local — every route table has this route
0.0.0.0/0	igw-id	All other traffic goes to the internet gateway → the subnet is **public**
0.0.0.0/0	nat-gateway-id	All other traffic goes to a NAT gateway → **private** subnet, outbound only
pl-xyz (a service)	vpce-id	Traffic to an AWS service is routed to a gateway endpoint

> **COMMON PITFALL**
>
> Do not equate a public IP address with an Elastic IP. Auto-assigned public IPv4 addresses change every time an instance stops and starts; if a service must always be reachable at the same address, attach an **Elastic IP** instead. And remember a subnet is not "public" because it has public IPs — it is public because its **route table** has a route to an internet gateway.

7.4 Elastic network interfaces

An **elastic network interface (ENI)** is a virtual network interface (NIC) attached to an EC2 instance — the connection point that lets the instance communicate with a network. Each ENI has one primary private IPv4 address plus additional secondary addresses, its own MAC address, and its own security groups. Every instance in a VPC has a default **primary network interface** (assigned a private IPv4 address from the VPC range) that **cannot be detached**. You can create and attach **additional** network interfaces; the number allowed **varies by instance type**. An additional ENI can be detached and moved to another instance, and its attributes — including IP addresses — **follow it**, redirecting traffic to the new instance.

> **WHY ATTACH EXTRA NETWORK INTERFACES**
>
> - **Network and security appliances** — load balancers, NAT servers, and proxy servers often prefer multiple interfaces, each with its own IPs and security groups.
> - **Connecting through AWS PrivateLink** — accessing a service over PrivateLink creates an **interface endpoint** (an ENI) that you attach to the invoking instance.

> **HOW MANY ENIS CAN AN INSTANCE HAVE?**
>
> The maximum number of network interfaces depends on the **instance class**. For example, a `t2.micro` supports a maximum of **two** network interfaces, while an `m5.2xlarge` supports up to **four**. There is no single fixed number — always check the limit for the specific instance type.

7.5 The default VPC and DNS options

When you create an AWS account, AWS automatically provisions a **default VPC** with the CIDR block `172.31.0.0/16` (up to 65,536 addresses), so you can start using VPC features immediately. AWS also creates, inside it, an **internet gateway**, a **default route table** (local route plus a `0.0.0.0/0` route to the internet gateway), and a **public `/20` subnet in each Availability Zone** — each providing up to 4,096 addresses with **auto-assign public IP enabled**. Those subnets are public precisely because they associate with the default route table that routes through the internet gateway.

A **DNS server** resolves a hostname (such as `www.example.com`) to an IP address. When you create a VPC, AWS assigns an Amazon-provided DNS server — since December 2018 called the **Amazon Route 53 Resolver** — which answers DNS queries for names inside the VPC directly and performs recursive lookups against public name servers for everything else. If you want a different DNS setup, the options are to run your **own DNS server** (specified through a special set of **DHCP options**) or to use an **Amazon Route 53 private hosted zone**, a container of records that routes traffic for domains within one or more VPCs without exposing resources to the internet.

A common pattern is **split-horizon DNS**, which pairs a private hosted zone with a public hosted zone so the same hostname resolves differently depending on where the lookup originates: from **inside** the VPC it resolves to an internal IP address, while from **outside** the same name resolves to a different public address. The typical use is maintaining an **internal and external version of the same website** under one domain name.

7.6 VPC connectivity options

By design a VPC is isolated. AWS provides a range of options to connect it to the internet, to other VPCs, to on-premises networks, and to AWS services. The table below maps the common scenario to the recommended service; the sections after it examine each.

TABLE 7.4 VPC connectivity scenarios and solutions

IF YOU NEED TO...	CONSIDER USING	SOLUTION CATEGORY
Connect a private subnet to the internet	NAT gateway or NAT instance	EC2 instance connectivity
Connect a VPC to another VPC	VPC peering	VPC to VPC
Connect a VPC to an external (on-premises) network	Site-to-Site VPN · Direct Connect + VPN · VPN CloudHub	Network to VPC
Reach AWS services without leaving the AWS network	AWS PrivateLink · gateway endpoint	VPC endpoints
Connect to many VPCs and external networks at once	AWS Transit Gateway	Network / VPC to VPC

Network address translation (NAT). When an instance in a private subnet needs outbound internet access — say, to download a software patch — but must stay unreachable from the internet, a **NAT device** is the answer: it forwards the instance's traffic out and returns the responses while **preventing the internet from initiating** connections inward. On the way out the source IP is replaced with the NAT device's address; on the way back it is translated to the instance's private IP. AWS offers a **NAT gateway** and a **NAT instance**, and **NAT gateways are recommended** for better availability and bandwidth and because they are a managed service that needs no administration.

NAT GATEWAY CHARACTERISTICS AND SETUP

- An **AWS-managed** service, created in a specific Availability Zone with **built-in redundancy within that AZ**.
- **Requires an Elastic IP address** and supports **TCP, UDP, and ICMP**.
- To create one: specify the subnet and the Elastic IP allocation, then **update the private subnets' route tables** to point internet-bound traffic to it.
- Best practice: if you use multiple Availability Zones, host a **NAT gateway in each AZ** for better availability.

VPC peering creates a networking connection between two VPCs so they route traffic to each other by **private IP address**. The VPCs can be in the **same Region, different Regions, or different AWS accounts**; the traffic traverses the **AWS backbone**, and **no gateways are required**. The connection is controlled by route tables that reference the peering connection as a target, giving a link with **no single point of failure and no bandwidth bottleneck**. Peering is a two-way agreement that must be **initiated and accepted** by both parties, and because both owners add routes to the peer's range, the two networks **must not have overlapping address spaces**.

PEERING CONNECTION STATES	PEERING LIMITATIONS
the normal creation lifecycle ▪ **Initiating-request** — a request has been initiated ▪ **Pending-acceptance** — awaiting the accepter; expires after **7 days** if ignored ▪ **Provisioning** — accepted, soon to be active ▪ **Active** — traffic can flow (if security groups and route tables allow it)	*restrictions that appear on exams* ▪ **No overlapping** private IP ranges ▪ **No transitive peering** — A↔B and B↔C does not give A↔C ▪ No transitive **edge routing** (through Direct Connect) or **internet gateway** access ▪ **No NAT routing** between VPCs ▪ Cannot resolve a peer's **private IP** via DNS ▪ **No cross-Region** security-group references — use the peer's CIDR instead

Connecting to on-premises networks (VPN). Most companies do not abandon their data centers overnight, so AWS provides several network-to-VPC options. All build on two components you already know: a **virtual private gateway (VGW)** on the AWS side and a **customer gateway** on the remote side.

TABLE 7.5 Network-to-VPC connectivity options

OPTION	WHAT IT IS	KEY POINT
Site-to-Site VPN	An IPsec VPN tunnel over the internet between a VGW and a customer gateway	The VGW has **two tunnels** — configure both for redundancy; use static or dynamic (BGP) routing per your device
Direct Connect + VPN	A dedicated fiber connection (a WAN, **not** the internet) plus IPsec encryption	Private virtual interface → a VPC via the VGW; public virtual interface → public services like Amazon S3; adds privacy and bandwidth
VPN CloudHub	A hub-and-spoke model on a VGW with **multiple** customer gateways	Lets remote sites reach the VPC **and each other**; each site advertises BGP routes

VPC endpoints let you privately connect a VPC to supported AWS services **without an internet gateway, NAT device, VPN, or Direct Connect** — and the traffic never leaves the Amazon network. There are two types, and the difference is a favorite exam point.

INTERFACE ENDPOINT (PRIVATELINK)	GATEWAY ENDPOINT
an ENI with a private IP ▪ Connects to services exposed through **AWS PrivateLink** ▪ Is an **elastic network interface** whose private IP is the entry point for traffic ▪ Supports many services — EC2, Elastic Load Balancing, Systems Manager, Marketplace, and your own services ▪ Onboard a custom service by placing a **Network Load Balancer** in front of it	*a target in your route table* ▪ Directly reaches **Amazon S3 and Amazon DynamoDB** only ▪ Added as a **route-table target** using the service's prefix list (`pl-…`) ▪ Receives a unique ID starting with `vpce-` ▪ Attach a **policy** that allows access to the service; no application code changes needed

AWS Transit Gateway connects Amazon VPCs and on-premises networks to a **single gateway**. Because connecting VPCs in pairs with peering grows an unmanageable mesh as their number climbs into the hundreds, Transit Gateway replaces it with a **hub-and-spoke** model: you manage just one connection from the central gateway to each VPC, data center, or remote office. It routes IPv4 and IPv6 packets between attachments using **transit gateway route tables**, and attached VPCs and VPN connections can **propagate their routes** to those tables. Attachments must be in the **same Region** as the transit gateway, and any newly attached VPC is automatically reachable by every other attached network.

> **COMMON PITFALL**
>
> Scale changes the answer. Peering is point-to-point and non-transitive, so fully connecting N VPCs needs $N \times (N-1)/2$ links; when many VPCs and on-premises networks must interconnect, the intended answer is **AWS Transit Gateway** — one hub, one attachment per network.

7.7 Securing your network

A virtual network can and should be secured at several levels. AWS describes a **layered network defense** with four tiers, from the broadest scope inward to the individual instance.

> **THE FOUR LAYERS OF NETWORK DEFENSE**
>
> - **Route table** — the most broadly scoped level; a private subnet with no direct path to the internet is one of the best protections for internal resources.
> - **Network ACLs** — define the default security behavior for a subnet; typically controlled by the **network security team**.
> - **Security groups** — control behavior at the **instance / elastic network interface** level.
> - **Host-based detection** — optional third-party software on individual EC2 instances that watches for malware, OS vulnerabilities, and other threats.

SECURITY GROUP	NETWORK ACL
instance-level, stateful	*subnet-level, stateless*
▪ Allows traffic to or from an **elastic network interface** ▪ Default: **deny all inbound**, **allow all outbound** ▪ **Stateful** — a response to an allowed request is automatically permitted ▪ You can write **allow rules only**; all rules are evaluated before deciding ▪ Usually configured with the **application development** team	▪ Allows or denies traffic in and out of **subnets** ▪ Default network ACL **allows all** inbound and outbound traffic ▪ **Stateless** — an inbound allow does not imply the outbound response is allowed; define it explicitly ▪ Rules are numbered and evaluated in **ascending order**; the asterisk rule is evaluated last as a catch-all ▪ A network ACL can be associated with **one or more subnets**

The Mom & Pop Café security group illustrates the model: inbound rules allow only **HTTP on port 80**, **SSH on port 22**, and **MySQL/Aurora on port 3306**, while a single outbound rule allows **all traffic**. Because the group is stateful, responses to those inbound requests leave without any extra rule.

Bastion hosts. A **bastion host** (or jump host) sits in a public subnet and provides controlled, public-to-private access — a jump point into a private subnet that keeps the private subnet's attack surface small. Users must hold valid keys for both the bastion and the private instances, and you should **never store the private subnet's keys on the bastion** — an attacker who breaches the bastion would otherwise inherit them. For **Linux**, use the SSH client's **agent forwarding** to pass your key through to private resources; for **Windows**, generate a login password from the private key in the EC2 console and connect with **RDP**.

CONFIGURING BASTION SECURITY GROUPS

- Restrict the bastion's own security group to your **corporate address range** (for example, allow SSH only from `17.5.0.0/16`) so outsiders cannot reach it.
- Give the private instance a rule that allows inbound only **from the bastion's security group** (for example, SSH from `sg-192d536a`) — more flexible and reusable than hardcoding an IP.
- For Windows, a **Remote Desktop Gateway** bastion accepts **HTTPS (TCP 443)** from clients and proxies **RDP (TCP 3389)** to protected instances, applying least privilege at the network layer.

7.8 Troubleshooting networks on AWS

When VPC connectivity fails, work through the layers methodically. The following common checks are a good starting point for most problem determination, and the scenario-specific lists that follow add detail. Keep in mind that a **symptom and its root cause are different things** — the place traffic stops is not always where the misconfiguration lives.

COMMON TROUBLESHOOTING TASKS

- Verify the **instance is up and running** — it has passed both the **System Status** and **Instance Status** checks.
- Verify the **security groups** on the instance allow the required protocols and ports.
- Verify the **network ACLs** on the subnet allow the necessary ports and protocols (in both directions).
- Verify the **route table** on the subnet has destination rules pointing to the appropriate targets.

CANNOT CONNECT TO AN INSTANCE OVER THE INTERNET

- Confirm the **public IP or DNS name** you are using is correct.
- Confirm the instance has a **public IP or Elastic IP** (recall a public IP changes on stop/start).
- Confirm an **internet gateway is attached** to the instance's VPC.
- Confirm the subnet's route table has a route for **`0.0.0.0/0` via the internet gateway**.

CANNOT CONNECT USING SSH, NAT, OR PEERING

- **SSH** — verify the **user name** (not every AMI defines `ec2-user`) and the correct **private key** with correct permissions; use `ping` to test reachability.
- **NAT** — verify the route table points to the NAT device; for a **NAT instance**, check that **Source/Dest Check is disabled**, restart it, and review its inbound security group rules.
- **Peering** — verify the peering request was **approved**, that security group rules use the peer's **CIDR range or security-group ID**, and that **network ACLs** are not forbidding the traffic.

For deeper analysis, **VPC Flow Logs** capture information about IP traffic going to and from network interfaces. In the module's troubleshooting activity, you enable VPC Flow Logs, store them in an **Amazon S3 bucket**, then download, extract, and analyze the entries — where actions taken during troubleshooting are reflected — to confirm whether traffic is actually reaching its destination.

Module summary

- An Amazon VPC is a logically isolated virtual network you provision in the AWS Cloud; it spans every Availability Zone in its Region, has an implicit router, and is defined by an IPv4 CIDR block from /16 to /28 (16–65,536 addresses), with optional secondary CIDRs and an AWS-assigned IPv6 /56.
- AWS reserves five addresses in every subnet (the first four and the last); a subnet lives in exactly one Availability Zone, must be a subset of the VPC range, and cannot overlap other subnets.
- A subnet is public or private based on its route table — a route sending 0.0.0.0/0 to an internet gateway makes it public; an auto-assigned public IP is not static (only an Elastic IP is).
- Security groups are stateful, operate at the instance/ENI level, and hold allow rules only; network ACLs are stateless, operate at the subnet level, and hold both allow and deny rules evaluated by number.
- Connectivity maps to needs: NAT gateway (private subnet → internet, outbound only); VPC peering (VPC ↔ VPC by private IP, non-transitive, no overlapping ranges); Site-to-Site VPN / Direct Connect / VPN CloudHub (on-premises); Transit Gateway (a hub for many networks); and VPC endpoints — interface (PrivateLink ENI, many services) or gateway (Amazon S3 and DynamoDB only).
- Elastic network interfaces are virtual NICs; the primary ENI cannot be detached, additional ENIs vary by instance type, and their attributes follow them when moved.
- Network defense is layered — route tables, network ACLs, security groups, and host-based detection — and a bastion host provides controlled public-to-private access without storing private keys on the bastion.
- Troubleshoot connectivity layer by layer: instance status checks, security groups, network ACLs (both directions), and route-table targets; use VPC Flow Logs stored in Amazon S3 for deeper analysis, and remember symptom ≠ root cause.

Review questions

1. What single setting makes a subnet public rather than private, and why is a public IP different from an Elastic IP?

Answer: A subnet is public when its route table has a route sending 0.0.0.0/0 to an internet gateway. A public IP is auto-assigned but not static — it is released when the instance stops or terminates — whereas an Elastic IP is a static IPv4 address that stays attached.

2. State the VPC CIDR size limits and how many addresses AWS reserves in each subnet.

Answer: A VPC CIDR must be between /16 and /28 (16 to 65,536 addresses), with up to four secondary CIDRs and an optional AWS-assigned IPv6 /56. AWS reserves five addresses in every subnet — the first four and the last.

3. Compare security groups and network ACLs on state, level, and rule types.

Answer: Security groups are stateful, operate at the instance/ENI level, and allow only (default deny inbound, allow outbound). Network ACLs are stateless, operate at the subnet level, and support both allow and deny rules evaluated in ascending rule-number order (default allows all).

4. Match each need to a connectivity option: private subnet to the internet; VPC to VPC; VPC to a data center; VPC to AWS services privately; many VPCs through one hub.

Answer: Private subnet → internet: NAT gateway. VPC ↔ VPC: VPC peering. VPC → data center: Site-to-Site VPN, Direct Connect + VPN, or VPN CloudHub. VPC → AWS services privately: VPC endpoints (interface/PrivateLink or gateway). Many VPCs through one hub: AWS Transit Gateway.

5. Why is VPC peering non-transitive, and what does that mean for three VPCs A, B, and C?

Answer: Peering is a direct one-to-one connection controlled by route tables, and routes cannot chain through a peer. If A peers with B and B peers with C, A still cannot reach C — nor use B's internet gateway or NAT. To connect A and C you must create a separate peering connection between them.

6. Name the two components of a Site-to-Site VPN, and the two kinds of VPC endpoint and what each reaches.

Answer: A Site-to-Site VPN uses a virtual private gateway (AWS side) and a customer gateway (on-premises side). VPC endpoints: an interface endpoint (AWS PrivateLink — an ENI with a private IP) reaches many services; a gateway endpoint reaches only Amazon S3 and Amazon DynamoDB via a route-table target.

7. You cannot connect to an EC2 instance over the internet. What do you check, and in what order?

Answer: Confirm the instance is running and has passed its status checks; verify security groups allow the port/protocol; verify network ACLs allow it both ways; confirm the instance has a public or Elastic IP and the correct address is used; confirm an internet gateway is attached and the route table has 0.0.0.0/0 pointed to it. VPC Flow Logs in Amazon S3 help confirm whether traffic is arriving.