

MALLA REDDY UNIVERSITY
(MR22-1CS0108) CLOUD COMPUTING

UNIT III

AWS EBS: Create EBS volumes, Delete EBS Volumes, Attach and detach EBS volumes, Mounting and unmounting EBS volume.

Networking and Content Delivery: Networking basics, Amazon VPC, VPC networking, VPC security, Route 53, Cloud Front.

Amazon Elastic Block Store (Amazon EBS):

Amazon EBS enables you to create individual storage volumes and attach them to an Amazon EC2 instance.

Amazon EBS offers **block-level storage**, where its volumes are automatically replicated within its Availability Zone.

Amazon EBS is designed to provide durable, detachable, block-level storage (which is like an external hard drive) for your Amazon EC2 instances.

Because they are directly attached to the instances, they can provide low latency between where the data is stored and where it might be used on the instance.

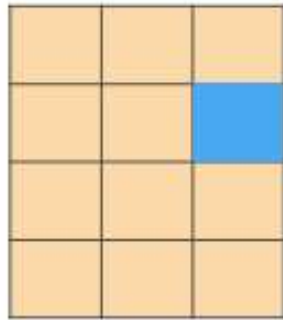
Amazon EBS volumes uses include:

- Boot volumes and storage for Amazon EC2 instances
- Data storage with a file system
- Database hosts
- Enterprise applications

AWS storage options: Block storage versus object storage

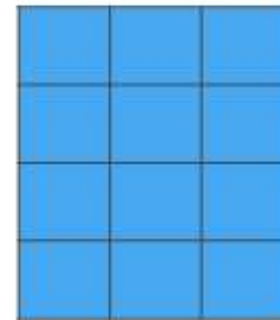


What if you want to change **one character** in a 1-GB file?



Block storage

Change one block (piece of the file)
that contains the character



Object storage

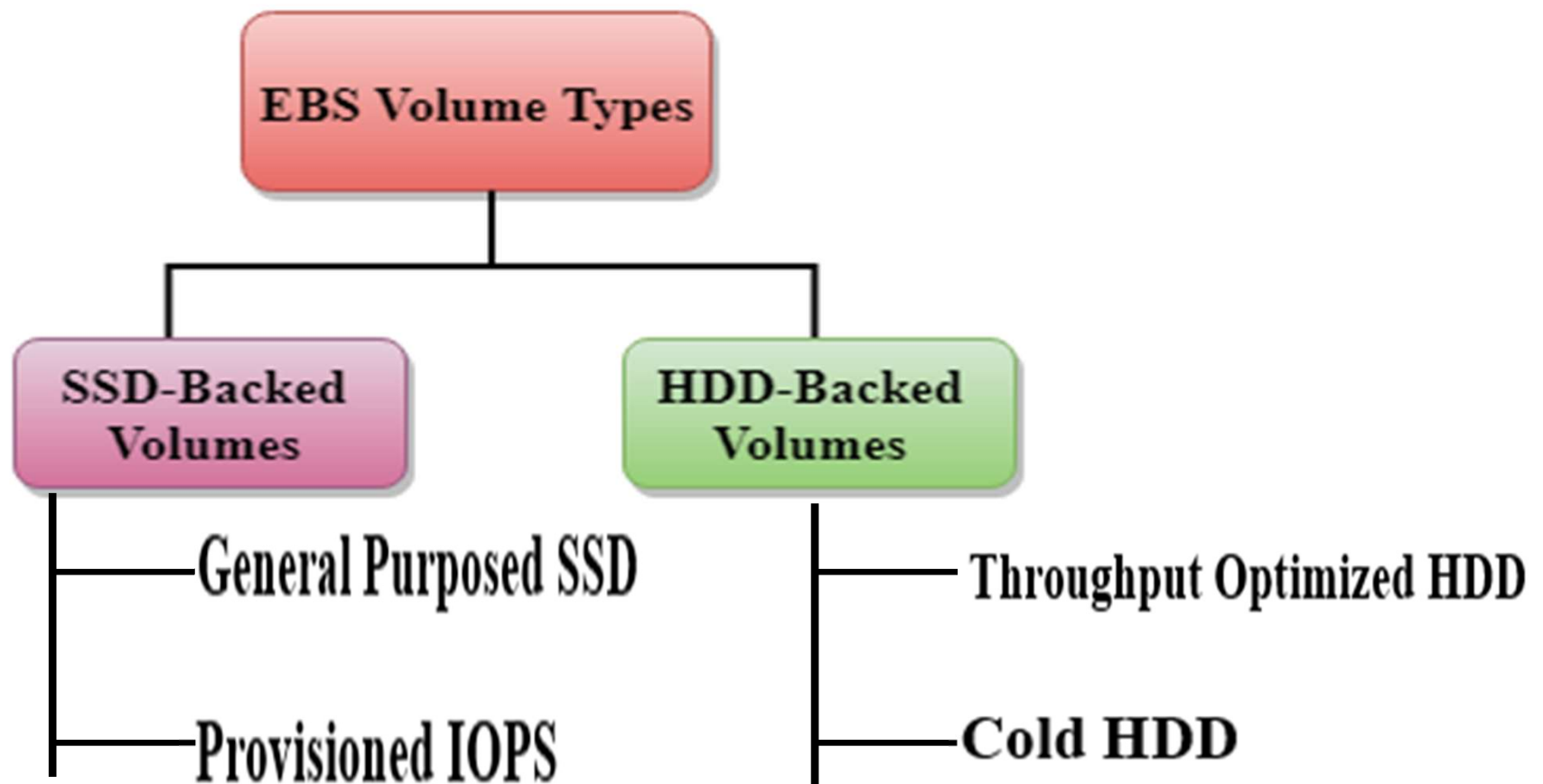
Entire file must be updated

What happens if you want to change one character in a 1-GB file? With block storage, you change only the block that contains the character. With object storage, the entire file must be updated.

Amazon EBS volume types:

These are broadly categorized into

- **SSD Backed Volumes**- ideal for transactional workloads like databases and Virtual desktops
- **HDD Backed Volumes**-suited for throughput-intensive tasks like MapReduce



Amazon EBS volume types:

Solid State Drive (SSD) volumes

Solid State Drives are usually preferred in cases where **high Input Output per Second (IOPS) is required** during a random or sequential I/O operation and are well suited for transactional workloads where the key performance attribute is IOPS.

. They are especially suitable for demanding applications like SAP HANA, Microsoft SQL Server, and IBM DB2.

SSDs in AWS EBS volume types are further classified into

- 1) **SSD-backed General Purpose SSD**

- 2) **SSD-backed Provisioned IOPS**

1) SSD-backed General Purpose SSD volumes

General Purpose SSDs strike a balance between **IOPS (Input/Output Operations Per Second)** and **throughput**, making them ideal for applications that need **a mix of read/write operations and quick response times**.

It provide a high-performance SSD volume at a comparably **lower price**, hence providing a balance between **cost and performance**, and are well suited for most workloads, including boot volumes, medium-sized databases, development environments, virtual desktops, MySQL, Cassandra and Hadoop analytics clusters etc. .

General Purpose SSD volumes in EBS volume types are **divided into**:

(i) GP2: The default EBS volume type for EC2 instances is gp2 volumes, and are highly recommended as **boot volumes** for EC2 instances. A gp2 volume can range in size from **1 GiB to 16 TiB**, while the performance of a gp2 volume scales linearly with its size, larger gp2 volumes have higher baseline performance levels.

(ii) GP3: The most recent generation of SSD general purpose volumes are referred to as gp3 volumes. At a cheaper cost than gp2, the gp3 volumes provide a constant baseline of **3000 IOPS and 125MB/s** throughput regardless of the storage size and additional performance can be provisioned for an additional fee.

2) SSD backed Provisioned IOPS:

- Provisioned IOPS SSD volumes are the **high-performance SSD volumes used for IOPS and throughput-intensive workloads** tailored for **mission-critical applications** with predictable, sub-millisecond latencies.
- It currently offers **io1, io2 and io2 Block Express**. For each of the three **IOPS SSD EBS** volume types, a different amount is paid based on the provisioned **IOPS** per month.
- These EBS volume types deliver the provisioned performance 99.9% of the time, making them ideal for applications like SAP HANA, Oracle, Microsoft SQL Server, and IBM DB2 that require high availability and reliability.

HDD-backed Storage Volumes

HDD-backed storage is ideal for workloads that need **high throughput**, during a large and sequential I/O operation such as reading video files, logs processing, MapReduce tasks etc.

HDDs are **cheaper compared to SSDs** and ideal for tasks that require high throughput.

HDDs in AWS EBS are further classified into

- 1) **Throughput Optimized HDD Volumes**
- 2) **Cold HDD Volumes**

1) **Throughput Optimized HDD volumes:**

- Throughput Optimized HDD volumes are designed for workloads with **high sequential data access patterns**.
- They provide **good throughput at a lower cost per gigabyte than SSDs**, making them suitable for tasks like Amazon EMR, ETL, data warehouses, and large-scale data processing.
- These EBS volume types offer a baseline throughput of 40 MB/s per TB and can burst up to 250 MB/s per TB and 500 MB/s per volume, supporting high throughput and I/O-intensive workloads.

2) Cold HDD :

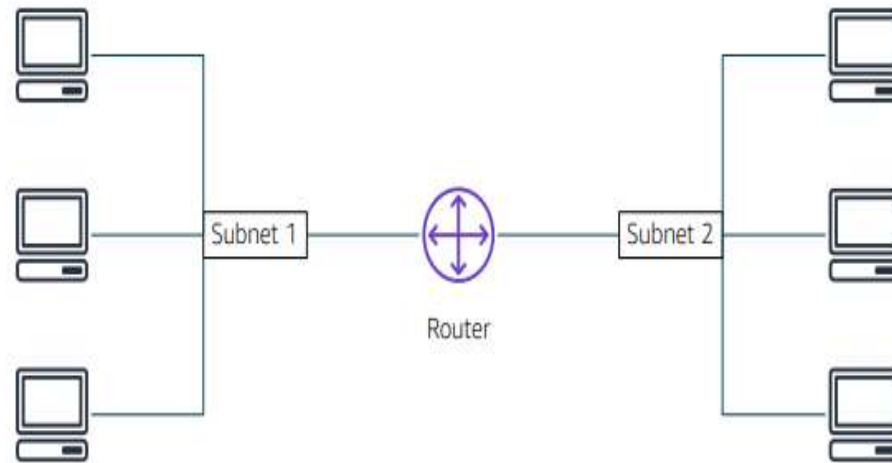
- Cold HDD volumes are the most cost-effective option for **infrequently accessed data, like archives.**
- They offer the lowest storage cost per gigabyte and have the slowest access times, making them perfect for **backups, disaster recovery data, or logs** that are rarely accessed.
- Cold HDD volumes are ideal for workloads with occasional bursts of I/O, providing a baseline throughput of 12 MB/s per TB, with the ability to burst up to 80 MB/s per TB or 250 MB/s per volume.
- This makes them a budget-friendly choice for workloads with relatively low average I/O needs.

| EBS Volume Type | Category | Performance Characteristics | Ideal Use Cases | Cost Consideration |
|---------------------------|------------|--|---|-------------------------------------|
| General Purpose SSD (gp2) | SSD-backed | <ul style="list-style-type: none"> – Baseline 3 IOPS per GB – Burst up to 3,000 IOPS | <ul style="list-style-type: none"> – Boot volumes – Small/medium databases – Development/test environments | Moderate cost, flexible performance |
| General Purpose SSD (gp3) | SSD-backed | <ul style="list-style-type: none"> – Consistent 3,000 IOPS baseline – Up to 16,000 IOPS – Max throughput 1,000 MB/s | <ul style="list-style-type: none"> – MySQL, Cassandra – Virtual desktops – Hadoop analytics clusters | Lower cost per GB compared to gp2 |
| Provisioned IOPS (io1) | SSD-backed | <ul style="list-style-type: none"> – Up to 64,000 IOPS per volume – Predictable, low-latency I/O | <ul style="list-style-type: none"> – Mission-critical databases – Large-scale transactional applications | Higher cost, very high performance |
| Provisioned IOPS (io2) | SSD-backed | <ul style="list-style-type: none"> – Enhanced durability – Same IOPS as io1 | <ul style="list-style-type: none"> – SAP HANA, Oracle | |

| EBS Volume Type | Category | Performance Characteristics | Ideal Use Cases | Cost Consideration |
|--------------------------------|------------|--|--|---------------------------------|
| Throughput Optimized HDD (st1) | HDD-backed | <ul style="list-style-type: none"> – Baseline 40 MB/s per TB – Burst up to 250 MB/s per TB – Max throughput 500 MB/s per volume | <ul style="list-style-type: none"> – Big data, data warehousing – Log processing – Streaming large datasets | Lower cost per GB than SSDs |
| Cold HDD (sc1) | HDD-backed | <ul style="list-style-type: none"> – Baseline 12 MB/s per TB – Burst up to 80 MB/s per TB – Max throughput 250 MB/s per volume | <ul style="list-style-type: none"> – Backup and archival – Disaster recovery data – Rarely accessed logs | Lowest cost, lowest performance |

Networking basics

Networks



A computer network is **two or more client machines that are connected together to share resources.**

A network can be **logically partitioned into subnets.**

Networking requires a **networking device (such as a router or switch)** to connect all the clients together and enable communication between them.

IP addresses

192 . 0 . 2 . 0

↓ ↓ ↓ ↓

11000000 00000000 00000010 00000000

IPv4 and IPv6 addresses

IPv4 (32-bit) address: 192.0.2.0

IPv6 (128-bit) address:

2600:1f18:22ba:8c00:ba86:a05e:a5ba:00FF

Each client machine in a network has a unique Internet Protocol (IP) address that identifies it.

An IP address is a **numerical label in decimal format**. Machines convert that **decimal number to a binary format**.

In this example, **the IP address is 192.0.2.0**. Each of the four dot (.)-separated numbers of the IP address represents **8 bits in octal number format**. That means **each of the four numbers can be anything from 0 to 255**.

The combined total of the four numbers for an **IP address is 32 bits in binary format**.

A **32-bit IP address is called an IPv4 address**. **IPv6 addresses, which are 128 bits**, are also available.

IPv6 addresses can **accommodate more user devices**. An IPv6 address is **composed of eight groups of four letters and numbers that are separated by colons (:)**. In this example, the IPv6 address is 2600:1f18:22ba:8c00:ba86:a05e:a5ba:00FF. Each of the eight colon-separated groups of the IPv6 address represents **16 bits in hexadecimal number format**. That means each of the eight groups can be anything **from 0 to FFFF**. The combined

IP address formats

An IP address **has two parts**:

1. The **network address** is a series of numerical digits pointing to the **network's unique identifier**
2. The **host address** is a series of **numbers indicating the host or individual device identifier on the network**

Until the early 1990s, IP addresses were allocated using the **classful addressing system**. The **total length of the address was fixed, and the number of bits allocated to the network and host portions were also fixed**.

Classful addresses

An IPv4 address consists of 32 bits. Each string of numbers separated by the period consists of 8 bits, represented by 0 to 255 in numerical forms. Organizations could purchase three classes of IPv4 addresses.

Class A

A Class A IPv4 address has **8 network** prefix bits. For example, consider 44.0.0.1, where 44 is the network address and 0.0.1 is the host address.

Class B

A Class B IPv4 address has **16 network** prefix bits. For example, consider 128.16.0.2, where 128.16 is the network address and 0.2 is the host address.

Class C

A Class C IPv4 address has **24 network** prefix bits. For instance, consider 192.168.1.100, where 192.168.1 is the network address and 100 is the host address.

Classless addresses

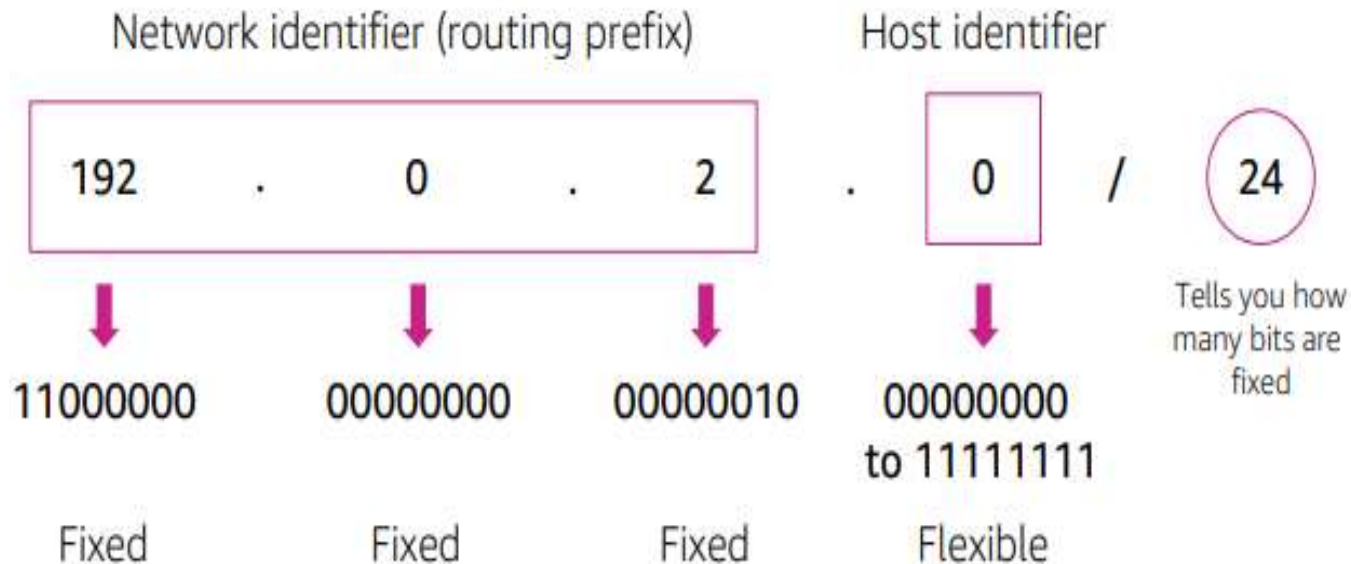
Classless or Classless Inter-Domain Routing (CIDR) addresses use **variable length subnet masking (VLSM)** to **alter the ratio between the network and host address bits in an IP address**.

A subnet mask is a set of identifiers that returns the network address's value from the IP address by turning the host address into zeroes.

A **VLSM sequence allows network administrators to break down an IP address space into subnets of various sizes**.

Each subnet can have a flexible host count and a limited number of IP addresses. A CIDR IP address appends a suffix value stating the number of network address prefix bits to a normal IP address.

Classless Inter-Domain Routing (CIDR)



A common method to describe networks is Classless Inter-Domain Routing (CIDR). The CIDR address is expressed as follows:

- **An IP address (which is the first address of the network)**
- **Next, a slash character (/)**
- **Finally, a number that tells you how many bits of the routing prefix must be fixed or allocated for the network identifier. The bits that are not fixed are allowed to change.**

CIDR is a way to express a group of IP addresses that are consecutive to each other.

In this example, the CIDR address is 192.0.2.0/24. **The last number (24) tells you that the first 24 bits must be fixed. The last 8 bits are flexible, which means that 2^8 ((2^{32-24}) or 256) IP addresses are available for the network, which range from 192.0.2.0 to 192.0.2.255. The fourth decimal digit is allowed to change from 0 to 255.**

If the CIDR was 192.0.2.0/16, the last number (16) tells you that the first 16 bits **must be fixed**. The last 16 bits are flexible, which means that 2^{16} ((2^{32-16}) or 65,536) IP addresses are available for the network, ranging from 192.0.0.0 to 192.0.255.255. The third and fourth decimal digits can each change from 0 to 255.

There are **two special cases**:

- **Fixed IP addresses**, in which every bit is fixed, represent a single IP address (for example, 192.0.2.0/32). This type of address is helpful **when you want to set up a firewall rule** and give access to a specific host.
- The **internet**, in which every bit is flexible, is represented as **0.0.0.0/0**

Open Systems Interconnection (OSI) model

| Layer | Number | Function | Protocol/Addresses |
|--------------|--------|---|--------------------------|
| Application | 7 | Means for an application to access a computer network | HTTP(S), FTP, DHCP, LDAP |
| Presentation | 6 | <ul style="list-style-type: none">Ensures that the application layer can read the dataEncryption | ASCII, ICA |
| Session | 5 | Enables orderly exchange of data | NetBIOS, RPC |
| Transport | 4 | Provides protocols to support host-to-host communication | TCP, UDP |
| Network | 3 | Routing and packet forwarding (routers) | IP |
| Data link | 2 | Transfer data in the same LAN network (hubs and switches) | MAC |
| Physical | 1 | Transmission and reception of raw bitstreams over a physical medium | Signals (1s and 0s) |

The Open Systems Interconnection (OSI) model is a conceptual model that is used to explain how data travels over a network. It consists of **seven layers and shows the common protocols and addresses that are used to send data at each layer.**

For example, hubs and switches work at layer 2 (the data link layer). Routers work at layer 3 (the network layer).

The OSI model can also be used in a **virtual private cloud (VPC),**

Amazon VPC

Amazon Virtual Private Cloud (Amazon VPC) **is a service that lets you provision a logically isolated section of the AWS Cloud** (called a virtual private cloud, or VPC) where you can **launch your AWS resources**.

Amazon VPC gives you

control over your virtual networking resources,

including the selection of your own IP address range,

the creation of subnets, and

the configuration of route tables and network gateways.

You can use **both IPv4 and IPv6** in your VPC for secure access to resources and applications.

You can also **customize the network configuration for your VPC**.

For example, You can create a **public subnet** for your web servers that can access the public internet.

You can place your **backend systems (such as databases or application servers)** in a **private subnet** with no public internet access.

Finally, you can use multiple layers of security, including security groups and network access control lists (network ACLs), to help control access to Amazon Elastic Compute Cloud (Amazon EC2) instances in each subnet

Amazon VPC

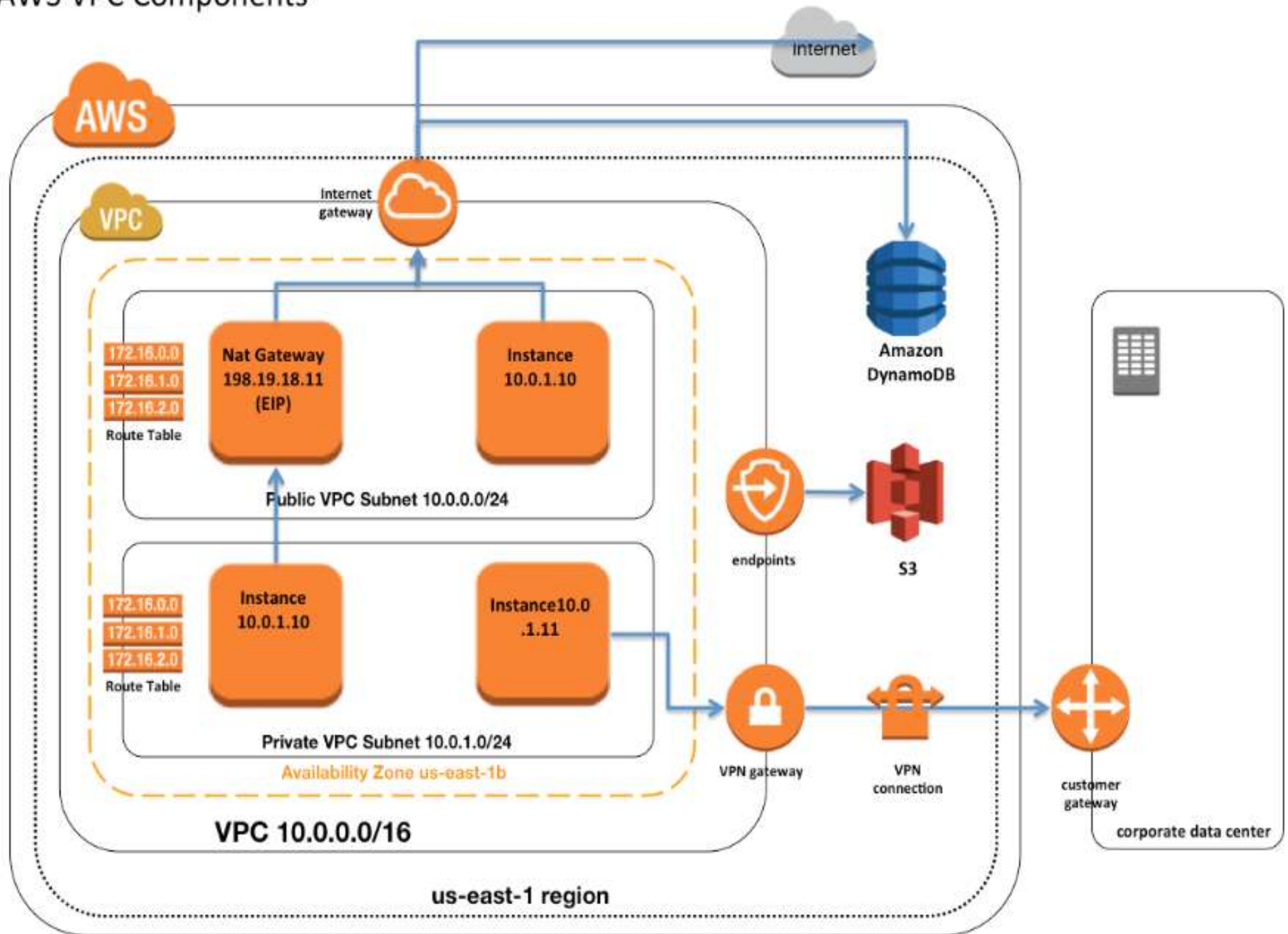


Amazon
VPC

- Enables you to provision a **logically isolated** section of the AWS Cloud where you can launch AWS resources in a virtual network that you define
- Gives you **control over your virtual networking resources**, including:
 - Selection of IP address range
 - Creation of subnets
 - Configuration of route tables and network gateways
- Enables you to **customize the network configuration** for your VPC
- Enables you to use **multiple layers of security**

Components of AWS VPC

AWS VPC Components



1. Internet Gateways – IGW

An Internet gateway is a horizontally scaled, redundant, and highly available VPC component that allows communication between instances in the VPC and the Internet.

2. Subnets

After you create a VPC, you can **divide it into one or more subnets**. A subnet is a **range of IP addresses in a VPC**. Subnets in AWS VPC are used to **divide a VPC into multiple logical networks**. This allows users to isolate resources and control access to those resources. **Subnets can be either public or private**, with public subnets having access to the internet and private subnets not having access to the internet. Instances within the Public Subnet should be assigned a Public IP or Elastic IP address to be able to communicate with the Internet

3. Route Tables

A *route table* contains a **set of rules, called routes**, that are used to determine where network traffic from your subnet or gateway is directed. A route table tells network packets which way they need to go to get to their destination.

4. NAT (Network Address Translation)

Network Address Translation devices, **launched in the public subnet, enables instances in a private subnet to connect to the Internet** but prevents the Internet from initiating connections with the instances.

5. VPC Endpoints

VPC Endpoints enable the **creation of a private connection** between VPC to supported AWS services and VPC endpoint services powered by Private Link using its private IP address. Endpoints do not require a public IP address, access over the Internet, NAT device, a VPN connection, or AWS Direct Connect.

IP Addresses

Instances launched in the VPC can have **Private, Public, and Elastic IP addresses**

❑ Private IP Addresses

- Private IP addresses are **not reachable over the Internet**, and can be used for communication only between the instances within the VPC
- Primary IP address is **associated with the network interface** for its lifetime, even when the instance is stopped and restarted and is released only when the instance is terminated

❑ Public IP address

- Public IP addresses **are reachable over the Internet**, and can be used for communication between instances and the Internet, or with other AWS services that have public endpoints
- Public IP address assignment to the Instance depends if the Public IP Addressing is enabled for the Subnet.
- Public IP address is **assigned from AWS pool of IP addresses** and it is not associated with the AWS account and hence is released when the instance is stopped and restarted or terminated.

❑ Elastic IP address

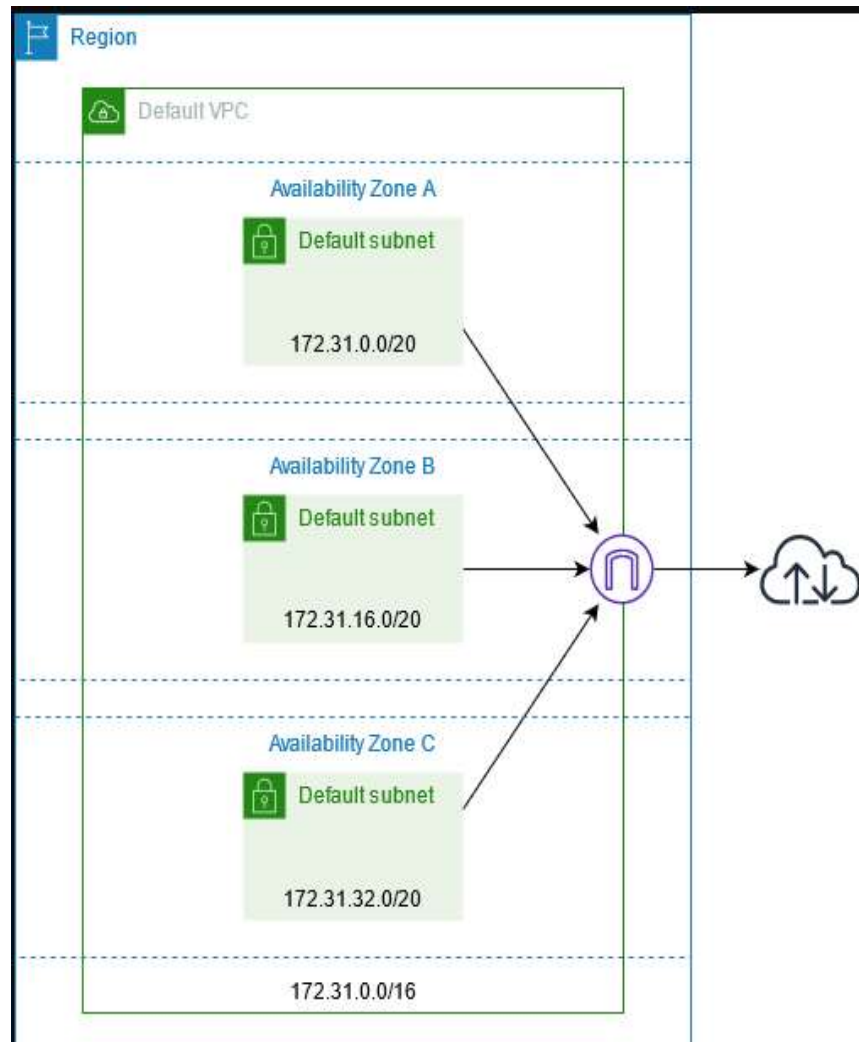
- Elastic IP addresses are **static, persistent public IP addresses that can be associated and disassociated with the instance, as required**
- Elastic IP address is allocated to the VPC and owned by the account unless released.
- A Network Interface can be assigned either a Public IP or an Elastic IP. If you assign an instance, that already has a Public IP, an Elastic IP, the public IP is released

Default VPCs

When you **start using Amazon VPC**, **you have a default VPC in each AWS Region**.

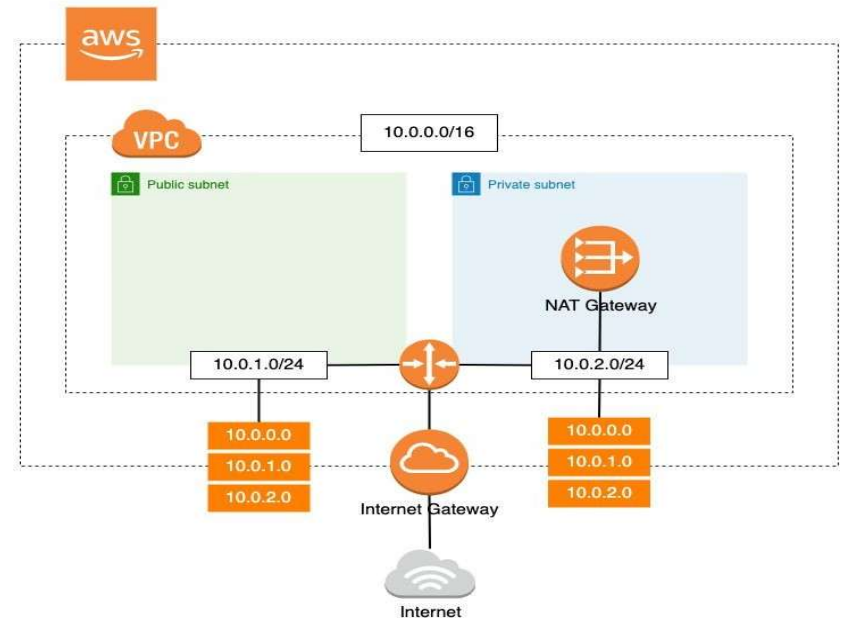
A default VPC comes with a **public subnet in each Availability Zone**, an internet gateway, and settings to enable DNS resolution. Therefore, you can immediately start launching Amazon EC2 instances into a default VPC. You can also use services such as Elastic Load Balancing, Amazon RDS, and Amazon EMR in your default VPC.

A default VPC is suitable for getting started quickly and for launching **public instances such as a blog or simple website**.



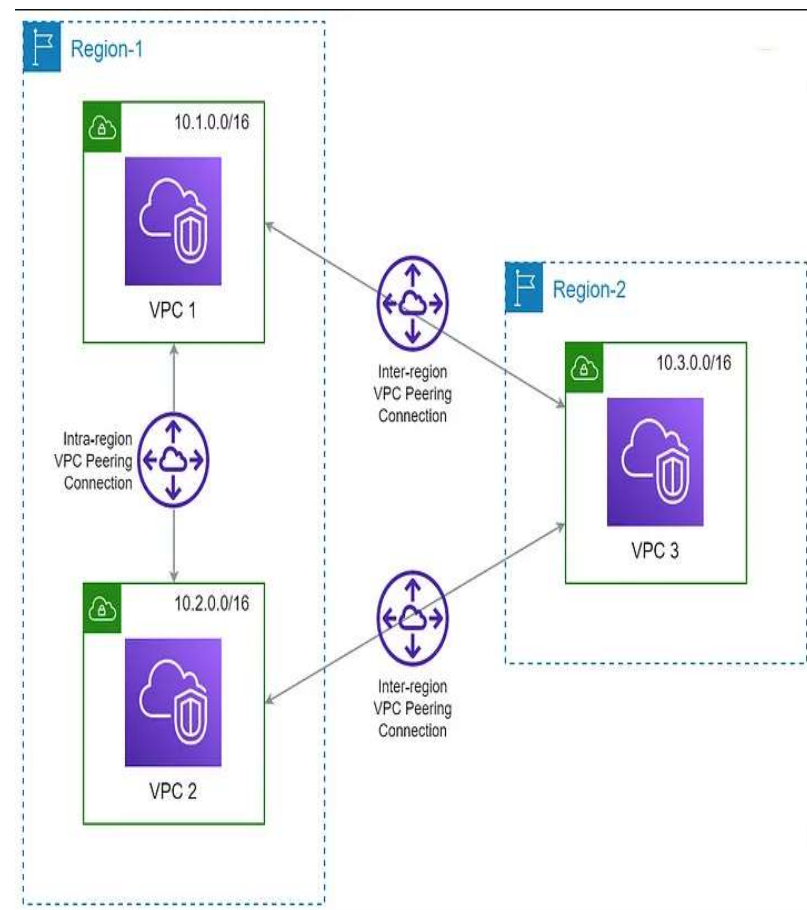
VPC with 2 Subnets (1 Public and 1 Private) in Single Zone

Each subnet must **reside entirely within one Availability Zone** and cannot span zones. By launching AWS resources in separate Availability Zones, you can protect your applications from the **failure of a single Availability Zone**.



VPC Peering

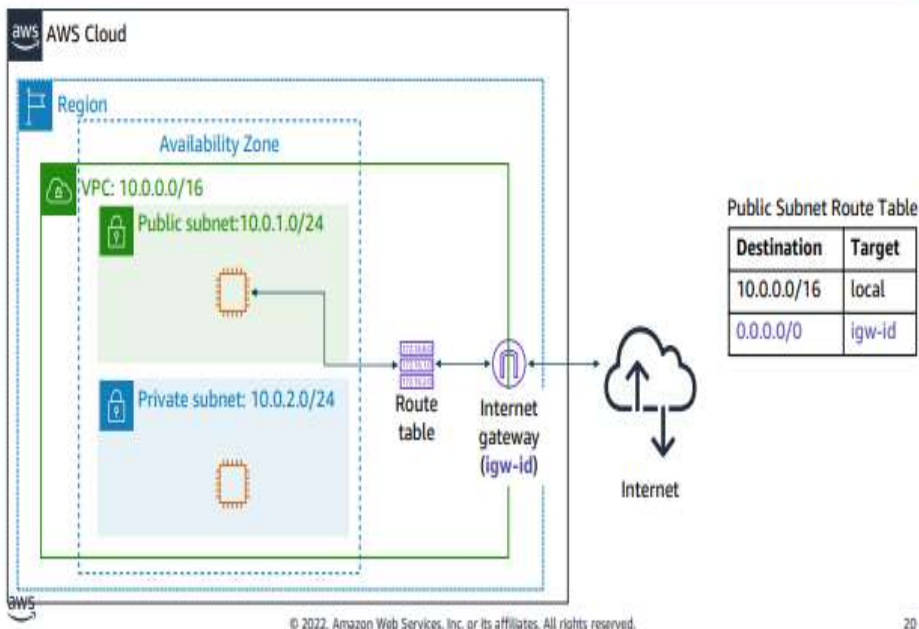
A VPC peering connection is a **networking connection between two VPCs** that enables you to route traffic between them using private IPv4 addresses or IPv6 addresses. Instances in either VPC can communicate with each other as if they are within the same network. You can create a VPC peering connection between your own VPCs, or with a VPC in another AWS account. The VPCs can be in different Regions (also known as an inter-Region VPC peering connection).



VPC networking

- Internet gateway
- Network address translation (NAT) gateway
- VPC sharing
- VPC peering
- AWS Site-to-Site VPN
- AWS Direct Connect
- VPC endpoints

- Internet gateway (igw)

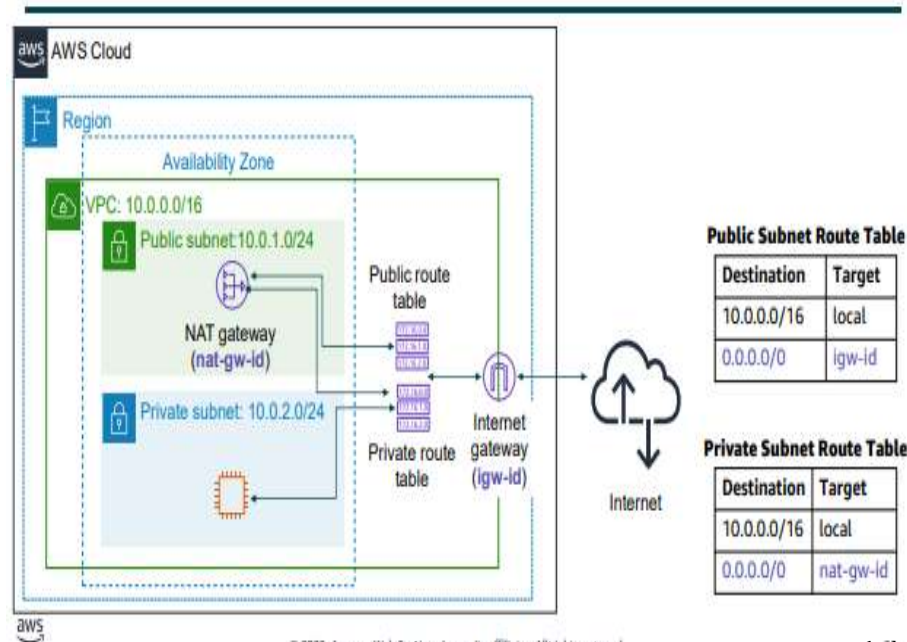


An **internet gateway** is a scalable, redundant, and highly available VPC component that allows **communication between instances in your VPC and the internet.**

An internet gateway serves **two purposes:**

- To provide a target in your VPC route tables for internet-routable traffic, and to perform network address translation for instances that were assigned public IPv4 addresses.
- To make a subnet public, you attach an internet gateway to your VPC and add a route to the route table to send non-local traffic through the internet gateway to the internet (0.0.0.0/0).

- **Network address translation (NAT) gateway**



A network address translation (NAT) gateway enables instances in a private subnet to connect to the internet or other AWS services, but prevents the internet from initiating a connection with those instances.

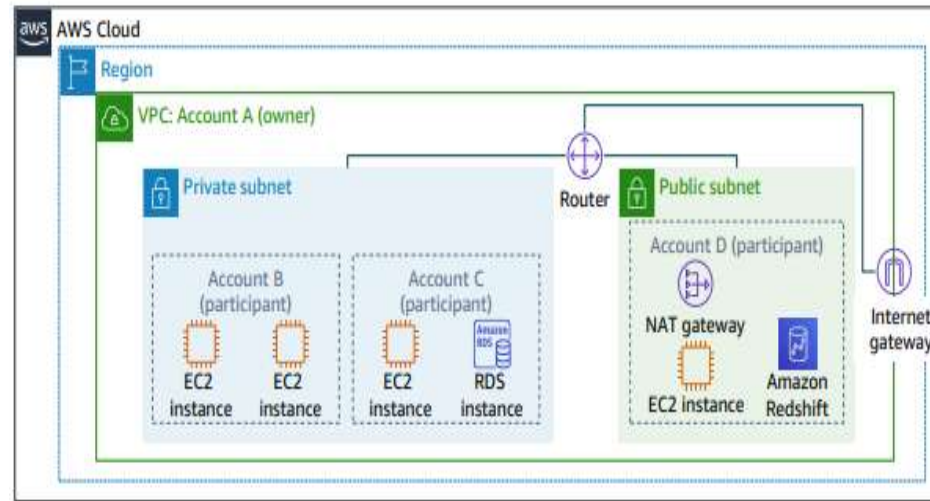
To create a NAT gateway, you must specify the public subnet in which the NAT gateway should reside. You must also **specify an Elastic IP address to associate with the NAT gateway** when you create it.

After you create a NAT gateway, you must **update the route table that is associated with one or more of your private subnets to point internet-bound traffic to the NAT gateway**. Thus, instances in your private subnets can communicate with the internet.

You can also use a **NAT instance** in a public subnet in your VPC instead of a NAT gateway. However, a NAT gateway is a managed NAT service that provides better availability, higher bandwidth, and less administrative effort.

For common use cases, AWS recommends that you use a NAT gateway instead of a NAT instance.

- **VPC sharing**

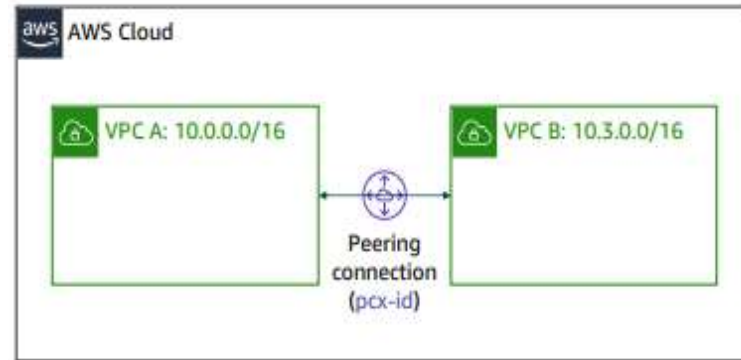


VPC sharing enables customers **to share subnets with other AWS accounts in the same organization** in AWS Organizations. VPC sharing enables multiple AWS accounts to create their application resources—such as Amazon EC2 instances, Amazon Relational Database Service (Amazon RDS) databases, Amazon Redshift clusters, and AWS Lambda functions—into shared, **centrally managed VPCs**. In this model, the account that owns the VPC (owner) shares one or more subnets with other accounts (participants) that belong to the same organization in AWS Organizations. After a subnet is shared, the participants can view, create, modify, and delete their application resources in the subnets that are shared with them. Participants cannot view, modify, or delete resources that belong to other participants or the VPC owner.

VPC sharing offers **several benefits**:

- **Separation of duties** – Centrally controlled VPC structure, routing, IP address allocation
- **Ownership** – Application owners continue to own resources, accounts, and security groups
- **Security groups** – VPC sharing participants can reference the security group IDs of each other
- **Efficiencies** – Higher density in subnets, efficient use of VPNs and AWS Direct Connect
- **No hard limits** – Hard limits can be avoided—for example, 50 virtual interfaces per AWS Direct Connect connection through simplified network architecture
- **Optimized costs** – Costs can be optimized through the reuse of NAT gateways, VPC interface endpoints, and intra-Availability Zone traffic

- **VPC peering**



Route Table for VPC A

| Destination | Target |
|-------------|--------|
| 10.0.0.0/16 | local |
| 10.3.0.0/16 | pcx-id |

Route Table for VPC B

| Destination | Target |
|-------------|--------|
| 10.3.0.0/16 | local |
| 10.0.0.0/16 | pcx-id |

You can connect VPCs in your own AWS account, between AWS accounts, or between AWS Regions.

Restrictions:

- IP spaces cannot overlap.
- Transitive peering is not supported.
- You can only have one peering resource between the same two VPCs.

A **VPC peering** connection is a networking connection between two VPCs that **enables you to route traffic between them privately**. Instances in either VPC can communicate with each other as if they are within the same network.

You can create a VPC peering **connection between your own VPCs, with a VPC in another AWS account, or with a VPC in a different AWS Region**.

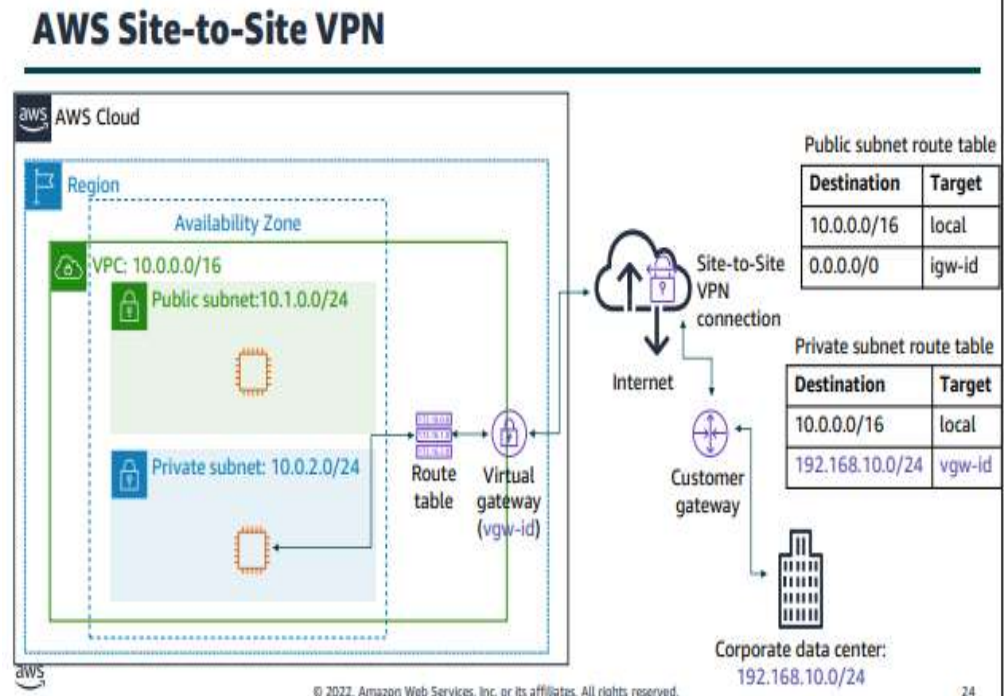
When you set up the peering connection, you **create rules in your route table** to allow the VPCs to communicate with each other through the peering resource.

For example, suppose that you have two VPCs. In the route table for VPC A, you set the destination to be the IP address of VPC B and the target to be the peering resource ID. In the route table for VPC B, you set the destination to be the IP address of VPC A and the target to be the peering resource ID.

VPC peering has some restrictions:

- IP address ranges cannot overlap.
- Transitive peering is not supported. For example, suppose that you have three VPCs: A, B, and C. VPC A is connected to VPC B, and VPC A is connected to VPC C. However, VPC B is not connected to VPC C implicitly. To connect VPC B to VPC C, you must explicitly establish that connectivity.
- You can only have one peering resource between the same two VPCs.

- **AWS Site-to-Site VPN**

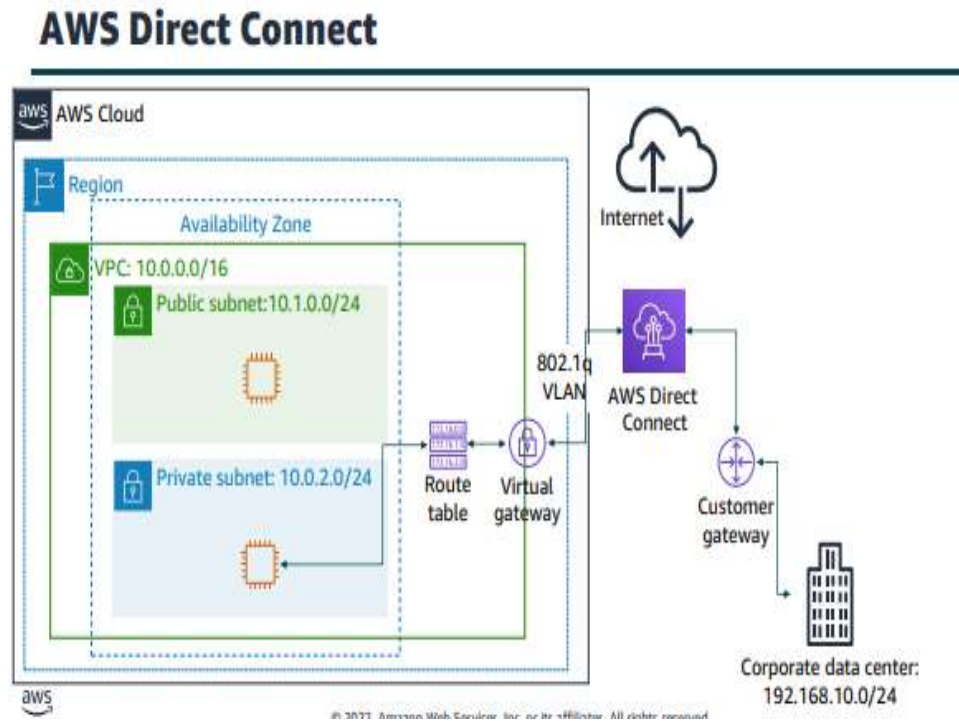


By default, instances that you launch into a VPC cannot communicate with a remote network.

To connect your **VPC to your remote network** (that is, create a virtual private network or VPN connection), you:

1. Create a new virtual gateway device (called a virtual private network (VPN) gateway) and attach it to your VPC.
2. Define the configuration of the VPN device or the customer gateway. The customer gateway is not a device but an AWS resource that provides information to AWS about your VPN device.
3. Create a **custom route table to point corporate data center-bound traffic to the VPN gateway**. You also must update security group rules.
4. Establish an **AWS Site-to-Site VPN (Site-to-Site VPN) connection** to link the two systems together.
5. Configure routing to pass traffic through the connection.

- **AWS Direct Connect**



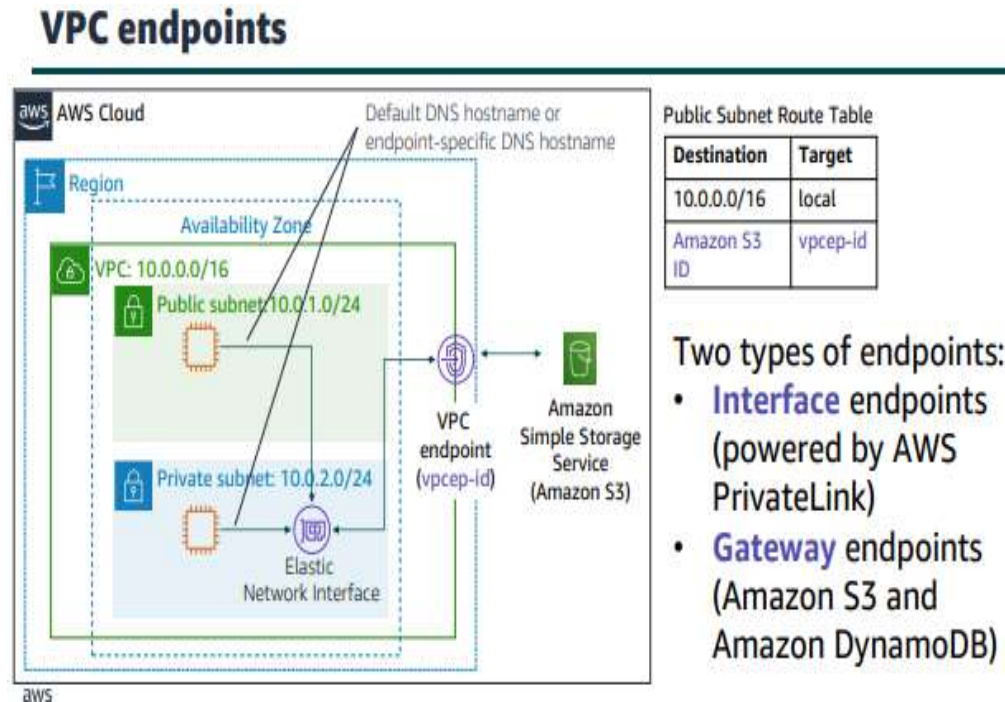
One of the challenges of network communication is **network performance**. Performance can be negatively affected if your data center is **located far away from your AWS Region**.

For such situations, AWS offers AWS Direct Connect, or DX.

AWS Direct Connect enables you to **establish a dedicated, private network connection between your network and one of the DX locations**.

This private connection can reduce your network costs, increase bandwidth throughput, and provide a more consistent network experience than internet-based connections. DX uses open standard 802.1q **virtual local area networks (VLANs)**

- **VPC endpoints**



A VPC endpoint is a virtual device that enables you to privately connect your VPC to supported AWS services and VPC endpoint services that are powered by AWS Private Link.

Connection to these services **does not require an internet gateway, NAT device, VPN connection, or AWS Direct Connect connection.** Instances in your VPC do not require public IP addresses to communicate with resources in the service. Traffic between your VPC and the other service does not leave the Amazon network.

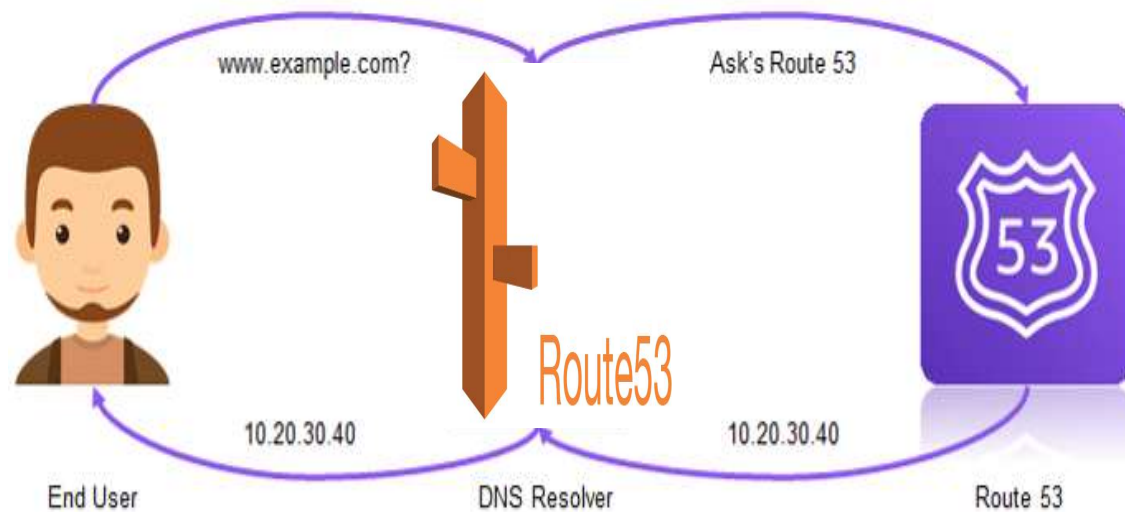
There are two types of VPC endpoints:

- An **interface VPC endpoint** (interface endpoint) enables you to connect to services that are powered by AWS PrivateLink. These services include some AWS services, services that are hosted by other AWS customers and AWS Partner Network (APN) Partners

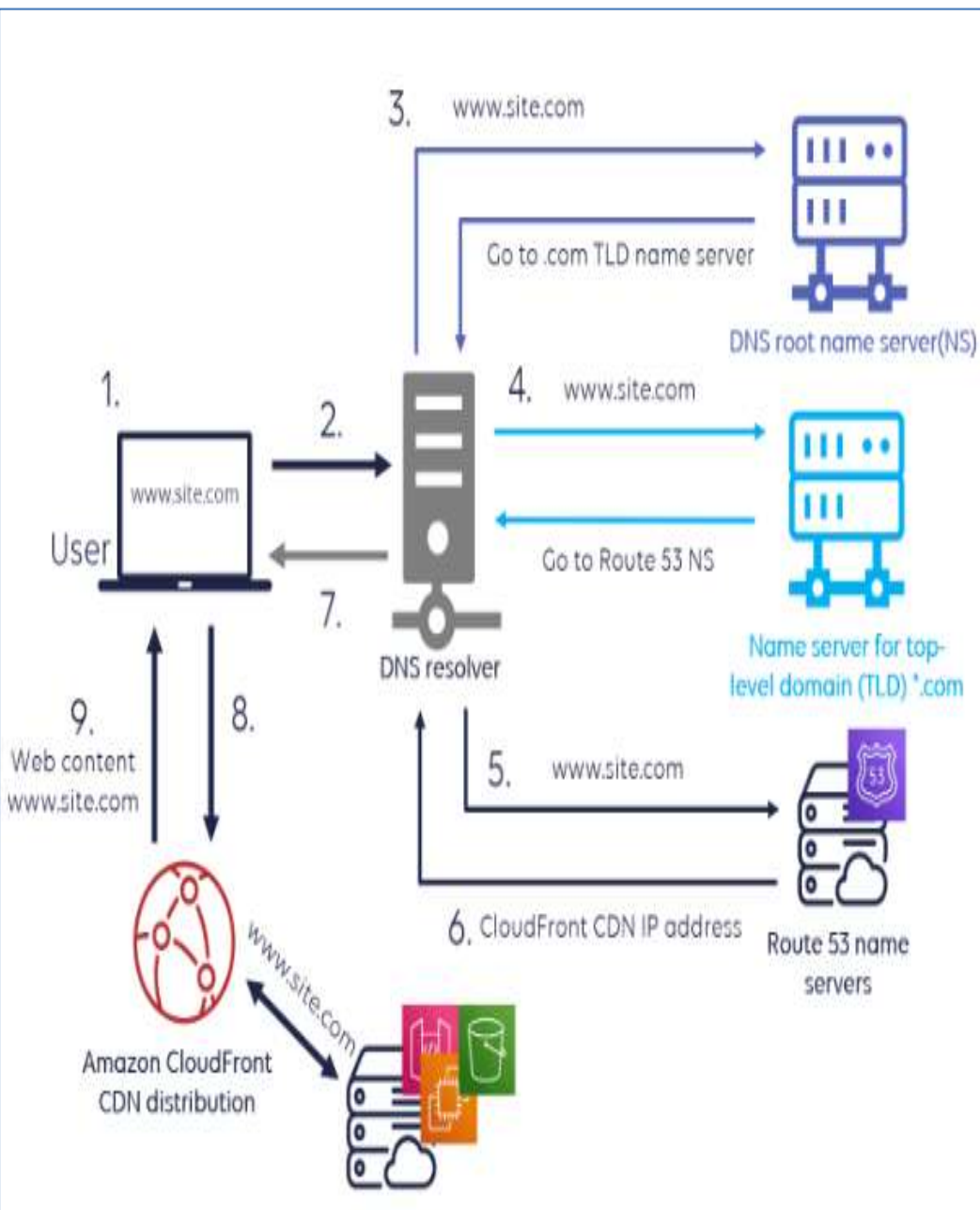
- **Gateway endpoints:** The use of gateway endpoints incurs no additional charge. Standard charges for data transfer and resource usage apply.

Amazon Route53

Amazon Route 53 is a **highly available and scalable cloud Domain Name System (DNS) web service**. It is basically designed for developers and corporate to route the end users to Internet applications by translating human-readable names like `www.geeksforgeeks.org` into the numeric IP addresses like `192.0.1.1` that computers use to connect to each other. You cannot use Amazon Route 53 to connect your on-premises network with AWS Cloud.



Route 53 work



1. A user opens a web browser and sends a request for www.site.com.

2. The request from `www.site.com` is routed to a DNS resolver, which is usually managed by the Internet Service Provider (ISP).

3. The ISP DNS resolver forwards the request from `www.site.com` to a DNS root name server.

4. The DNS resolver forwards the request from `www.site.com` again, this time to one of the top-level domain (TLD) name servers of `.com` domains. The `.com` domain name server responds with the names of the four Route 53 name servers associated with the `example.com` domain. The DNS resolver caches the four Route 53 name servers for future use.


5. The DNS resolver chooses a Route 53 name server and forwards the request from `www.site.com` to that Route 53 name server.
6. The Route 53 name server looks for the record `www.site.com` in the hosted zone `site.com`, gets its value, such as the alias of [Amazon CloudFront](#) distribution in the case of simple routing.
7. The DNS resolver finally has the right route (CloudFront IP) the user needs and returns the value for the user's web browser.
8. The web browser sends a request from `www.site.com` to the IP address of the CloudFront distribution.
9. The example CloudFront distribution returns the web page from cache or origin server for `www.site.com` to the web browser.


Different routing policies available in Route 53


Route 53 offers powerful policies to allow for efficient DNS requests. Once you've got your domain up and running, you can choose a routing policy that best fits your needs.


Function of each policy type.


Routing policy[Switch to quick create](#)


☒ **Simple routing**
Use if you want all of your clients to receive the same response(s).


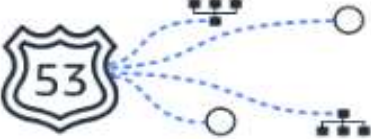
☐ **Weighted**
Use when you have multiple resources that do the same job, and you want to specify the proportion of traffic that goes to each resource. For example: two or more EC2 instances.


☐ **Geolocation**
Use when you want to route traffic based on the location of your users.


☐ **Latency**
Use when you have resources in multiple AWS Regions and you want to route traffic to the Region that provides the best latency.


☐ **Failover**
Use to route traffic to a resource when the resource is healthy, or to a different resource when the first resource is unhealthy.


☐ **Multivalue answer**
Use when you want Route 53 to respond to DNS queries with up to eight healthy records selected at random.


☐ **IP-based**
Use to route traffic to locations of IP address ranges in CIDR notation.


Different routing policies available in Route 53

1. Simple routing policy: Use for a single resource that performs a given function for your domain, for example, an Amazon EC2 instance that serves content for the example.com website.

2. Weighted: This allows you to assign weights to resource record sets. For instance, you can specify 25 for one resource and 75 for another, meaning that 25% of requests will go to the first resource and 75% will be routed to the second.

3. LBR (Latency based routing): Use when you have resources in multiple AWS Regions and you want to route end users to the AWS region that provides the lowest latency.

4. Failover: Use when you want to configure active-passive failover.

5. Geolocation: This lets you balance the load on your resources by directing requests to specific endpoints based on the geographic location from which the request originates.

6. Multivalue answer: Use when you want Route 53 to respond to DNS queries with up to eight healthy records selected at random.

7. IP-based: With IP-based routing, you can create a series of Classless Inter-Domain Routing (CIDR) blocks that represent the client IP network range and associate these CIDR blocks with locations.

Benefits of Route 53



5. Integrated routing policies

The routing of traffic based on different criteria such as latency, endpoint health and geographic location is advantageous. The flexibility of route 53 allows the configuration of multiple traffic policies and determines the activity of policies at a particular point in time.

6. Compatibility with other AWS services

Route 53 can help in mapping domain names to Amazon CloudFront distributions, Elastic Load Balancers, EC2 instances, S3 buckets, and other AWS resources.

1. High availability, reliability, and scalability

The distributed nature of our DNS servers helps ensure a **consistent ability to route your end users to your application**. Route 53 is designed to provide the level of dependability required by important applications and is backed by the [Amazon Route 53 SLA \(Service Level Agreement\)](#).

The close integration of AWS services allows users to perform changes to their architecture and scale resources to accommodate increasing Internet traffic volume without significant configuration and management requirements

2. Security

You can manage permissions for each user in your AWS account and **control who has access to which parts of the Route 53 service**. When you enable the Route 53 Resolver DNS firewall, you can configure it to check outbound DNS requests against a list of known malicious domains.

3. Global network

The **DNS database is replicated between regions**. This makes Route 53 a globally resilient service, meaning it can tolerate failure in one or more regions and continue to operate.

4. Cost-effective

You pay only for the resources you use, such as

Amazon CloudFront

Amazon CloudFront, which is a **content delivery network (CDN) service**. Content delivery occurs over networks, —for example, when you stream a movie from your favorite streaming service.

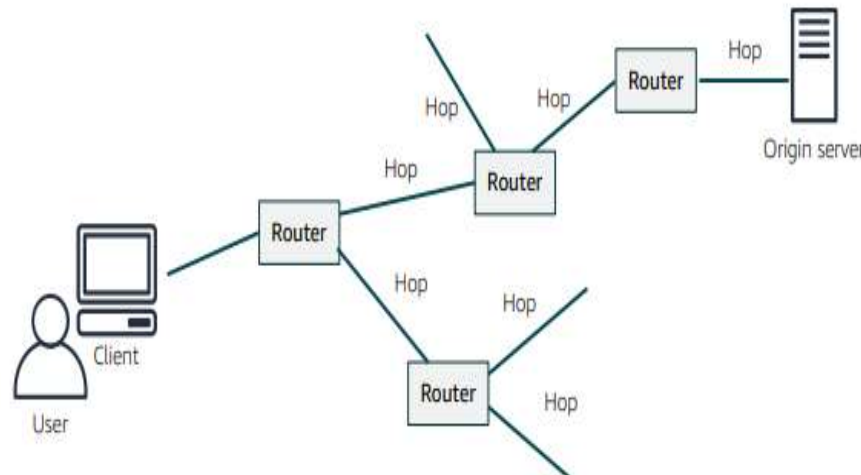
One of the **challenges of network communication is network performance.**

When you browse a website or stream a video, your request is routed through many different networks to reach an origin server. The origin server (or origin) stores the original, definitive versions of the objects (webpages, images, and media files). The **number of network hops and the distance that the request must travel** significantly affect the performance and responsiveness of the website.

Further, **network latency is different in various geographic locations.**

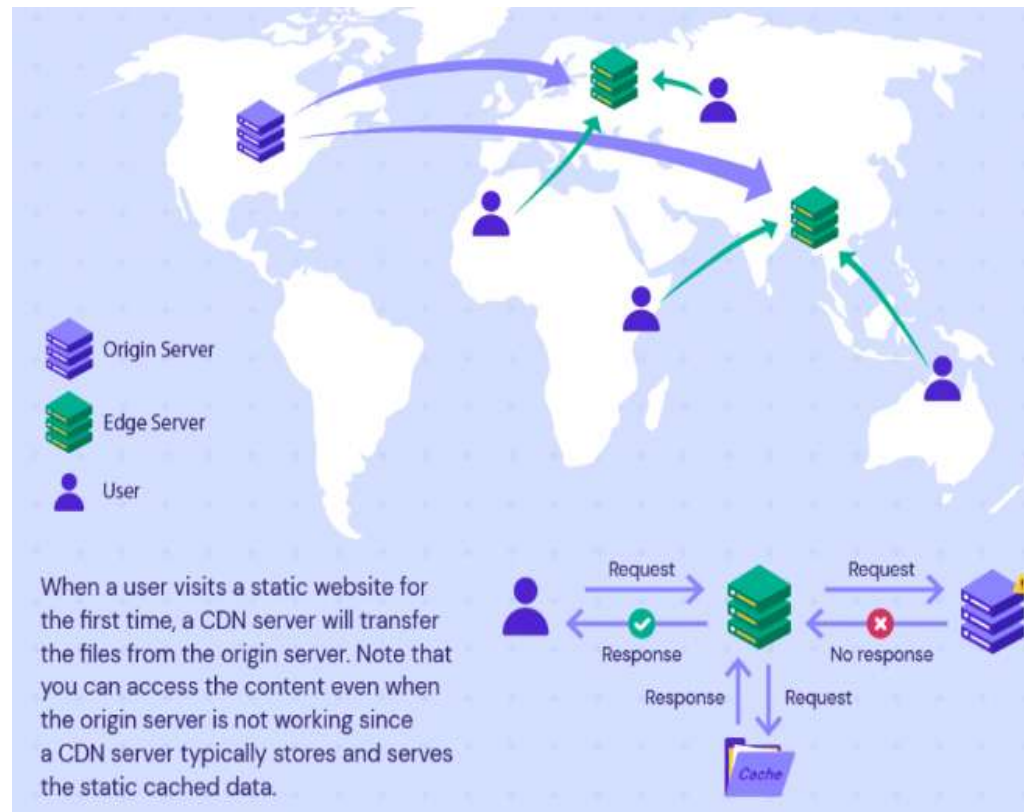
For these reasons, a **content delivery network might be the solution.**

Content delivery and network latency



Content delivery network (CDN)

- Is a globally distributed system of caching servers
- Caches copies of commonly requested files (static content such as Hypertext Markup Language, or HTML; Cascading Style Sheets, or CSS; JavaScript; and image files) that are hosted on the application origin server)
 - Delivers a local copy of the requested content from a nearby cache edge or Point of Presence that provides the fastest delivery to the requester.
 - Accelerates delivery of dynamic content that is unique to the requester and is not cacheable.
- Improves application performance and scaling
- The CDN establishes and maintains secure connections closer to the requester



Amazon CloudFront



- Fast, global, and secure CDN service : Amazon CloudFront is a fast CDN service that securely delivers data, videos, applications, and application programming interfaces (APIs) to customers globally with low latency and high transfer speeds
- Global network of edge locations and Regional edge caches:. Amazon CloudFront delivers files to users over a global network of edge locations and Regional edge caches.
- Amazon CloudFront is different from traditional content delivery solutions because it enables you to quickly obtain the benefits of high-performance content delivery without negotiated contracts, high prices, or minimum fees.
- Self-service model & Pay-as-you-go pricing: Like other AWS services, Amazon CloudFront is a self-service offering with pay-as-you-go pricing

Amazon CloudFront infrastructure

- Edge locations
- Multiple edge locations
- Regional edge caches

- **Edge locations** – Network of data centers that CloudFront uses to serve popular content quickly to customers.
- **Regional edge cache** – CloudFront location that caches content that is popular enough to stay at an edge location. It is located between the origin server and the global edge location.

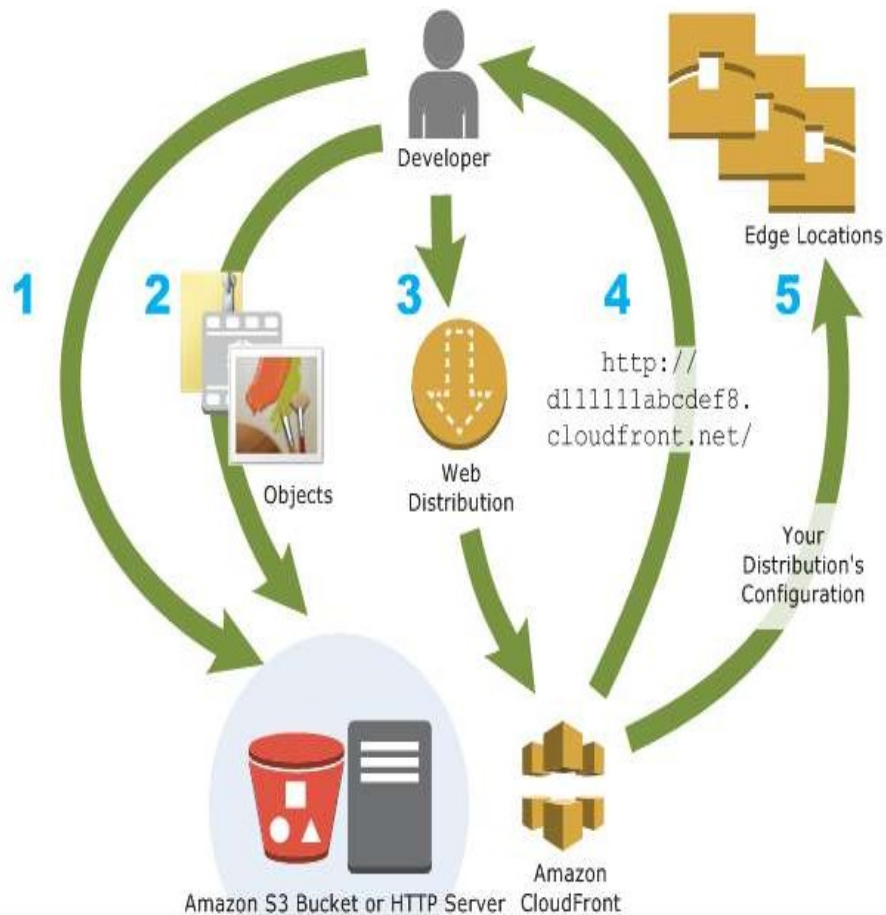


Amazon CloudFront delivers content through a worldwide **network of data centers that are called edge locations**.

When a user requests content that you serve with CloudFront, the user is routed to the **edge location that provides the lowest latency (or time delay)** so that content is delivered with the best possible performance.

CloudFront edge locations are designed to serve popular **content quickly to your viewers**.

Amazon CloudFront works



1. The first step is to decide on an origin server. Like all CDN providers, CloudFront requires you to define the server hosting the content you want CloudFront to deliver across the distributed network. The origin server can be an S3 bucket or an HTTP server (either based in Amazon's EC2 or locally in your own datacenter).

2. Next, you will upload your content to the origin server. Anything that can be **served over HTTP or a supported version of Adobe RTMP (Real-Time Messaging Protocol)** can be used. Typically, the content consists of web pages, images, and media files (video and audio).

3. The next step is the most important one. You will need to create a **distribution**. There are **two kinds of distribution** that you can create: **web distributions** for HTTP/HTTPS, and **RTMP Distributions** for RTMP and its variants. Distributions are the way you tell CloudFront what content to use and what to do with it.

4. If you want your content to be delivered over either HTTP or HTTPS, select a **Web** distribution, but if your deployment involves real-time data using RTMP protocols, then you should choose an **RTMP** distribution.

5. Finally, use the domain name endpoint

Amazon CloudFront benefits

- Fast and global
- Security at the edge
- Highly programmable
- Deeply integrated with AWS
- Cost-effective

Amazon CloudFront pricing

Data transfer out

- Charged for the volume of data transferred out from Amazon CloudFront edge location to the internet or to your origin.

HTTP(S) requests

- Charged for number of HTTP(S) requests.

Invalidation requests

- No additional charge for the first 1,000 paths that are requested for invalidation each month. Thereafter, \$0.005 per path that is requested for invalidation.

Dedicated IP custom SSL

- \$600 per month for each custom SSL certificate that is associated with one or more CloudFront distributions that use the Dedicated IP version of custom SSL certificate support.