**4**

**Networking in AWS**

Enterprises today have become exponentially more agile by leveraging the power of the cloud. In this chapter, we will highlight the scale of AWS Global Infrastructure and teach you about AWS networking foundations.

Networking is the first step for any organization to set up its landing zone and the entire IT workload built on top of it. You could say that networking is the backbone of the IT application and infrastructure workload. AWS provides various networking services for building your IT landscape in the cloud and in this chapter, you will dive deep into AWS networking services.

Every business is now running at a global scale and organizations need to target global populations with their product. With a traditional on-premise IT workload, it becomes challenging to scale globally and provide the same user experience across the globe. AWS helps solve these problems through edge networking, and you will learn more about deploying your application for global users without compromising their experience. Furthermore, you will learn about network security and building a hybrid cloud.

In this chapter, you will learn about the following topics:

* Learning about the AWS Global Infrastructure
* AWS networking foundations
* Edge networking
* Building hybrid cloud connectivity in AWS
* AWS cloud network security

Without further ado, let’s get down to business.

**Learning about the AWS Global Infrastructure**

The infrastructure offered by AWS is highly secure and reliable. It offers over 200 services. Most are available in all AWS Regions worldwide, spread across 245 countries. Regardless of the type of technology application you are planning to build and deploy, AWS is sure to provide a service that will facilitate its deployment.

AWS has millions of customers and thousands of consulting and technology partners worldwide. Businesses large and small across all industries rely on AWS to handle their workloads. Here are some statistics to give you an idea of the breadth of AWS’s scale. AWS provides the following as its global infrastructure:

* 26 launched Regions and 8 announced Regions
* 84 Availability Zones
* Over 110 Direct Connect locations
* Over 310 Points of Presence
* 17 Local Zones and 32 announced LZs
* 24 Wavelength Zones

**IMPORTANT NOTE**

These numbers are accurate at the time of writing this book. By the time you read this, it would not be surprising for the numbers to have changed.

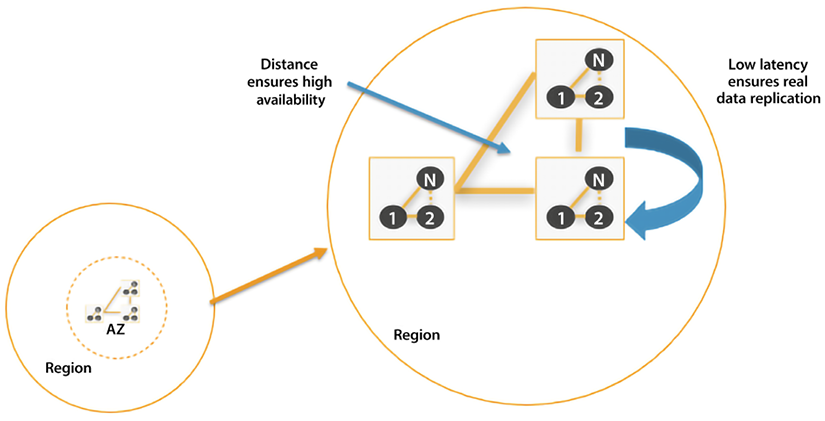
Now that we have covered how the AWS infrastructure is organized at a high level, let’s learn about the elements of the AWS Global Infrastructure in detail.

**Regions, Availability Zones, and Local zones**

How can Amazon provide such a reliable service across the globe? How can they offer reliability and durability guarantees for some of their services? The answer reveals why they are the cloud leaders and why it’s difficult to replicate what they offer. AWS has billions of dollars worth of infrastructure deployed across the world. These locations are organized into different Regions and Zones. More formally, AWS calls them the following:

* AWS Regions
* **Availability Zones** (**AZs**)
* **Local Zones** (**LZs**)

As shown in the following diagram, an AZ is comprised of multiple distinct data centers, each equipped with redundant power, networking, and connectivity, and located in separate facilities.



*Figure 4.1: AWS Regions and AZs*

**AWS Regions** exist in separate geographic areas. Each AWS Region comprises several independent and isolated data centers (AZs) that provide a full array of AWS services.

AWS is continuously enhancing its data centers to provide the latest technology. AWS’s data centers have a high degree of redundancy. AWS uses highly reliable hardware, but the hardware is not foolproof. Occasionally, a failure can happen that interferes with the availability of resources in each data center. Suppose all instances were hosted in only one data center. If a failure occurred with the whole data center, then none of your resources would be available. AWS mitigates this issue by having multiple data centers in each Region.

By default, users will always be assigned a default Region. You can obtain more information about how to change and maintain AWS environment variables using the AWS CLI by visiting this link: <https://docs.aws.amazon.com/cli/latest/userguide/cli-configure-envvars.html>.

In the following subsection, we will look at AWS Regions in greater detail and see why they are essential.

**AWS Regions**

**AWS Regions** are groups of data centers in one geographic location that are specially designed to be independent and isolated from each other. A single Region consists of a collection of data centers spread within that Region’s geographic boundary. This independence promotes availability and enhances fault tolerance and stability. While working on the console, you will see AWS services available in that Region. There is a possibility that a particular service is not available in your Region. Eventually, all services become **generally available** (**GA**) after their launch; however, the timing of availability may differ between different Regions. Some services are global – for example, **Direct Connect Gateway** (**DXGW**), **Identity and Access Management** (**IAM**), CloudFront, and Route53.

Other services are not global but allow you to create inter-Region fault tolerance and availability. For example, Amazon **Relational Database Service** (**RDS**) allows you to create read replicas in multiple Regions. To find out more about this, visit <https://aws.amazon.com/blogs/aws/cross-region-read-replicas-for-amazon-rds-for-mysql/>.

One of the advantages of using such an architecture is that resources will be closer to users, increasing access speed and reducing latency.

Another obvious advantage is that you can serve your clients without disruption even if a whole Region becomes unavailable by planning your disaster recovery workload to be in another Region. You will be able to recover faster if something goes wrong, as these read replicas can be automatically converted to the primary database if the need arises.

As of May 2022, there are 26 AWS Regions and 8 announced Regions. The naming convention that is usually followed is to list the country code, followed by the geographic region, and the number. For example, the US East Region in Ohio is named as follows:

* Location: US East (Ohio)
* Name: us-east-2

AWS has a global cloud infrastructure, so you can likely find a Region near your user base, with a few exceptions, such as Russia. If you live in Greenland, it may be a little further away from you – however, you will still be able to connect as long as you have an internet connection.

In addition, AWS has dedicated Regions specifically and exclusively for the US government called AWS GovCloud. This allows US government agencies and customers to run highly sensitive applications in this environment. AWS GovCloud offers the same services as other Regions, but it complies explicitly with requirements and regulations specific to the needs of the US government.

The full list of available Regions can be found here: <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-regions-availability-zones.html#concepts-available-regions>.

As AWS continues to grow, do not be surprised if it offers similar Regions to other governments worldwide, depending on their importance and the demand they can generate.

**AWS AZs**

As we discussed earlier, AZs are components of AWS Regions. The clusters of data centers within a Region are called AZs. A single AZ consists of multiple data centers. These data centers are connected to each other using AWS-owned dedicated fiber optic cables and located within a 60-mile radius, which is far enough to avoid localized failures, yet achieves a faster data transfer between the data centers. AZs have multiple power sources, redundant connectivity, and redundant resources. All this translates into unparalleled customer service, allowing them to deliver highly available, fault-tolerant, and scalable applications.

The AZs within an AWS Region are interconnected. These connections have the following properties:

* Fully redundant
* High-bandwidth
* Low-latency
* Scalable
* Encrypted
* Dedicated

Depending on the service you are using, if you decide to perform a multi-AZ deployment, an AZ will automatically be assigned to the service, but for some services, you may be able to designate which AZ is to be used.

Every AZ forms a completely segregated section of the AWS Global Infrastructure, physically detached from other AZs by a substantial distance, often spanning several miles. Each AZ operates on a dedicated power infrastructure, providing customers with the ability to run production applications and databases that are more resilient, fault-tolerant, and scalable compared to relying on a single data center.

High-bandwidth, low-latency networking interconnects all the AZs – but what about fulfilling the need for low-latency bandwidth within a highly populated city? For that, AWS launched LZs. Let’s learn more about them.

**AWS LZs**

While AZs focus on covering larger areas throughout regions, such as US-West and US-East, AWS fulfills the needs of highly populated cities through LZs. AWS LZs are newer components in the AWS infrastructure family. LZs place select services close to end users, allowing them to create AWS applications that deliver single-digit, millisecond responses. An LZ is the compute and storage infrastructure located close to high-population areas and industrial centers, and offers high-bandwidth, low-latency connectivity to the broader AWS infrastructure. Due to their proximity to the customer, LZs facilitate the delivery of applications that necessitate latency in single-digit milliseconds to end-users. As of March 2023, AWS had has 32 LZs.

AWS LZs can run various AWS services, such as Amazon Elastic Compute Cloud, Amazon Virtual Private Cloud, Amazon Elastic Block Store, Amazon Elastic Load Balancing, Amazon FSx, Amazon EMR, Amazon ElastiCache, and Amazon RDS in geographic proximity to your end users.

The naming convention for LZs is to use the AWS Region followed by a location identifier, for example, **us-west-2-lax-2a**. Please refer to this link for the latest supported services in LZs – <https://aws.amazon.com/about-aws/global-infrastructure/localzones/features/?nc=sn&loc=2>.

Now you have learned about the different components of the AWS Global Infrastructure, let’s look at the benefits of using AWS infrastructure.

**Benefits of the AWS Global Infrastructure**

The following are the key benefits of using AWS’s cloud infrastructure:

* **Security** – One of the most complex and risky tasks is to maintain security, especially when it comes to the data center’s physical security. With AWS’s shared security responsibility model, you offload infrastructure security to AWS and focus on the application security that matters for your business.
* **Availability** – One of the most important factors for the user’s experience is to make sure your application is highly available, which means you need to have your workload deployed in a physically separated geographic location to reduce the impact of natural disasters. AWS Regions are fully isolated, and within each Region, the AZs are further isolated partitions of AWS infrastructure. You can use AWS infrastructure with an on-demand model to deploy your applications across multiple AZs in the same Region or any Region globally.
* **Performance** – Performance is another critical factor in retaining and increasing the user base. AWS provides low-latency network infrastructure by using redundant 100 GbE fiber, which leads to terabits of capacity between regions. Also, you can use AWS Edge AZs for applications that require low millisecond latency, such as 5G, gaming, AR/VR, and IoT.
* **Scalability** – When user demands increase, you must have the required capacity to scale your application. With AWS, you can quickly spin up resources, deploying thousands of servers in minutes to handle any user demand. You can also scale down when demand goes down and don’t need to pay for any overprovisioned resources.
* **Flexibility** – With AWS, you can choose how and where to run your workloads; for example, you can run applications globally by deploying into any of the AWS Regions and AZs worldwide. You can run your applications with single-digit millisecond latencies by choosing AWS LZs or AWS Wavelength. You can choose AWS Outposts to run applications on-premises.

Now that you have learned about the AWS Global Infrastructure and its benefits, the next question that comes to mind is how am I going to use this infrastructure? Don’t worry – AWS has you covered by providing network services that allow you to create your own secure logical data center in the cloud and completely control your IT workload and applications. Furthermore, these network services help you to establish connectivity to your users, employees, on-premises data centers, and content distributions. Let’s learn more about AWS’s networking services and how they can help you to build your cloud data center.

**AWS networking foundations**

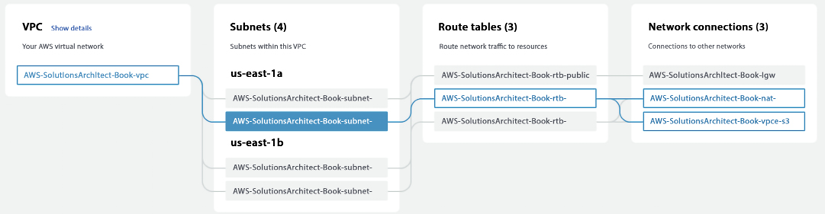
When you set up your IT infrastructure, what comes to mind first? I have the servers now – how can I connect them to the internet and each other so that they can communicate? This connectivity is achieved by networking, without which you cannot do anything.

Networking concepts are the same when it comes to the cloud. In this chapter, you will not learn what networking is, but instead how to set up your private network in the AWS cloud and establish connectivity between the different servers in the cloud and from on-premises to an AWS cloud. First, let’s start with the foundation; the first step to building your networking backbone in AWS is using Amazon VPC.

**Amazon Virtual Private Cloud (VPC)**

VPC is one of the core services AWS provides. Simply speaking, a VPC is your version of the AWS cloud, and as the name suggests, it is “private,” which means that by default, your VPC is a logically isolated and private network inside AWS. You can imagine a VPC as being the same as your own logical data center in a virtual setting inside the AWS cloud, where you have complete control over the resources inside your VPC. AWS resources like AWS servers, and Amazon EC2 and Amazon RDS instances are placed inside the VPC, including all the required networking components to control the data traffic as per your needs.

Creating a VPC could be a very complex task, but AWS has made it easy by providing **Launch VPC Wizard**. You can visualize your network configuration when creating the VPC. The following screenshot shows the VPC network configuration across two AZs, **us-east-1a** and **us-east-1b**:



*Figure 4.2: AWS VPC configuration with a private subnet flow*

In the preceding diagram, you can see VPCs spread across two AZs, where each AZ has two subnets – one public and one private. The highlighted flow shows the data flow of a server deployed into a private subnet of the **us-east-1** AZ. Before going into further details, let’s look at key VPC concepts to understand them better:

* **Classless Inter-Domain Routing (CIDR) blocks**: CIDR is the IP address range allocated to your VPC. When you create a VPC, you specify its set of IP addresses with CIDR notation. CIDR notation is a simplified way of showing a specific range of IP addresses. For example, 10.0.0.0/16 covers all IPs from 10.0.0.0 to 10.0.255.255, providing 65,535 IP addresses to use. All resources in your VPC must fall within the CIDR range.
* **Subnets**: As the name suggests, the subnet is the VPC CIDR block subset. Partitions of the network are divided by the CIDR range within the range of IP addresses in your VPC. A VPC can have multiple subnets for different kinds of services or functions, like a frontend subnet (for internet access to a web page), a backend subnet (for business logic processing), and a database subnet (for database services).

Subnets create trusted boundaries between private and public resources. You should organize your subnets based on internet accessibility. A subnet allows you to define clear isolation between public and private resources. The majority of resources on AWS can be hosted in private subnets. You should use public subnets under controlled access and use them only when it is necessary. As you will keep most of your resources under restricted access, you should plan your subnets so that your private subnets have substantially more IPs available than your public subnets.

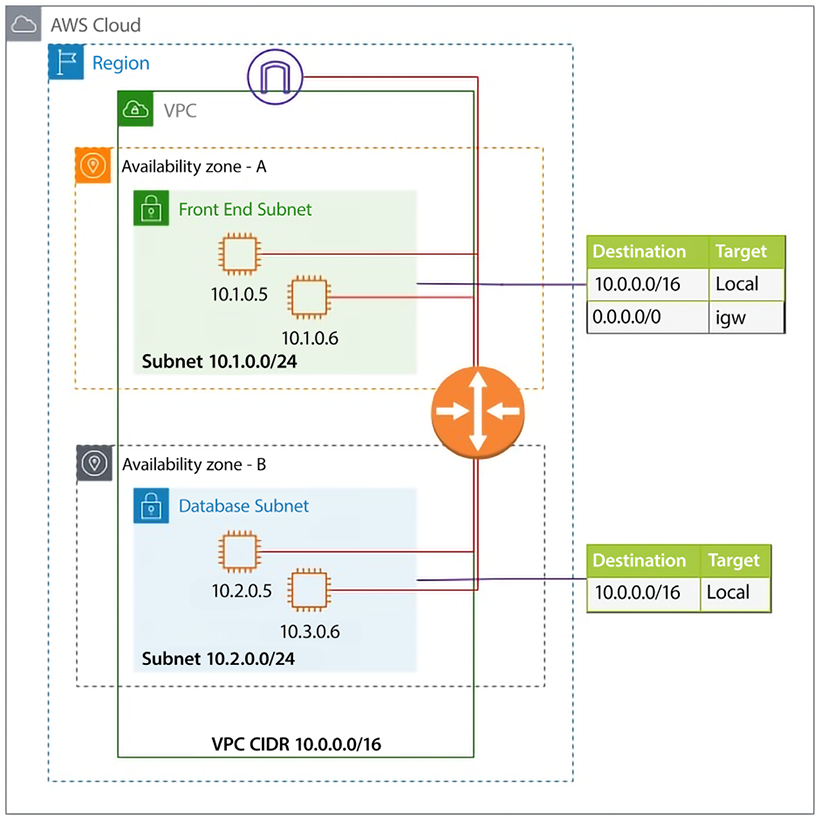
* **Route tables**: A routing table contains a set of rules called routes. Routes determine where the traffic will flow. By default, every subnet has a routing table. You can manually create a new route table and assign subnets to it. For better security, use the custom route table for each subnet.
* An **Internet Gateway (IGW)**: The IGW sits at the edge of the VPC and provides connectivity between your VPC resources and the public network (the internet). By default, internet accessibility is denied for internet traffic in your environment. An IGW needs to be attached to your public subnet through the subnet’s route table, defining the rules to the IGW. All of your resources that require direct access to the internet (public-facing load balancers, NAT instances, bastion hosts, and so on) would go into the public subnet.
* **Network Address Translation (NAT) gateways**: A NAT gateway provides outbound internet access to the private subnet and prevents connections from being initiated from outside to your VPC resources. A private subnet blocks all incoming and outgoing internet traffic, but servers may need outgoing internet traffic for software and security patch installation. A NAT gateway enables instances in a private subnet to initiate outbound traffic to the internet and protects resources from incoming internet traffic. All restricted servers (such as database and application resources) should deploy inside your private subnet.
* **Security Groups (SGs)**: SGs are the virtual firewalls for your instances to control inbound and outbound packets. You can only use allow statements in the SG, and everything else is denied implicitly. SGs control inbound and outbound traffic as designated resources for one or more instances from the CIDR block range or another SG. As per the principle of least privilege, deny all incoming traffic by default and create rules that can filter traffic based on TCP, UDP, and **Internet Control Message Protocol** (**ICMP**).
* **Network Access Control List (NACL)**: A NACL is another firewall that sits at the subnet boundary and allows or denies incoming and outgoing packets. The main difference between a NACL and an SG is that the NACL is stateless – therefore, you need to have rules for incoming and outgoing traffic. With an SG, you need to allow traffic in one direction, and return traffic is, by default, allowed.

You should use an SG in most places as it is a firewall at the EC2 instance level, while a NACL is a firewall at the subnet level. You should use a NACL where you want to put control at the VPC level and also deny specific IPs, as an SG cannot have a deny rule for network traffic coming for a particular IP or IP range.

* **Egress-only IGWs**: These provide outbound communication from **Internet Protocol version 6** (**IPv6**) instances in your VPC to the internet and prevent the inbound connection from the internet to your instances on IPv6. IPv6, the sixth iteration of the Internet Protocol, succeeds IPv4 and employs a 128-bit IP address. Like IPv4, it facilitates the provision of unique IP addresses required for internet-connected devices to communicate.
* **DHCP option sets**: This is a group of network information, such as DNS name server and domain name used by EC2 instances when they launch.
* **VPC Flow Logs**: These enable you to monitor traffic flow to your system VPC, such as accepted and rejected traffic information for the designated resource to understand traffic patterns. Flow Logs can also be used as a security tool for monitoring traffic reaching your instance. You can create alarms to notify you if certain types of traffic are detected. You can also create metrics to help you identify trends and patterns.

To access servers in a private subnet, you can create a bastion host, which acts like a jump server. It needs to be hardened with tighter security so that only appropriate people can access it. To log in to the server, always use public-key cryptography for authentication rather than a regular user ID and password method.

A VPC resides only within an AWS Region where it can span across one or more AZs within the Region. In the following diagram, two AZs are being utilized within a Region. Furthermore, you can create one or more subnets inside each AZ, and resources like EC2 and RDS are placed inside the VPC in specific subnets. This architecture diagram shows a VPC configuration with a private and public subnet:



*Figure 4.3: AWS VPC network architecture*

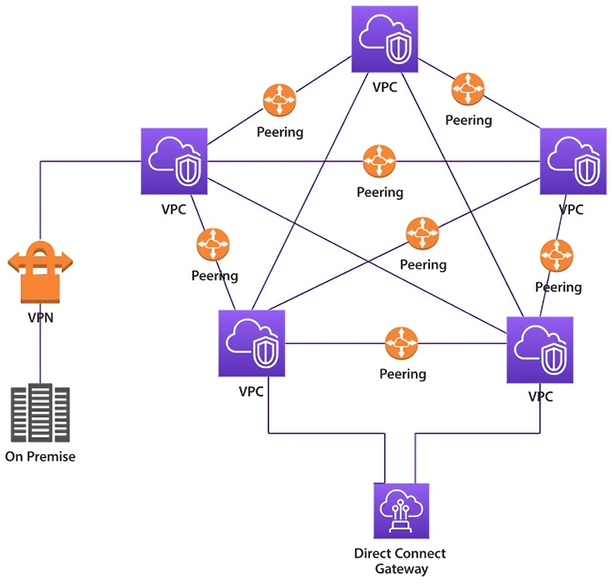
As shown in this diagram, VPC subnets can be either private or public. As the name suggests, a private subnet doesn’t have access to and from the internet, and a public subnet does. By default, any subnet you create is private; what makes it public is the default route – as in, 0.0.0.0/0, via the IGW.

The VPC’s route tables comprise directives for packet routing, and a default route table exists. However, unique route tables can be assigned to individual subnets. By default, all VPC subnets possess interconnectivity. This default behavior can be modified with VPC enhancements for more precise subnet routing, which enables configuration of subnet route tables that direct traffic between two subnets in a VPC through virtual appliances like intrusion detection systems, network firewalls, and protection systems.

As you can see, AWS provides multiple layers for network configuration and security at each layer that can help to build and protect your infrastructure. If attackers can access one component, they must restrict to limited resources by keeping them in their isolated subnet. Due to the ease of VPC creation and building tighter security, organizations tend to create multiple VPCs, which makes things more complicated when these VPCs need to communicate with each other. To simplify this, AWS provides **Transit Gateway** (**TGW**). Let’s learn more about it.

**AWS TGW**

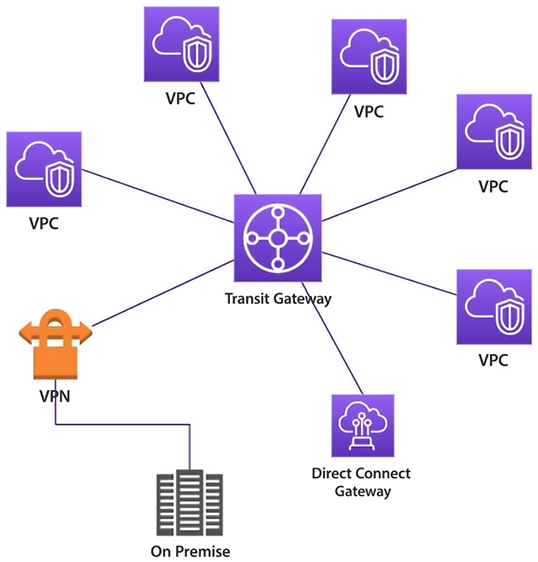
As customers are spinning more and more VPCs in AWS, there is an ever-increasing need to connect various VPCs. Before TGW, you could connect VPCs using VPC peering, but VPC peering is a one-to-one connection, which means that resources within peered VPCs only can communicate with each other. If multiple VPCs need to communicate with each other, which is often the case, it results in a complex mesh of VPC peering. For example, a shown in the diagram below, if you have 5 VPCs, you need 10 peering connections.



*Figure 4.4: VPC connectivity using VPC peering without TGW*

As you can see in the preceding diagram, managing so many VPC peering connections will become challenging, and there is also a limit on the number of peering connections per account. To overcome this challenge, AWS released TGW. TGW needs one connection called an attachment to a VPC, and you can establish full- or part-mesh connectivity easily without maintaining so many peering connections.

The following diagram shows simplified communication between five VPCs using TGW.



*Figure 4.5: VPC connectivity with TGW*

AWS TGW is a central aggregation service spanned within a Region, which can be used to connect your VPCs and on-premises networks. TGW is a managed service that takes care of your availability and scalability and eliminates complex VPN or peering connection scenarios when connecting with multiple VPCs and on-premises infrastructure. You can connect TGWs in different Regions by using TGW peering.

TGW is a Regional entity, meaning you can only attach VPCs to the TGW within the same Region, and per VPC, the bandwidth reserved is 50 Gbps. However, one TGW can have up to 5,000 VPC attachments.

In this section, you have learned how to establish network communication between VPCs, but what about securely connecting to resources such as Amazon S3 that live outside of a VPC or other AWS accounts? AWS provides PrivateLink to establish a private connection between VPCs and other AWS services. Let’s learn more about AWS PrivateLink.

**AWS PrivateLink**

AWS PrivateLink establishes secure connectivity between VPCs and AWS services, preventing exposure of traffic to the internet. PrivateLink allows for the private connection of a VPC with supported AWS services hosted by different AWS accounts.

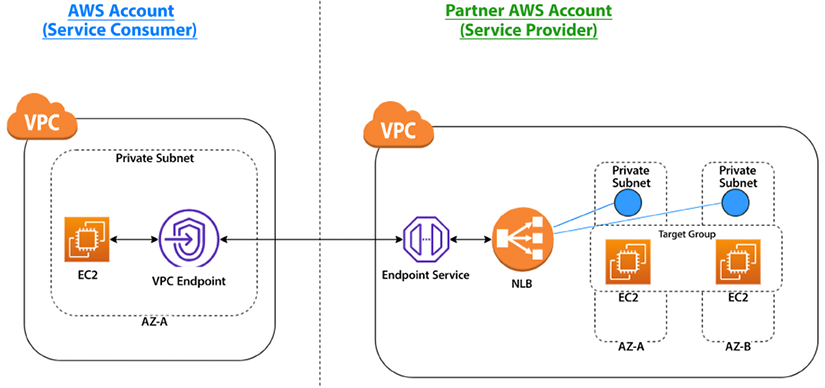
You can access AWS services from a VPC using the Gateway VPC endpoint and Interface VPC endpoint. Gateway endpoints do not support PrivateLink but allow for connection to Amazon S3 and DynamoDB without the need for an IGW or NAT device in your VPC. For other AWS services, an interface VPC endpoint can be created to establish a connection to services through AWS PrivateLink.

Enabling PrivateLink in AWS requires the creation of an endpoint network interface within the desired subnet, and assignment of a private IP address from the subnet address range for each specified subnet in the VPC. You can view the endpoint network interface in your AWS account, but you can’t manage it yourself.

PrivateLink essentially provides access to the resources hosted in other VPC or other AWS accounts within the same subnet as the requester. This eliminates the need to use any NAT gateway, IGW, public IP address, or VPN. Therefore, it provides better control over your services, which are reachable via a client VPC.

As shown in the following diagram, AWS PrivateLink enables private connectivity between the **Service Provider** and **Service Consumer** using AWS infrastructure to exchange data without going over the public internet. To achieve this, the **Service Provider** creates an **Endpoint Service** in a private subnet.

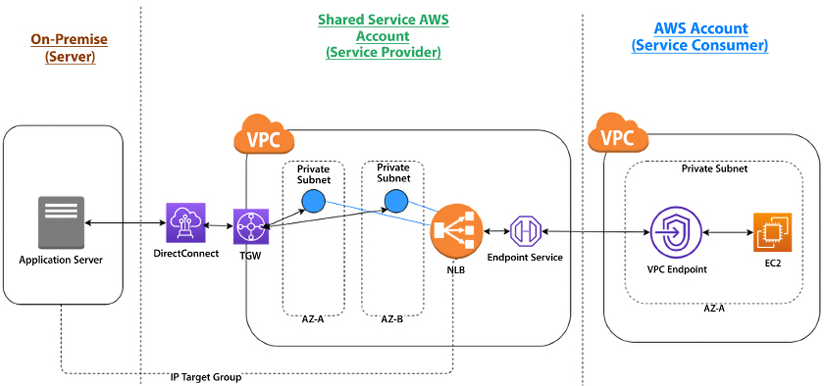
In contrast, the **Service Consumer** creates an endpoint in a private subnet with the **Service Provider**’s service API as the target.



*Figure 4.6: PrivateLink between partner Service Provider and Service Consumer accounts*

As shown in the preceding architecture diagram, the partner sets up an **Endpoint Service** to expose the service running behind the load balancer (**NLB**). An **NLB** is created in each **Private Subnet**. These services are running on **EC2** instances hosted inside a **Private Subnet**. The client can then create a **VPC Endpoint** with the target as the **Endpoint Service** and use it to consume the service.

Let’s look at another pattern shown in the following diagram, which depicts the use of PrivateLink between a **Service Consumer** on an AWS account and an on-premises **Service Provider**.



*Figure 4.7: PrivateLink between a shared Service Provider, on-premise server, and a Service Consumer account*

In this setup, the on-premise servers are the service providers. The **NLB** in the **Shared Service** account is configured with an auto-scaling group with targets referencing the IP addresses of the on-premise servers. The **NLB** is then exposed as an **Endpoint Service**. The **Service Consumer** account can consume this **Endpoint Service** by creating a **VPC Endpoint**. Here, **DirectConnect** provides a dedicated high-speed fiber optics line between the on-premises server and AWS Regions. You will learn more about DirectConnect in this chapter in the *Building Hybrid Cloud Connectivity in AWS* section.

Now, many applications run globally and target to harness users in every corner of the world to accelerate their business. In such a situation, it becomes essential that your users have the same experience while accessing your application regardless of their physical location. AWS provides various edge networking services to handle global traffic. Let’s learn more about this.

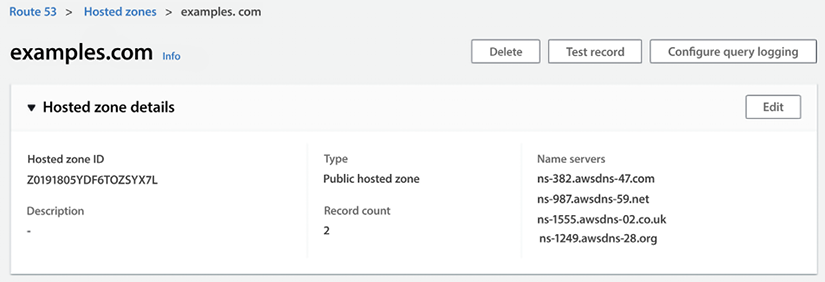
**Edge networking**

Edge networking is like last-mile delivery in the supply chain world. When you have users across the world, from the USA to Australia and India to Brazil, you want each user to have the same experience regardless of the physical location of your server where the application is hosted. There are serval components that play their role in building last mile networking. Let’s explore them in detail.

**Route 53**

Amazon Route 53 is a fully managed, simple, fast, secure, highly available, and scalable DNS service. It provides a reliable and cost-effective means for systems and users to translate names like www.example.com into IP addresses like 1.2.3.4. Route 53 is a domain register where you can register a new domain. You can choose an available domain and add it to the cart from the AWS Console and define contacts for the domain. AWS allows you to transfer your domains to AWS and between accounts.

In Route 53, AWS assigns four name servers for all domains, as shown in the screenshot: one for **.com**, one for **.net**, one for **.co.uk**, and one for **.org**. Why? For higher availability! If there is an issue with the **.net** DNS services, the other three continue to provide high availability for your domains.



*Figure 4.8: Route 53 name server configuration*

Route 53 supports both public and private hosted zones. Public hosted zones have a route to internet-facing resources and resolve from the internet using global routing policies. Meanwhile, private hosted zones have a route to VPC resources and resolve from inside the VPC. It helps to integrate with on-premises private zones using forwarding rules and endpoints.

Route 53 provides IPV6 support with end-to-end DNS resolution and support for IPv6 forward (AAAA) and reverse (PTR) DNS records, along with health check monitoring for IPv6 endpoints. For PrivateLink support, when configuring, you can specify a private DNS name, and the Route 53 resolver will resolve it to the PrivateLink endpoint.

Route 53 provides the following seven types of routing policies for traffic:

* **Simple routing policy** – This is used for a single resource (for example, a web server created for the www.example.com website).
* **Failover routing policy** – This is used to configure active-passive failover.
* **Geolocation routing policy** – This routes traffic based on the user’s location.
* **Geoproximity routing policy** – This is used for geolocation when users are shifting from one location to another.
* **Latency routing policy** – This optimizes the best latency for the resources deployed in multiple AWS Regions.
* **Multivalue answer routing policy** – This is used to respond to DNS queries with up to eight healthy, randomly selected records.
* **Weighted routing policy** – This is used to route traffic to multiple resource properties as defined by you (for example, you want to say 80% traffic to site A and 20% to site B).

You can build advanced routing policies by nesting these primary routing policies into traffic policies; the following diagram shows a nested policy architecture.



*Figure 4.9: Route 53 nested routing policy*

In the preceding diagram, you can see the policy is **Geolocation**-based, which routes traffic based on the user’s location and its proximity to the nearest Region. In the second level, you have a nested policy defined as a **Weighted** policy within the region that routes traffic to servers based on the weight you have defined to route traffic to individual application servers. Advanced routing policies can be built by nesting the seven primary routing policies into traffic policies. You can find more details on this routing policy in the AWS user document here – <https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/routing-policy.html>.

Route 53 Resolver rules tell Route 53 to query a domain. For DNS zones that should resolve on-premises, add a forward rule and point toward the appropriate outbound resolver. Public hosted zones route traffic to internet-facing resources and resolve from the internet using global routing policies. Private hosted zones route traffic to VPC resources and resolve from inside the VPC. Private hosted zones integrate with on-premises private zones using forwarding rules and endpoints.

Route 53 is the only service in which AWS offers 100% SLA, which means AWS makes its best effort to ensure it is 100% available. If Route 53 does not meet the availability commitment, you will be eligible to receive a Service Credit.

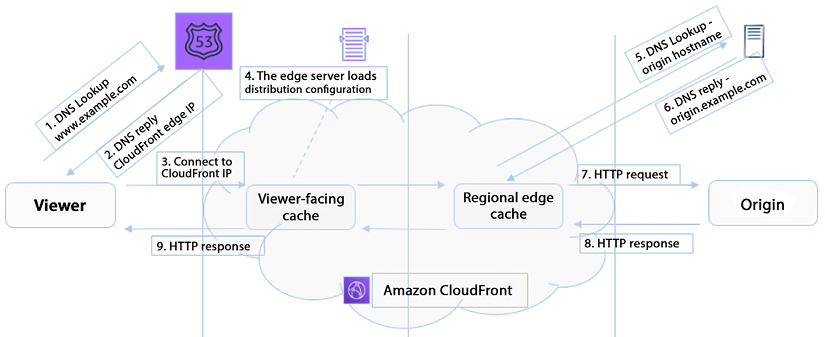
While Route 53 helps to direct global traffic to your server, there could be latency if you wanted to deliver significant static assets, such as images and videos, to users far from your servers’ deployment Region. AWS provides CloudFront as a content distribution network to solve these latency problems. Let’s learn more about it.

**Amazon CloudFront**

Amazon CloudFront is a content delivery service that accelerates the distribution of both static and dynamic content like image files, video files, and JavaScript, CSS, or HTML files, through a network of data centers spread across the globe. These data centers are referred to as **edge locations**. When you use CloudFront to distribute your content, users requesting content get served by the nearest edge location, providing lower latency and better performance.

As of 2022, AWS has over 300 high-density edge locations spread across over 90 cities in 47 countries. All edge locations are equipped with ample cache storage space and intelligent routing mechanisms to increase the edge cache hit ratio. AWS content distribution edge locations are connected with high-performance 100 GbE network devices and are fully redundant, with parallel global networks with default physical layer encryption.

Suppose you have an image distribution website, www.example.com, hosted in the US, which serves art images. Users can access the URL www.example.com/art.png, and the image is loaded. If your server is close to the user, then the image load time will be faster, but if users from other locations like Australia or South Africa want to access the same URL, the request has to cross multiple networks before delivering the content to the user’s browser. The following diagram shows the HTTP request flow with Amazon CloudFront.



*Figure 4.10: HTTP request flow with Amazon CloudFront*

As shown in the preceding diagram, when a viewer requests access to page content from the origin server – in this case, www.example.com – Route 53 replies with the CloudFront edge IP and redirects the user to the CloudFront location. CloudFront uses the following rules for content distribution:

* If the requested content is already in the edge data center, which means it is a “cache hit,” it will be served immediately.
* If the content is not at the edge location (a “cache miss”), CloudFront will request the content from the original location (the web server or S3). The request flows through the AWS backbone, is delivered to the customer, and a copy is kept for future requests.
* If you are using CloudFront, it also provides an extra layer of security since your origin server is not directly exposed to the public network.

CloudFront eliminates the need to go to the origin server for user requests, and content is served from the nearest location. CloudFront provides security by safeguarding the connection between end-users and the content edge, as well as between the edge network and the origin. By offloading SSL termination to CloudFront, the performance of applications is improved since the burden of processing the required negotiation and SSL handshakes is removed from the origins.

CloudFront is a vast topic, and you can learn more about it here – <https://aws.amazon.com/cloudfront/features/>.

In this section, you learned how CloudFront improves performance for both cacheable content and a wide range of applications over TCP, UDP, and MQTT. To address this traffic, AWS provides **AWS Global Accelerator** (**AGA**), which improves the availability and performance of your applications with local or global users. Let’s learn more about it.

**AWS Global Accelerator (AGA)**

AGA enhances application availability and performance by offering fixed static IP addresses as a single entry points, or multiple entry points, to AWS Regions, including ALBs, NLBs, and EC2 instances. AGA utilizes the AWS global network to optimize the path from users to applications, thereby improving the performance of TCP and UDP traffic. AGA continuously monitors the health of application endpoints and promptly redirects traffic to healthy endpoints within 1 minute, in the event of an unhealthy endpoint detection.

AGA and CloudFront are distinct services offered by AWS that employ the AWS global network and its edge locations. While CloudFront accelerates the performance of both cacheable (e.g., videos and images) and dynamic (e.g., dynamic site delivery and API acceleration) content, AGA enhances the performance of various applications over TCP or UDP. Both services are compatible with AWS Shield, providing protection against DDoS attacks. You will learn more about AWS Shield in *Chapter 8*, *Best Practices for Application Security, Identity, and Compliance*.

AGA automatically reroutes your traffic to the nearest healthy endpoint to avoid failure. AGA health checks will react to customer backend failure within 30 seconds, which is in line with other AWS load-balancing solutions (such as NLB) and Route 53. Where AGA raises the bar is with its ability to shift traffic to healthy backends in as short a timeframe as 30 seconds, whereas DNS-based solutions can take minutes to hours to shift the traffic load. Some key reasons to use AGA are:

* **Accelerate your global applications** – AGA intelligently directs TCP or UDP traffic from users to the AWS-based application endpoint, providing consistent performance regardless of their geographic location.
* **Improve global application availability** – AGA constantly monitors your application endpoints, including but not limited to ALBs, NLBs, and EC2 instances. It instantly reacts to changes in their health or configuration, redirecting traffic to the next closest available endpoint when problems arise. As a result, your users experience higher availability. AGA delivers inter-Region load balancing, while ELB provides intra-Region load balancing.

ELB in a Region is a suitable candidate for AGA as it evenly distributes incoming application traffic across backends, such as Amazon EC2 instances or ECS tasks, within the Region. AGA complements ELB by expanding these capabilities beyond any single Region, enabling you to create a global interface for applications with application stacks located in a single Region or multiple Regions.

* **Fixed entry point** – AGA provides a set of static IP addresses for use as a fixed entry point to your AWS application. Announced via anycast and delivered from AWS edge locations worldwide, these eliminate the complexity of managing the IP addresses of multiple endpoints and allow you to scale your application and maintain DDoS resiliency with AWS Shield.
* **Protect your applications** – AGA allows you to serve internet users while keeping your ALBs and EC2 instances private.

AGA allows customers to run global applications in multiple AWS Regions. Traffic destined to static IPs is globally distributed, and end user requests are ingested through AWS’s closest edge location and routed to the correct regional resource for better availability and latency. This global endpoint supports TCP and UDP and does not change even as customers move resources between Regions for failover or other reasons (i.e., client applications are no longer tightly coupled to the specific AWS Region an application runs in). Customers will like the simplicity of this managed service.

As technology becomes more accessible with the high-speed networks provided by 5G, there is a need to run applications such as connected cars, autonomous vehicles, and live video recognition with ultra-low latency. AWS provides a service called AWS Wavelength, which delivers AWS services to the edge of the 5G network. Let’s learn more about it.

**AWS Wavelength**

AWS Wavelength is designed to reduce network latency when connecting to applications from 5G-connected devices by providing infrastructure deployments within the Telco 5G network service providers’ data centers. It allows application traffic to reach application servers running in Wavelength Zones, as well as AWS compute and storage services. This eliminates the need for traffic to go through the internet, which can introduce latency of up to 10s of milliseconds and limit the full potential of the bandwidth and latency advancements of 5G.

AWS Wavelength allows for the creation and implementation of real-time, low-latency applications, such as edge inference, smart factories, IoT devices, and live streaming. This service enables the deployment of emerging, interactive applications that require ultra-low latency to function effectively.

Some key benefits of AWS Wavelength are:

* **Ultra-low latency for 5G** – Wavelength combines the AWS core services, such as compute and storage, with low-latency 5G networks. It helps you to build applications with ultra-low latencies using the 5G network.
* **Consistent AWS experience** – You can use the same AWS services you use daily on the AWS platform.
* **Global 5G network** – Wavelength is available in popular Telco networks such as Verizon, Vodafone, and SK Telecom across the globe, including the US, Europe, Korea, and Japan, which enables ultra-low latency applications for a global user base.

Wavelength Zones are connected to a Region and provide access to AWS services. Architecting edge applications using a hub-and-spoke model with the Region is recommended for scalable and cost-effective options for less latency-sensitive applications.

As enterprises adopt the cloud, it will not be possible to instantly move all IT workloads to the cloud. Some applications need to run on-premises and still communicate with the cloud. Let’s learn about AWS services for setting up a hybrid cloud.

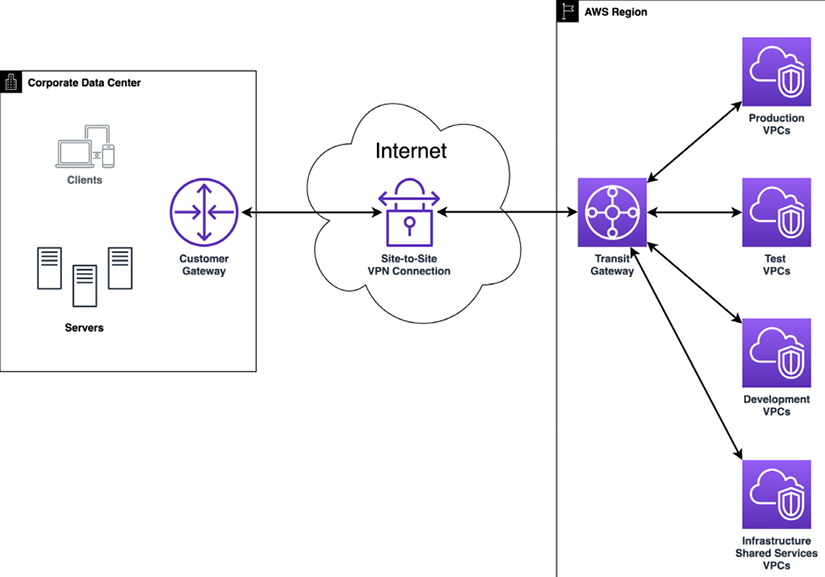
**Building hybrid cloud connectivity in AWS**

A hybrid cloud comes into the picture when you need to keep some of your IT workload on-premises while creating your cloud migration strategy. You may have decided to keep them out of the cloud for various reasons, such as compliance and the unavailability of out-of-the-box cloud services such as a mainframe, when you need more time to re-architect them, or when you are waiting to complete your license term with an existing vendor. In such cases, you need highly reliable connectivity between your on-premises and cloud infrastructure. Let’s learn about the various options available from AWS to set up hybrid cloud connectivity.

**AWS Virtual Private Network (VPN)**

AWS VPN is a networking service to establish a secure connection between AWS, on-premises networks, and remote client devices. There are two variants of AWS VPN: Site-to-Site VPN and AWS Client VPN. AWS Site-To-Site VPN establishes a secure tunnel between on-premises and AWS Virtual Private Gateways or AWS TGWs.

It offers fully managed and highly available VPN termination endpoints at AWS Regions. You can add two VPN tunnels per VPN connection, which is secured with an IPsec Site-to-Site tunnel with AES-256, SHA-2, and the latest DH groups. The following diagram shows the AWS Site-to-Site VPN connection.

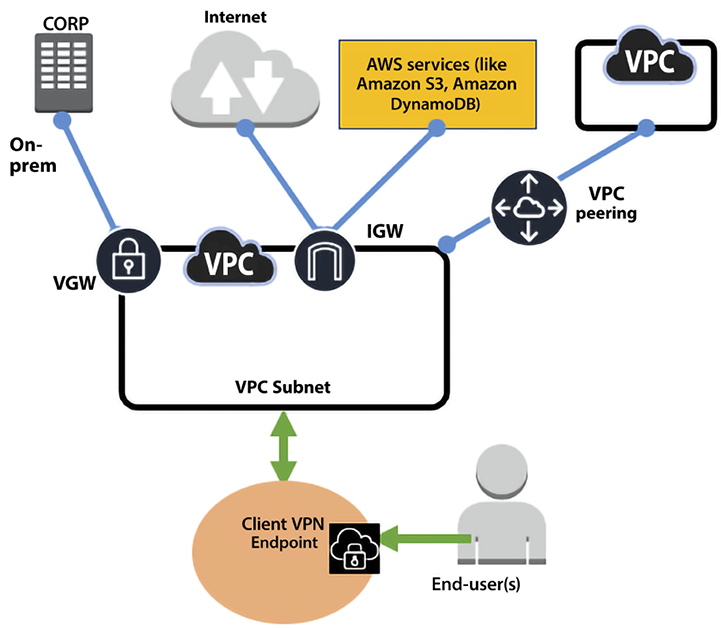


*Figure 4.11: AWS Site-to-Site VPN with TGW*

As depicted in the preceding diagram, a connection has been established from the customer gateway to the AWS TGW using Site-To-Site VPN, which is further connected to multiple VPCs.

AWS Client VPN can be used with OpenVPN-based VPN client software to access your AWS resources and on-premises resources from any location across the globe.

AWS Client VPN also supports a split tunneling feature, which can be used if you only want to send traffic destined to AWS via Client VPN and the rest of the traffic via a local internet breakout.



*Figure 4.12: AWS Client VPN*

AWS Client VPN provides secure access to any resource in AWS and on-premises from anywhere using OpenVPN clients. It seamlessly integrates with existing infrastructure, like Amazon VPC, AWS Directory Service, and so on.

**AWS Direct Connect**

**AWS Direct Connect** is a low-level infrastructure service that enables AWS customers to set up a dedicated network connection between their on-premises facilities and AWS. You can bypass any public internet connection using AWS Direct Connect and establish a private connection linking your data centers with AWS. This solution provides higher network throughput, increases the consistency of connections, and, counterintuitively, can often reduce network costs.

There are two variants of Direct Connect, dedicated and hosted. A dedicated connection is made through a 1 Gbps, 10 Gbps, or 100 Gbps dedicated Ethernet connection for a single customer. Hosted connections are obtained via an AWS Direct Connect Delivery Partner, who provides the connectivity between your data center and AWS via the partner’s infrastructure.

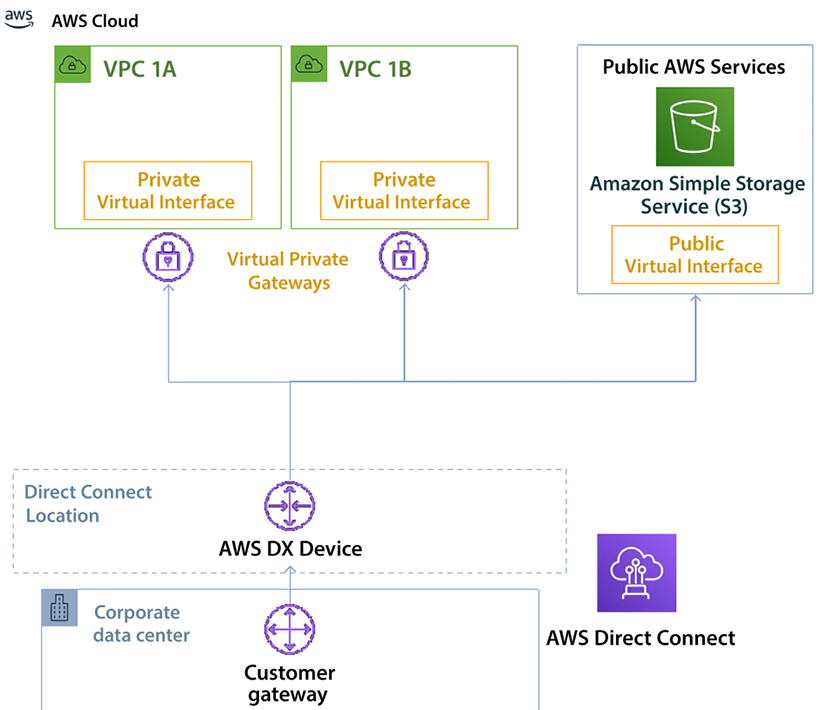
However, it is essential to note that AWS Direct Connect does not provide encryption in transit by default. Suppose you want to have encryption in transit. In that case, you have two choices – you can either use AWS Site-To-Site VPN to provide IPsec encryption for your packets, or you can combine AWS Direct Connect with AWS Site-to-Site VPN to deliver an IPsec-encrypted private connection while, at the same time, lowering network costs and increasing network bandwidth throughput.

The other option is to activate the MACsec feature, which provides line-rate, bi-directional encryption for 10 Gpbs and 100 Gpbs dedicated connections between your data centers and AWS Direct Connect locations. MACsec is done at the hardware; hence, it provides better performance. To encrypt the traffic, you can also use an AWS technology partner as an alternative solution to encrypt this network traffic.

AWS Direct Connect uses the 802.1q industry standard to create VLANs. These connections can be split into several **virtual interfaces** (**VIFs**). This enables us to leverage the same connection to reach publicly accessible services such as Amazon S3 by using an IP address space and private services such as EC2 instances running in a VPC within AWS. The following are AWS Direct Connect interface types:

* **Public virtual interface:** This is the interface you can use to access any AWS public services globally, which are accessible via public IP addresses such as Amazon S3. A public VIF can access all AWS public services using public IP addresses.
* **Private virtual interface**: You can connect to your VPCs using private VIFs using private IP addresses. You can connect to the AWS Direct Connect gateway using a private VIF, allowing you to connect to up to 10 VPCs globally with a single VIF, unlike connecting a private VIF to a Virtual Private Gateway associated with a single VPC.
* **Transit virtual interface**: A transit VIF is connected to your TGW via a Direct Connect gateway. A transit VIF is supported for a bandwidth of 1 Gbps or higher.

The following diagram has put together various AWS Direct Connect interfaces, showing the use of various interfaces while designing your hybrid cloud connectivity.



*Figure 4.13: AWS Direct Connect interface types*

The preceding diagram shows the corporate data center connected to the AWS cloud using the Direct Connect location. Most of your application workloads, such as the web server, app server, and database server, run inside the VPC under a private restricted network, so a private VIF connects to VPCs across different AZs. Conversely, Amazon S3 is in the public domain, where you might host static pages, images, and videos for your applications connected through a public VIF.

AWS Direct Connect can reduce costs when workloads require high bandwidth. It can reduce these costs in two ways:

* It transfers data from on-premises environments to the cloud, directly reducing cost commitments to **Internet Service Providers** (**ISPs**).
* The costs of transferring the data using a dedicated connection are billed using the AWS Direct Connect data transfer rates and not the internet data transfer rates, which are lower.

Network latency and responses to requests can be highly variable. Workloads that use AWS Direct Connect have a much more homogenous latency and consistent user experience. Direct connection comes with a cost; you may only sometimes need such high bandwidth and want to optimize costs better. For such cases, you may want to use AWS VPN, as discussed in the *AWS Virtual Private Network (VPN)* section.

You have learned about different patterns of setting up network connectivity within an AWS cloud and to or from an AWS cloud, but for a large enterprise with branch offices or chain stores, connecting multiple data centers, office locations, and cloud resources can be a very tedious task. AWS provides Cloud WAN to simplify this issue. Let’s learn more about it.

**AWS Cloud WAN**

How many of you would be paged if your entire network suddenly went down? Let’s take an example, Petco, which has over 1,500 locations. Imagine what happens on a network like that on any given day. One way to connect your data centers and branch offices is to use fixed, physical network connections. These connections are long-lived and not easy to change quickly.

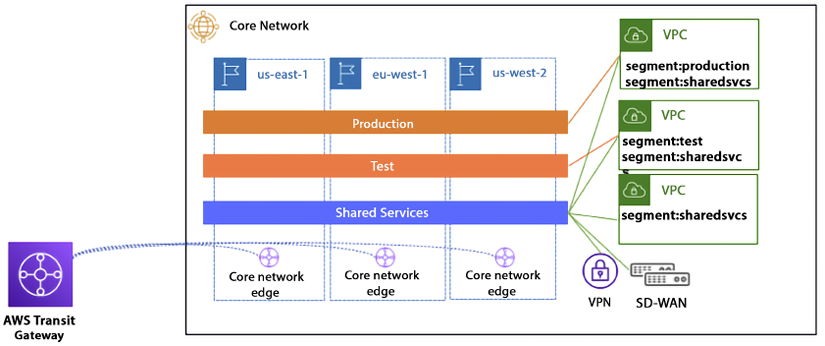
Many use AWS Site-to-Site VPN connections for connectivity between their locations and AWS. Alternatively, you bypass the internet altogether and use AWS Direct Connect to create a dedicated network link to AWS. And some use broadband internet with SD-WAN hardware to create virtual overlay networks between locations. Inside AWS, you build networks within VPCs and route traffic between them with TGW. The problem is that these networks all take different approaches to connectivity, security, and monitoring. As a result, you are faced with a patchwork of tools and networks to manage and maintain.

For example, to keep your network secure, you must configure firewalls at every location, but you are faced with many different firewalls from many different vendors, and each is configured slightly differently than the others. Ensuring your access policies are synced across the entire network quickly becomes daunting. Likewise, managing and troubleshooting your network is difficult when the information you need is kept in many different systems.

Every new location, network appliance, and security requirement makes things more and more complicated. We see many customers struggle to keep up. To solve these problems, network architects need a way to unify their networks so that there is one central place to build, manage, and secure their network. They need easy ways to make and change connections between their data centers, branch offices, and cloud applications, regardless of what they’re running on today. And they need a backbone network that can smoothly adapt to these changes.

AWS Cloud WAN is a global network service enabling you to create and manage your infrastructure globally in any AWS Region or on-premises. Consider it a global router that you can use to connect your on-premises networks and AWS infrastructure. AWS Cloud WAN provides network segmentation to group your network or workloads, which can be located across any AWS Region. It takes care of the route propagation to connect your infrastructure without manually maintaining the routing.

AWS Cloud WAN provides network segmentation, which means you can divide your global network into separate and isolated networks. This helps you to control the traffic flow and cross-network access tightly. For example, a corporate firm can have a network segment for invoicing and order processing, another for web traffic, and another for logging and monitoring traffic.



*Figure 4.14: AWS Cloud WAN architecture*

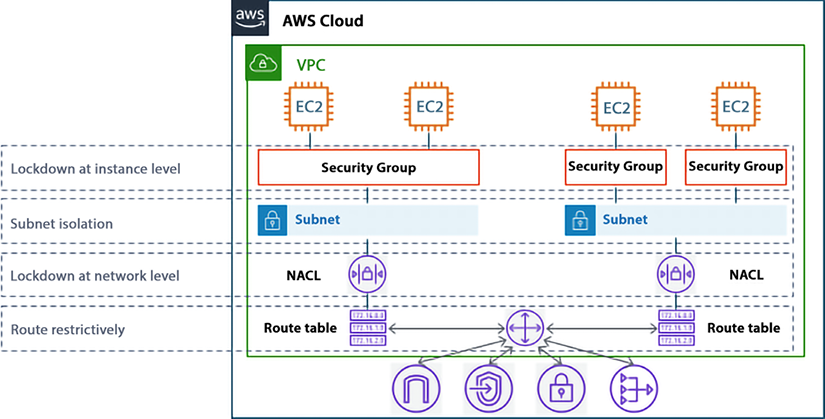
When using Cloud WAN, you can see your entire network on one dashboard, giving you one place to monitor and track metrics for your entire network. Cloud WAN lets you spot problems early and respond quickly, which minimizes downtime and bottlenecks while helping you troubleshoot problems, even when data comes from separate systems. Security is the top priority, and throughout this chapter, you have learned about the security aspect of network design.

Let’s go into more detail to learn about network security best practices.

**AWS cloud network security**

Security is always a top priority for any organization. As a general rule, you need to protect every infrastructure element individually and as a group – like the saying, “Dance as if nobody is watching and secure as if everybody is.” AWS provides various managed security services and a well-architected pillar to help you design a secure solution.

You can implement secure network infrastructure by creating an allow-list of permitted protocols, ports, CIDR networks, and SG sources, and enforcing several policies on the AWS cloud. A NACL is used to explicitly block malicious traffic and segment the edge connectivity to the internet, as shown in the following diagram.



*Figure 4.15: SGs and NACLs in network defense*

As shown in the preceding diagram, SGs are the host-level virtual firewalls that protect your EC2 instance from the traffic coming to or leaving from a particular instance. An SG is a stateful firewall, which means that you need to create a rule in one direction only, and the return traffic will automatically be allowed. A NACL is also a virtual firewall that sits at the subnet boundary, regulating traffic in and out of one or more subnets. Unlike an SG, a NACL is stateless, which means you need both inbound and outbound rules to allow specific traffic.

The following table outlines the difference between SGs and NACLs.

|  |  |
| --- | --- |
| **SG** | **NACL** |
| The SG is the first layer of defense, which operates at the instance level. | A NACL works at the subnet level inside an AWS VPC. |
| You can only add “allow” security rules. | You can explicitly add a “deny” rule in addition to allow rules. For example, you can deny access to a specific IP address. |
| The SG is stateful, which means once you add an inbound allow rule, it automatically adds an outbound allow rule. For example, for a CRM server to respond, you have only to allow a rule to accept traffic from the IP range. | A NACL is stateless, meaning if you add an inbound allow rule, you must add an explicit allow rule. For a CRM server to respond, you must add both allow and deny rules to accept traffic from the IP range. |
| If you have added multiple rules in the instance, the SG will validate all rules before deciding whether to allow traffic. | In a NACL, you define rule priority by assigning values, such as 100 or 200. The NACL processes the rules in numerical order. |
| As an SG is at the instance level, it applies to an instance only. | As a NACL is at the subnet level, it automatically applies to all instances in the subnets it’s linked to. |

*Table 4.1: Comparison between security groups and network access control lists*

While the SG and NACL provide security inside the VPC, let’s learn more about overall AWS network security best practices.

**AWS Network Firewall (ANFW)**

ANFW is a highly available, fully redundant, and easy-to-deploy managed network firewall service for your Amazon VPC, which offers a service-level agreement with an uptime commitment of 99.99%. ANFW scales automatically based on your network traffic, eliminating the need for the capacity planning, deployment, and management of the firewall infrastructure.

ANFW supports open source Suricata-compatible rules for stateful inspection. It provides fine controls for your network traffic, such as allowing or blocking specific protocol traffic from specific prefixes. ANFW also supports third-party integration to source-managed intelligent feeds.

ANFW also provides alert logs that detail a particular rule that has been triggered. There is native integration with Amazon S3, Amazon Kinesis, and Amazon CloudWatch, which can act as the destination for these logs.

You can learn about various ANFW deployment models by referring to the detailed AWS blog here – <https://aws.amazon.com/blogs/networking-and-content-delivery/deployment-models-for-aws-network-firewall/>.

Let’s learn more about network security patterns and anti-patterns, keeping SGs and NACL in mind.

**AWS network security patterns – best practices**

Several patterns can be used when creating an SG and NACL strategy for your organization, such as:

* **Create the SG before launching the instance(s), resource(s), or cluster(s)** – This will force you to determine if a new SG is necessary or if an existing SG should be used. This enables you to pre-define all the rules and reference the SG at creation time, especially when creating these programmatically or via a CloudFormation/Landing Zone pipeline. Additionally, you should not use the default SG for your applications.
* **Logically construct SGs into functional categories based on their application tier or role they perform** – Consider the number of distinct tiers your application has and then logically construct the SGs to match those functional components. For a typical three-tier architecture, a minimum of three SGs should be used (e.g., a web tier, an app tier, and a DB tier).
* **Configure rules to chain SGs to each other** – In an SG rule, you can authorize network access from a specific CIDR address range or another SG in your VPC. Either option could be appropriate, depending on your environment. Generally speaking, organizations can “chain” the SGs together between application tiers, thus building a logical flow of allowed traffic from one SG to another. However, an exception to this pattern is described in the following pattern.
* **Restrict privileged administrative ports (e.g., SSH or RDP) to internal systems (e.g., bastion hosts)** – As a best practice, administrative access to instances should be blocked or restricted to a small number of protected and monitored instances – sometimes referred to as bastion hosts or jump boxes.
* **Create NACL rule numbers with the future in mind** – NACLs rules are evaluated in numerical order based on the rule number, and the first rule that matches the traffic will be used. For example, if there are 10 rules in an NACL and rule number 2 matches DENY SSH traffic on port 22, the other 8 rules never get evaluated, regardless of their content. Therefore, when creating NACL rules, it is best practice to leave gaps between the rule numbers – 100 is typically used. So, the first rule has a rule number of 100, and the second rule has a rule number of 200.

If you have to add a rule between these rules in the future, you can add a new rule with rule number 150, which still leaves space for future planning.

* **In well-architected, high-availability VPCs, share NACLs based on subnet tiers** – The subnet tiers in a well-architected, high-availability VPC should have the same resources and applications deployed in the same subnet tiers (e.g., the web and app tiers). These tiers should have the same inbound and outbound rule requirements. In this case, it is recommended to use the same NACL to avoid making administrative changes in multiple places.
* **Limit inbound rules to the minimum required ports and restrict access to commonly vulnerable ports** – As with hardware firewalls, it is important to carefully determine the minimum baseline for inbound rules required for an application tier to function. Reducing the ports allowed greatly reduces the overall attack surface and simplifies the management of the NACLs.
* **Finally, audit and eliminate unnecessary, unused, or redundant NACL rules and SGs** – As your environment scales, you may find that unnecessary SGs and NACL rules were created or mistakenly left behind from previous changes.

**AWS network security anti-patterns**

While the previous section described patterns for using VPC SGs and NACLs, there are a number of ways you might attempt to configure or utilize your SGs and NACL in non-recommended ways. These are referred to as “anti-patterns” and should be avoided.

* The default SG is included automatically with your VPC. Whenever you launch an EC2 instance or any other AWS resource in your VPC, it is linked to the default SG. Using the default SG does not provide granular control. You can create custom SGs and add them to instances or resources. You can create multiple SGs as per your applications’ needs, such as a web server EC2 instance or an Aurora database server.
* If you already have SGs applied to all instances, **do not** create a rule that references these in other SGs. By referencing an SG applied to all instances, you are defining a source or destination of all instances. This is a wide-open pointer to or from everything in your environment. There are better ways to apply least-privileged access.
* Multiple SGs can be applied to an instance. Within a specific application tier, all instances should have the same set of SGs applied to them. However, ensure a tier’s instances have a uniform consistency. Suppose web servers, application servers, and database servers co-mingle in the same tier. In this case, they should be separated into distinct tiers and have tier-specific SGs applied to them. **Do not** create unique SGs for related instances (one-off configurations or permutations).
* Refrain from sharing or reusing NACLs in subnets with different resources – although the NACL rules may be the same now, there is no way to determine future rules required for the resources. Since the resources in the subnet differ (e.g., between different applications), resources in one subnet may need a new rule that isn’t needed for the resources in the other subnet. However, since the NACL is shared, opening up the rule applies resources in both subnets. This approach does not follow the principle of least privilege.
* NACLs are stateless, so rules are evaluated when traffic enters and leaves the subnet. You will need an inbound and outbound rule for each two-way communication. This anti-pattern shouldn’t be encountered because you are only using NACLs as guardrails. Evaluating large complex rule sets on traffic coming in and out of the subnet will eventually lead to performance degradation.

It is recommended to periodically audit for SGs and NACLs rules that are unnecessary or redundant and delete them. This will reduce complexity and help prevent reaching the service limit accidentally.

**AWS network security with third-party solutions**

You may not always want to get into all the nitty-gritty details of AWS security configuration and look for more managed solutions. AWS’s extensive partner network builds managed AWS solutions to fulfill your network security needs. Some of the most popular network security solutions provided by the following **integrated software vendor** (**ISV**) partners in AWS Marketplace are below:

* **Palo Alto Networks** – Palo Alto Networks has introduced a Next-Generation Firewall service that simplifies securing AWS deployments. This service enables developers and cloud security architects to incorporate inline threat and data loss prevention into their application development workflows.
* **Aviatrix** – The Aviatrix Secure Networking Platform is made up of two components: the Aviatrix controller (which manages the gateways and orchestrates all connectivity) and the Aviatrix Gateways that are deployed in VPCs using AWS IAM roles.
* **Check Point** – Check Point CloudGuard Network Security is a comprehensive security solution designed to protect your AWS cloud environment and assets. It provides advanced, multi-layered network security features such as a firewall, IPS, application control, IPsec VPN, antivirus, and anti-bot functionality.
* **Fortinet** – This provides firewall technology to deliver complete content and network protection, including application control, IPS, VPN, and web filtering. It also provides more advanced features like vulnerability management and flow-based inspection work.
* **Cohesive Networks** – Cohesive’s VNS3 is a software-only virtual appliance for connectivity, federation, and security in AWS.

In addition to these, many more AWS-managed solutions are available through partners like Netskope, Valtix, IBM, and Cisco. You can find the complete list of network security solutions available in the AWS Marketplace using this link – <https://aws.amazon.com/marketplace/search/results?searchTerms=network+security>.

AWS cloud security has multiple components, starting with networking as a top job. You can learn more about AWS security by visiting their security page at <https://aws.amazon.com/security/>.

**Summary**

In this chapter, you started with learning about the AWS Global Infrastructure and understanding the details of AWS Regions, AZs, and LZs. You also learned about the various benefits of using the AWS Global Infrastructure.

Networking is the backbone of any IT workload, whether in the cloud or in an on-premises network. To start your cloud journey in AWS, you must have good knowledge of AWS networking. When you start with AWS, you create your VPC within AWS. You learned about using an AWS VPC with various components such as an SG, a NACL, a route table, an IGW, and a NAT gateway. You learned how to segregate and secure your IT resources by putting them into private and public subnets.

With the ease of creating VPC in AWS organizations, multiple VPCs tend to be created, whether it is intentional to give each team their own VPC, or unintentional when the dev team creates multiple test workloads. Often, these VPCs need to communicate with each other; for example, the finance department needs to get information from accounting. You learned about setting up communication between multiple VPCs using VPC peering and TGW. You learned to establish secure connections with services on the public internet or other accounts using AWS PrivateLink.

Further, you learned about AWS edge networking to address the global nature of user traffic. These services include Route 53, CloudFront, AGA, and AWS Wavelength. You then learned about connecting an on-premises server and an AWS cloud. Finally, you closed the chapter with the network security best practices, patterns and anti-patterns, and third-party managed network security solutions available via the AWS partner network.

As you start your journey of learning about the AWS core services, the next topic is storage. AWS provides various types of storage to match your workload needs. In the next chapter, you will learn about storage in AWS and how to choose the right storage for your IT workload needs.