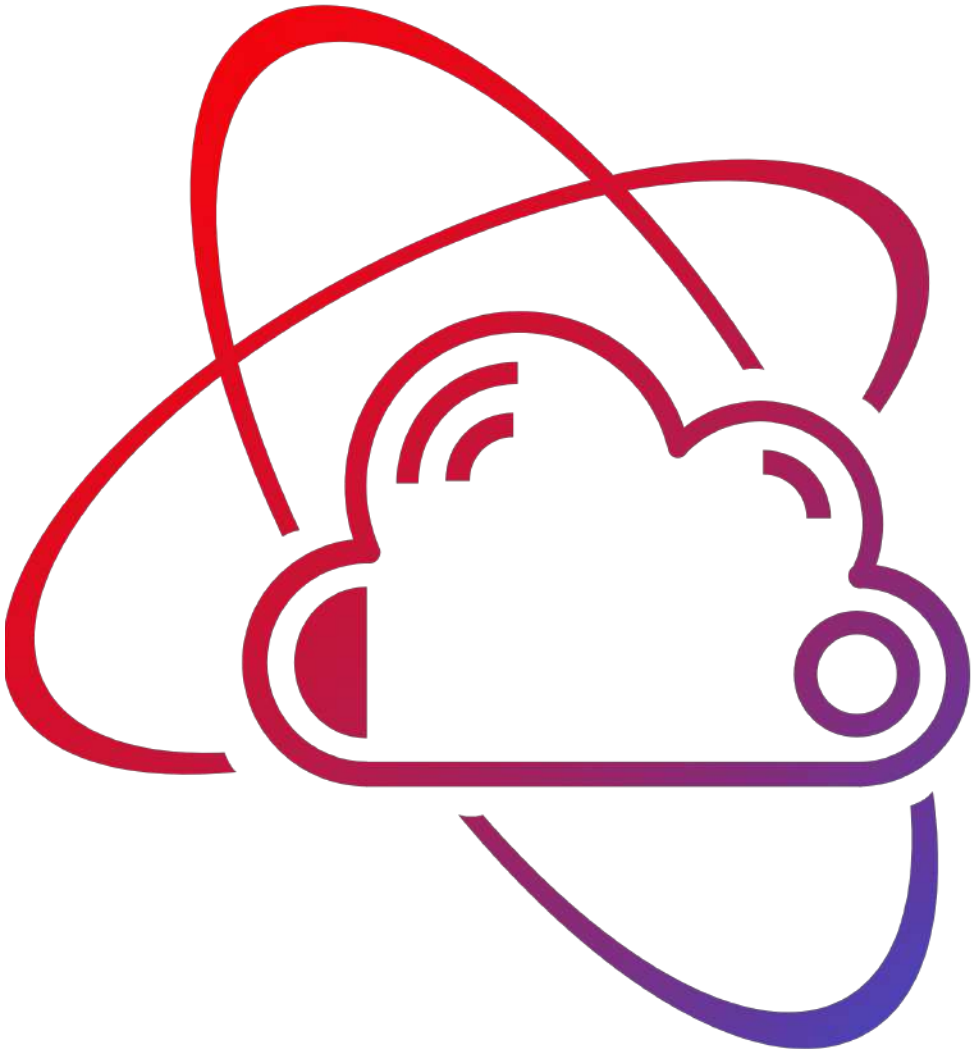
Cloud Computing Architecture

Week 3 - Introduction



SWINBURNE UNIVERSITY OF TECHNOLOGY

Acknowledgement of Country

We respectfully acknowledge the Wurundjeri People of the Kulin Nation, who are the Traditional Owners of the land on which Swinburne's Australian campuses are located in Melbourne's east and outer-east, and pay our respect to their Elders past, present and emerging.

We are honoured to recognise our connection to Wurundjeri Country, history, culture, and spirituality through these locations, and strive to ensure that we operate in a manner that respects and honours the Elders and Ancestors of these lands.

We also respectfully acknowledge Swinburne's Aboriginal and Torres Strait Islander staff, students, alumni, partners and visitors.

We also acknowledge and respect the Traditional Owners of lands across Australia, their Elders, Ancestors, cultures, and heritage, and recognise the continuing sovereignties of all Aboriginal and Torres Strait Islander Nations.

In this Presentation:

- AWS Networking Services Overview
- Introduction to Amazon VPC
- Assignment 1a Reminder (Due Week 4)

Image from: https://digitalcloud.training/aws-networking-services/

AWS Networking Services Overview

AWS Networking Services Overview

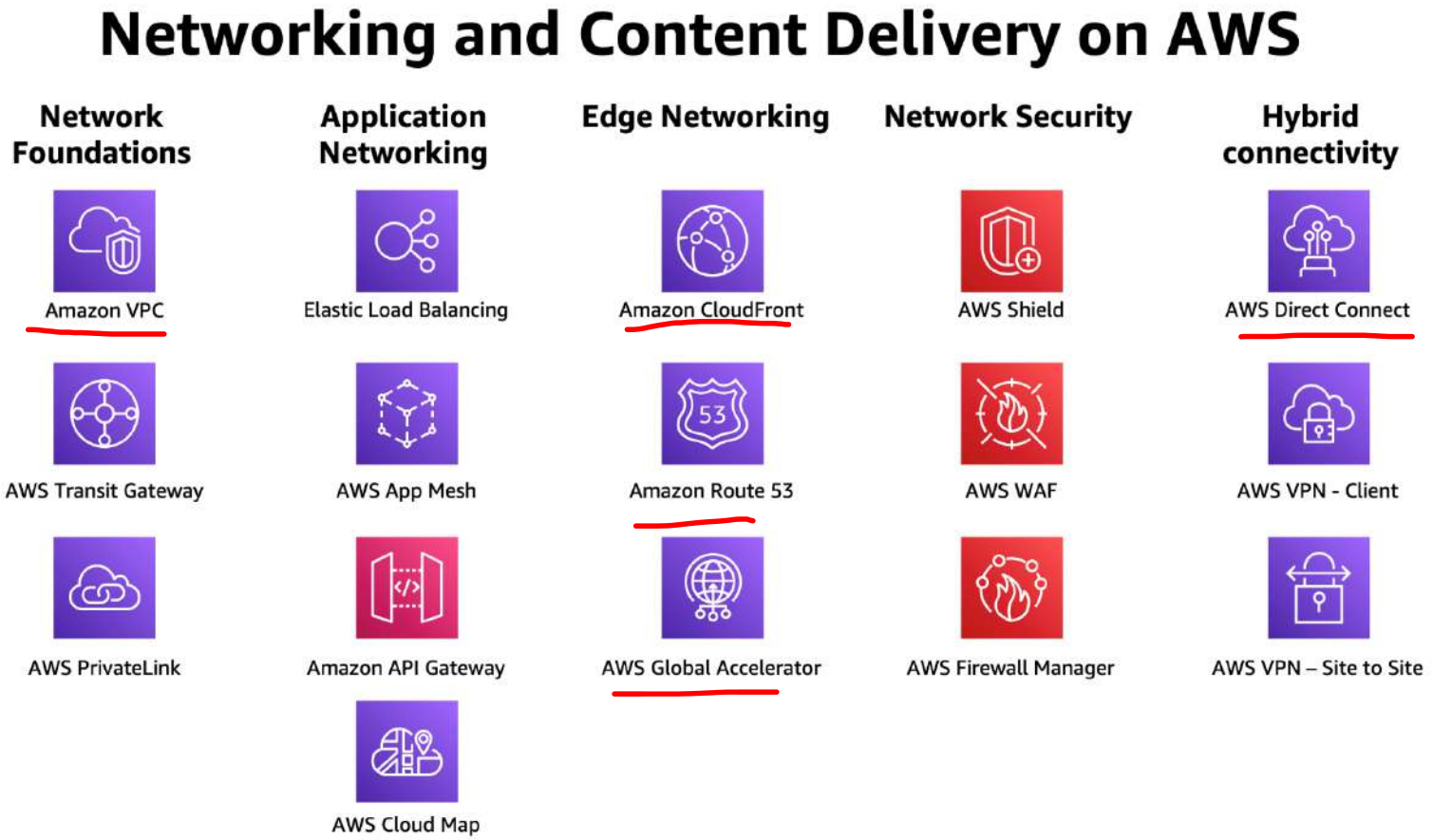
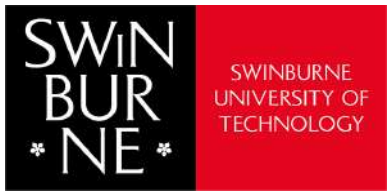


Image from: https://catalog.workshops.aws/general-immersionday/en-US/basic-modules/20-vpc

Introduction to Amazon VPC



Amazon VPC

Amazon VPC

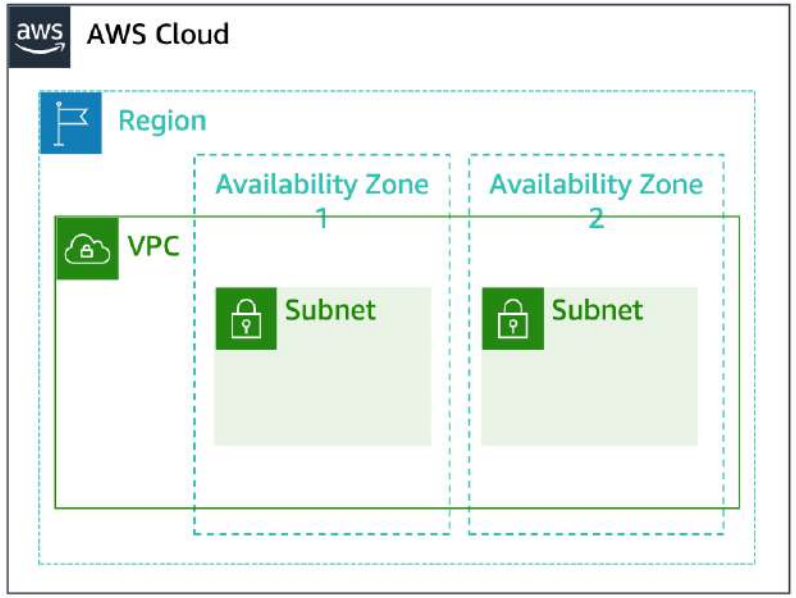
- Enables you to provision a logically isolated section of the AWS Cloud where you can launch AWS resources in a virtual network that you define
- Gives you control over your virtual networking resources, including:
  - Selection of IP address range
  - Creation of subnets
  - Configuration of route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

aws

11

VPCs and subnets

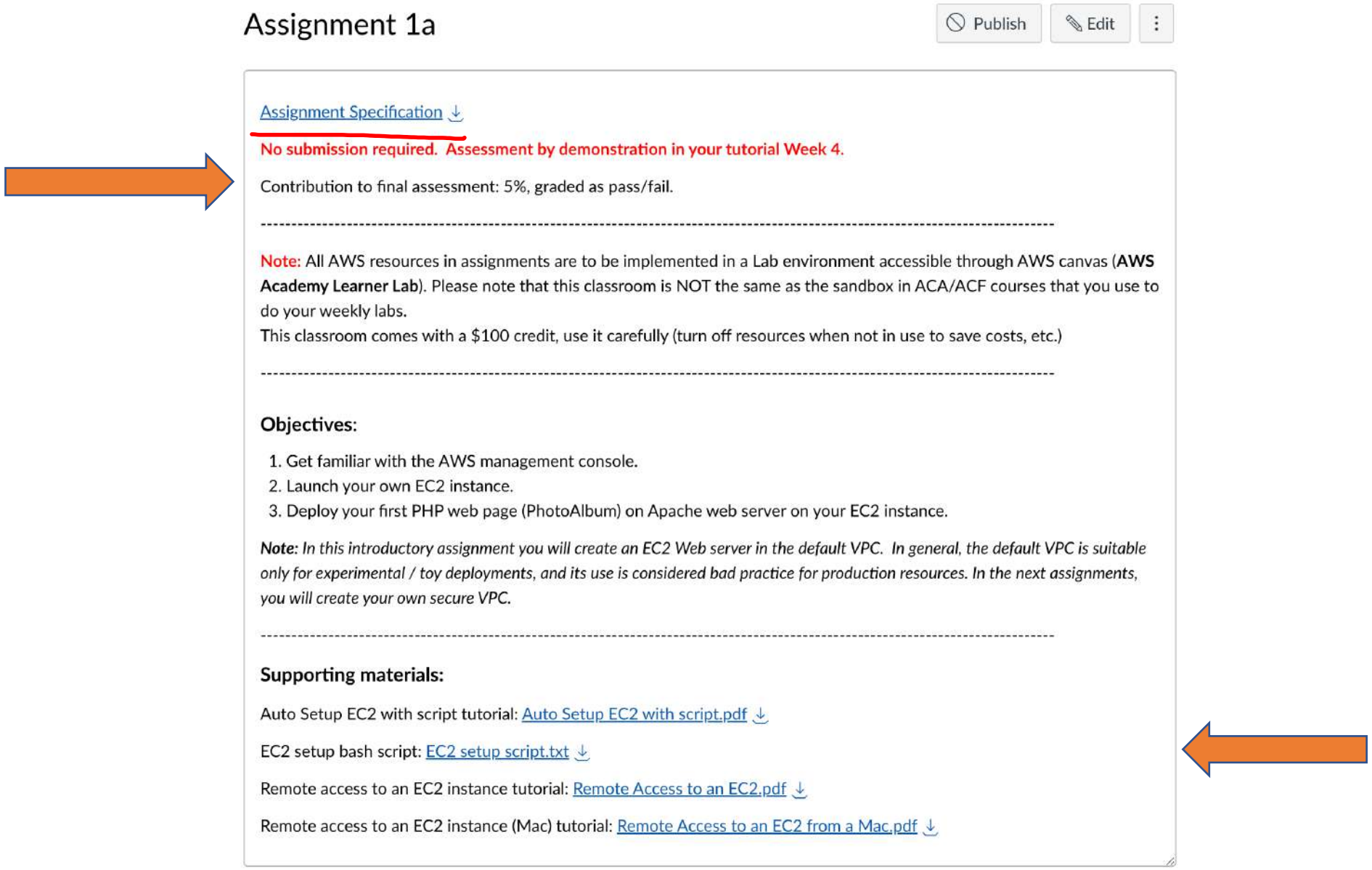
- VPCs:
  - Logically isolated from other VPCs
  - Dedicated to your AWS account
  - Belong to a single AWS Region and can span multiple Availability Zones
- Subnets:
  - Range of IP addresses that divide a VPC
  - Belong to a single Availability Zone
  - Classified as public or private



Assignment 1a (Due Week 4)

SWINBURNE UNIVERSITY OF TECHNOLOGY

Week 3 – Introduction: Introduction to Amazon VPC



Assignment 1a

Publish | Edit

Assignment Specification

No submission required. Assessment by demonstration in your tutorial Week 4.

Contribution to final assessment: 5%, graded as pass/fail.

Note: All AWS resources in assignments are to be implemented in a Lab environment accessible through AWS canvas (**AWS Academy Learner Lab**). Please note that this classroom is NOT the same as the sandbox in ACA/ACF courses that you use to do your weekly labs.
This classroom comes with a $100 credit, use it carefully (turn off resources when not in use to save costs, etc.)

Objectives:

1. Get familiar with the AWS management console.
2. Launch your own EC2 instance.
3. Deploy your first PHP web page (PhotoAlbum) on Apache web server on your EC2 instance.

Note: *In this introductory assignment you will create an EC2 Web server in the default VPC. In general, the default VPC is suitable only for experimental / toy deployments, and its use is considered bad practice for production resources. In the next assignments, you will create your own secure VPC.*

Supporting materials:

Auto Setup EC2 with script tutorial: Auto Setup EC2 with script.pdf

EC2 setup bash script: EC2 setup script.txt

Remote access to an EC2 instance tutorial: Remote Access to an EC2.pdf

Remote access to an EC2 instance (Mac) tutorial: Remote Access to an EC2 from a Mac.pdf

SWINBURNE UNIVERSITY OF TECHNOLOGY

COS80001
Cloud Computing Architecture

Lecture 03 Network Services and Environment Design

includes material from
ACF Module 2.3 – Virtual Private Cloud
ACA Module 2 – Designing your Environment

Reminder

- Assignment 1a due next week (demo to your tutor in your lab)
- Pass/Fail mark.

SWINBURNE UNIVERSITY OF TECHNOLOGY

Last week

- **Virtualisation of Computation**
  - ☐ Virtual machines
  - ☐ Containers
  - ☐ Serverless Computing
- **AWS Compute services (ACF Module 6)**
  - ☐ 1: Compute Services Overview
  - ☐ 2: Introduction to Amazon Elastic Compute Cloud (Amazon EC2)
  - ☐ 3: EC2 Cost Optimization
  - ☐ 4. Container services
  - ☐ 5: Introduction to AWS Lambda
  - ☐ ~~6: Introduction to AWS Elastic Beanstalk~~

SWINBURNE UNIVERSITY OF TECHNOLOGY

Quiz…

This week – Network Design (inside the VPC)

- Choosing a Region and Selecting Availability Zones
- Creating a Virtual Private Cloud (VPC) and Subnets
  - VPC components and network address - CIDR
  - Private and Public Subnets
  - Default VPCs and Default Subnets
- Controlling VPC Traffic
  - Route tables, Security groups, Network ACLs, Internet gateways, NATs, Bastion Hosts
- Multiple VPCs and AWS Accounts

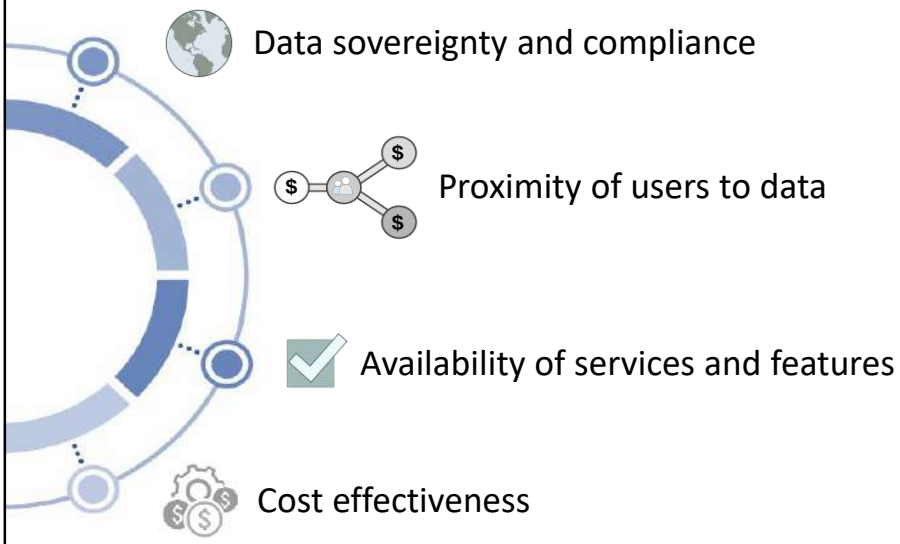
Extra Notes: Integrating On-Premises Components

SWINBURNE UNIVERSITY OF TECHNOLOGY

4

Introducing Part 1: How to Choose a Region.

How to Choose a Region?



Let's discover how to choose a Region.

Choosing a region involves four main considerations:

- Data sovereignty and compliance
- Proximity of users to the data
- Service and feature availability
- And cost effectiveness

Let's take a look at each of these in greater detail.

Data Sovereignty and Compliance

Where can you legally host your infrastructure?

What are the national and local data security laws?

Is customer data allowed outside of the country?

Can you meet governance requirements?

Did you know?
AWS opened its first carbon-neutral Region in 2011 and now offers five!

Learn more.

Your first consideration is understanding where you can legally host your infrastructure. What are the national and local data security laws? Keep in mind that your data will be subject to the laws of the country and locality where it's stored.
Is customer data allowed outside of the country? Some laws dictate that if you're operating your business in their jurisdiction, you can't store that data anywhere else.

Can you meet governance requirements? Compliance standards—such as the United States' Health Insurance Portability and Accountability Act, or HIPAA—have strict guidelines on how and where data can be stored. AWS Architects take all of these considerations into account when they evaluate where to host infrastructures.

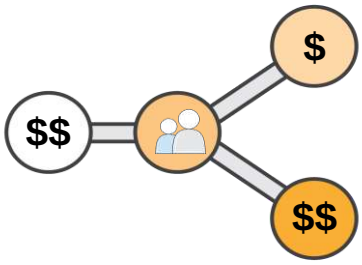
Did you know that AWS opened its first carbon-neutral Region in 2011, and we now offer five separate carbon-neutral Regions?

For more information on carbon-neutral options, select the link.
https://aws.amazon.com/about-aws/sustainability

Proximity of Users to Data

What is the proximity to your user base?

Study: 100-ms delays can cost 1% in sales on Amazon.com



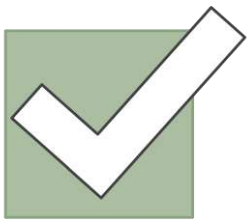
Equidistant regions? Compare costs

A second consideration to look for is proximity to your user base. Proximity is a big factor in choosing your Region, especially when latency is critical. In most cases, the latency difference between using the closest Region and the farthest Region is relatively small, but even small differences in latency can impact a customer experience.

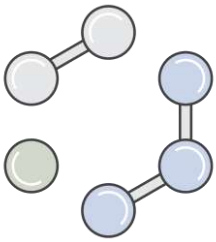
An internal study in 2006 found that every 100 millisecond delay on Amazon.com corresponded to a 1 percent drop in sales. Customers expect responsive environments, and as time goes by and technology becomes more powerful, customers' expectations rise as well.
If you have two Regions that are equidistant from one another, compare the costs. All AWS services are priced per Region, which means some Regions are more expensive than others.

Availability of Services and Features

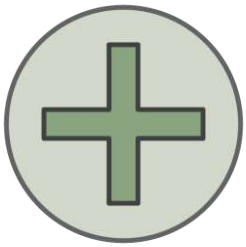
What services and features are available?



Some services available in limited regions



Some services can cross-regions, but at increased latency



Services expanded to new regions regularly

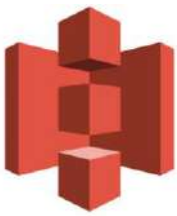
The next consideration should be understanding which services and features are available. Some services are only available in limited regions. While we strive to make all of our services and features available everywhere, the complications that arise from having a global reach make it challenging to accomplish that goal.

Some services can cross Regions, but have increased latency. Services are expanded to new Regions regularly. Rather than waiting until a service is available everywhere before launching it, we release our service when it's ready, and expand its availability as soon as possible. If you're interested in using a service that's not available in your Region, you can still explore the possibilities of first going to market with a Region where the service is available. Doing so allows you to embrace a faster "go to market" strategy.

Cost-effectiveness

Consider cost-effectiveness.

Service costs vary by region.



Some services (i.e. Amazon S3) have costs for transferring data out.

Consider replicating entire environment to another region.

Finally, you want to consider cost effectiveness because service costs vary by Region. For example, the cost of running an Amazon Elastic Compute Cloud, or Amazon EC2, instance in the US-East 1 Region might not be the same if you ran it in the EU-West 1 Region. Typically, the difference in cost might not be enough to supersede the other three considerations. However, in cases where the differences in latency, compliance, or service availability between Regions are minimal, you might be able to save money by using the lower-cost Region for your environment.

Some services—such as Amazon Simple Storage Service, or Amazon S3—have costs for transferring data out. If you have Amazon EC2 instances, make sure that they're in the same Region because you start incurring costs when you transfer data outside of a Region. Keep this in mind when you decide where to place your infrastructure and host your data. The best case for an infrastructure is to use at least two Regions, so if one Region goes down because of a catastrophic event, your infrastructure can still serve your customers. Most applications support this type of setup.

In circumstances where your customers are in different areas of the globe, you might consider optimizing the customer experience by replicating your entire environment to multiple Regions that are closer to your customers. Because you would then be distributing your load across multiple environments, your costs for the components in each environment might go down even as you add more infrastructure. For example, adding a second application environment might allow you to cut your processing and storage capacity requirements in half for

environment down as a way to mitigate the cost of adding another environment.

The downside to that approach is that you now have two environments to manage, and not all of your components will scale down enough to mitigate all of the new component costs. Additionally, you might have to maintain one single storage "source of truth" in one Region—such as a master Amazon Relational Database Service, or Amazon RDS, instance. Your secondary Region would have to communicate with the primary Region, which can increase latency and cost for those operations.

How Many Availability Zones Should You Use?

Once you have determined where you want to host your data, consider how many Availability Zones you should use.

How Many Availability Zones Should You Use?

Recommendation: Start with two Availability Zones per Region.

- **Best practice:** If resources in one Availability Zone are unreachable, your application shouldn't fail.
- Most applications can support two Availability Zones.

Something to consider:
For heavy usage (Amazon DynamoDB) it may be beneficial to use more than two Availability Zones.

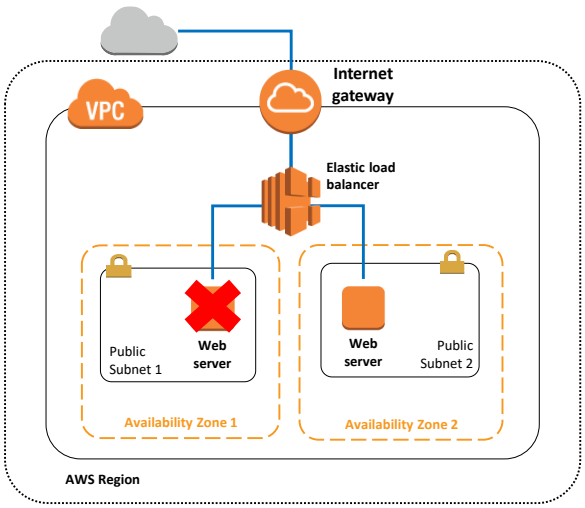
The best recommendation is to start with two Availability Zones per Region.

If resources in one Availability Zone are unreachable, your application shouldn’t fail.

Most applications can be designed to support two Availability Zones, but may not benefit from more, because they use data sources that only support primary and secondary failures.

For heavy Amazon EC2 Spot instance usage or data sources that go beyond active or passive, such as Amazon DynamoDB, there might be a benefit to using more than two Availability Zones. However, using more than two Availability Zones isn’t usually cost-effective.

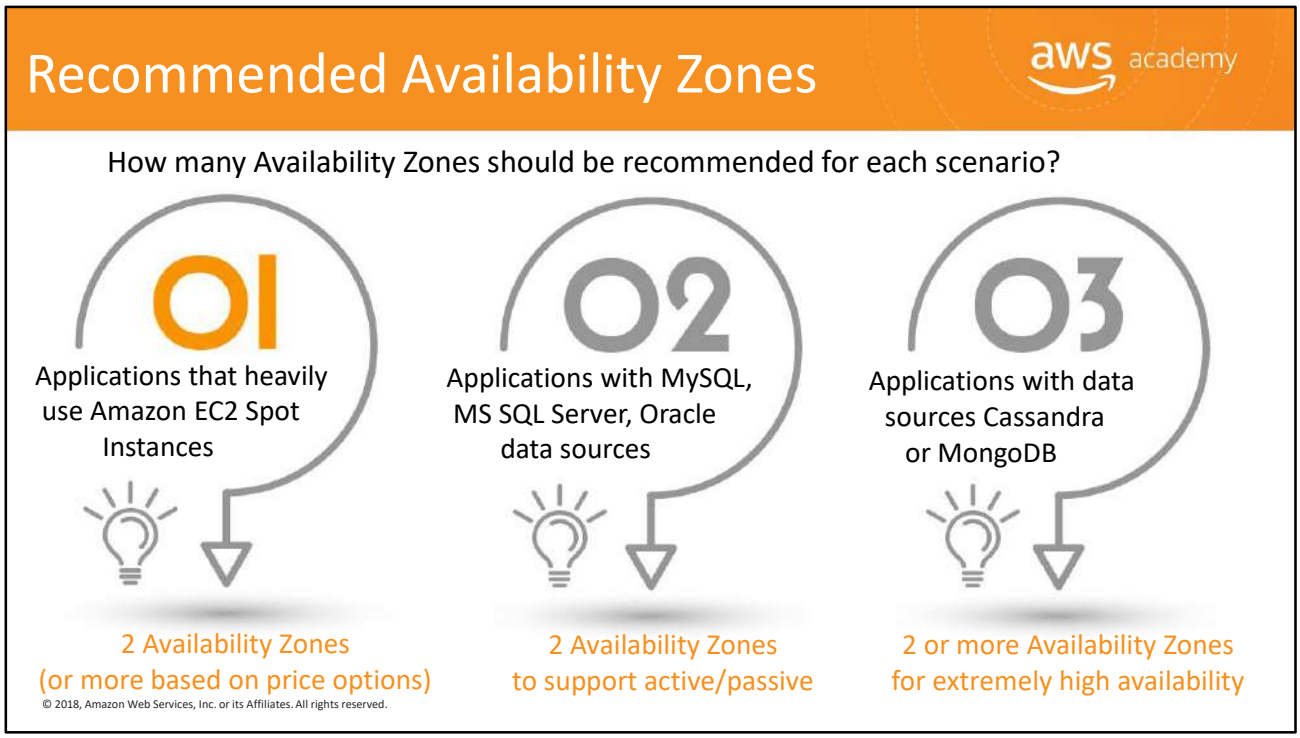
Using Two Availability Zones



This chart shows an example architecture that uses two Availability Zones, an elastic load balancer that distributes traffic between them, and an internet gateway. If one of the web servers becomes unavailable, the load balancer recognizes this change and stops distributing traffic to the unhealthy instance.

This architecture ensures that if there's a problem in one of the Availability Zones where a component resides, your application is still available. Customers shouldn’t experience any differences if a failure occurs.

Recommended Availability Zones



Let's review a few scenarios. How many Availability Zones should be recommended for each scenario?

For applications that heavily use Amazon EC2 Spot Instances, two Availability Zones should be recommended, or you can use more for additional price options. Because Amazon EC2 Spot instances are priced according to Availability Zone, you could use two Availability Zones to get the best price, even when prices change.

For applications that have data sources such as MySQL, Microsoft SQL Server, and Oracle, two Availability Zones should be used to support both active and passive.

For applications with data sources such as Cassandra or MongoDB, two or more Availability Zones should be used for extremely high availability.

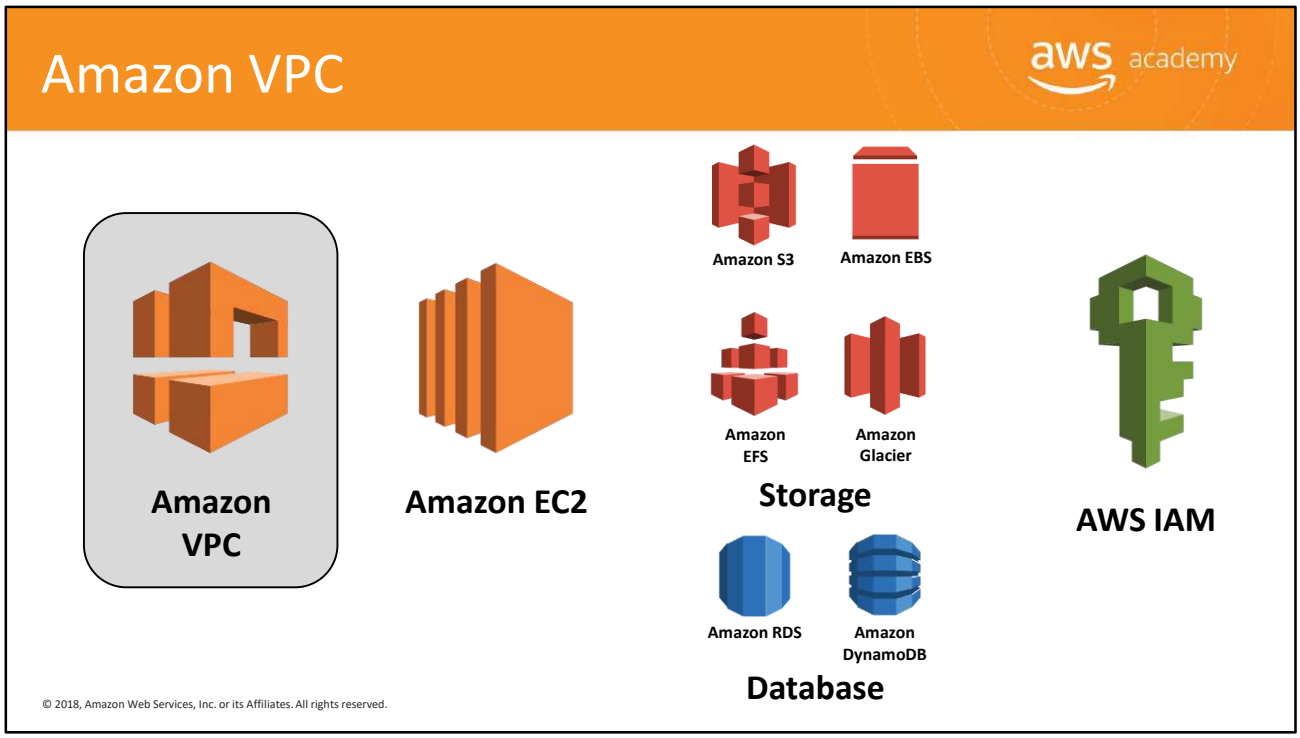
This week – Network Design

- Choosing a Region and Selecting Availability Zones
- Creating a Virtual Private Cloud (VPC) and Subnets
  - VPC components and network address - CIDR
  - Private and Public Subnets
  - Default VPCs and Default Subnets
- Controlling VPC Traffic
  - Route tables, Security groups, Network ACLs, Internet gateways, NATs, Bastion Hosts
- Multiple VPCs and AWS Accounts

SWINBURNE UNIVERSITY OF TECHNOLOGY

Virtual Private Cloud (VPC)

A *virtual private cloud* (VPC) is a virtual network dedicated to your AWS account. It is logically isolated from other virtual networks in the AWS Cloud. You can launch your AWS resources, such as Amazon EC2 instances, into your VPC. You can specify an IP address range for the VPC, add subnets, associate security groups, and configure route tables. Now, let's consider if you should fit everything into one Virtual Private Cloud.



The AWS cloud offers pay-as-you-go, on-demand compute as well as managed services, all accessible via the web. These compute resources and services must be accessible via normal IP protocols implemented with familiar network structures. Customers must adhere to networking best practices, as well as meet regulatory and organizational requirements. Amazon VPC is the AWS service that will meet your networking requirements and enable you to build your own virtual private network in AWS.

Let’s dive a little deeper into Amazon VPC.

Amazon VPC

Amazon Virtual Private Cloud (Amazon VPC) allows you to provision **virtual networks** hosted on the AWS cloud and dedicated to your AWS account.

- A private, virtual network in the AWS Cloud, Amazon VPCs are **logically isolated** from other virtual networks.
- Many AWS resources, such as Amazon Elastic Compute Cloud (Amazon EC2) instances, are launched into Amazon VPCs.
- Allows complete control of network configuration, including:
  - Internet Protocol (IP) address ranges
  - Subnet creation
  - Route table creation
  - Network gateways
  - Security settings

Amazon VPC is your network environment in the cloud. It allows you to create a private network within the AWS cloud that uses many of the same concepts and constructs as an on-premises network, but as we shall see later, much of the complexity of setting up a network has been abstracted without sacrificing control, security, and usability.

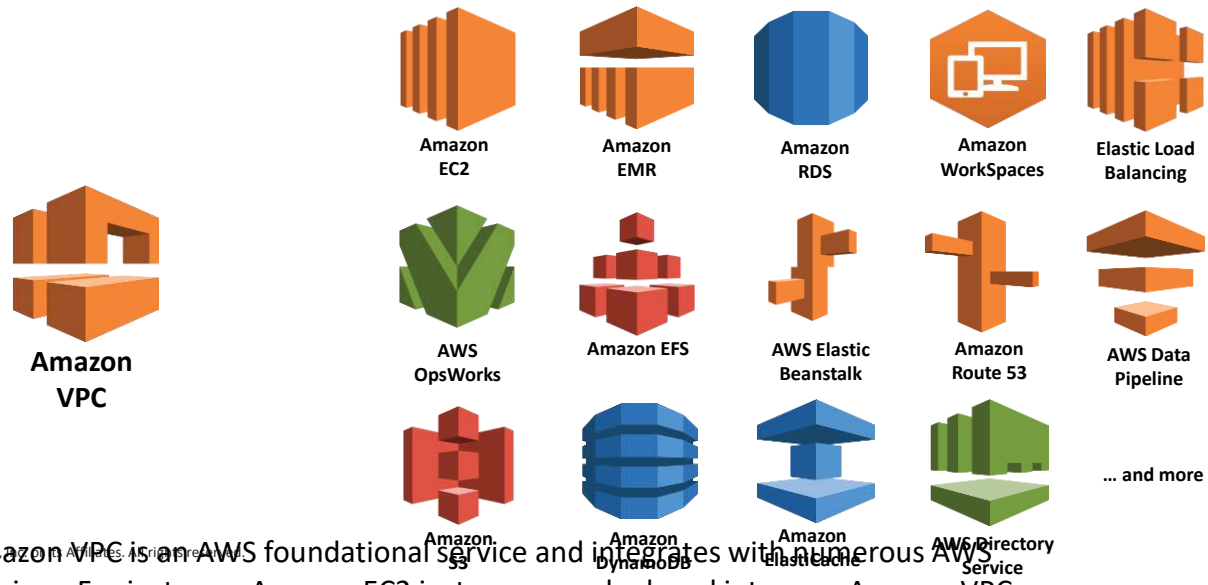
Amazon VPC is where you will launch many of your resources, and it's designed to provide greater control over the isolation of your environments and their resources from each other. Within a region, you can create multiple Amazon VPCs, and each Amazon VPC is logically isolated even if it shares its Internet Protocol (IP) address space.

Amazon VPC also gives you complete control of the network configuration. Customers can define normal networking configuration items such as IP address ranges, subnet creation, route table creation, network gateways, and security settings. This allows you to control what you expose to the Internet and what you isolate within the Amazon VPC.

Amazon VPC Integration

Other AWS services deploy into Amazon VPC:

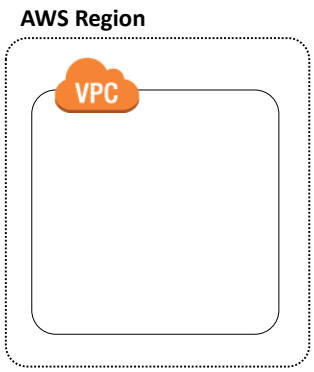
Service inherits security build into network.



Amazon VPC is an AWS foundational service and integrates with numerous AWS services. For instance, Amazon EC2 instances are deployed into your Amazon VPC. Similarly, **Amazon Relational Database Service (Amazon RDS)** database instances deploy into your Amazon VPC, where the database is protected by the structure of the network just like your on-premises network. Understanding and implementing Amazon VPC will allow you to fully use other AWS services.

Amazon VPC Features

- Builds upon high availability of AWS Regions and Availability Zones (AZ):
  - Each Amazon VPC lives in a single region
  - Multiple Amazon VPCs per account
- Subnets:
  - Used to divide Amazon VPC
  - Allow Amazon VPC to span multiple AZs



Let's take a look at the features of Amazon VPC. Amazon VPC builds upon the AWS global infrastructure of Regions and Availability Zones (AZ), and allows you to easily take advantage of the high availability provided by the AWS cloud. It also allows you to provision virtual networks hosted on the AWS cloud and dedicated to your AWS account. Amazon VPCs live within regions, as they can exist only in a single region.

There are ways to connect Amazon VPCs in different regions to each other without going through the public Internet. Each AWS account can create multiple Amazon VPCs that can be used to segregate environments.

An Amazon VPC defines an IP address space that is then divided by subnets. These subnets are deployed within Availability Zones causing the Amazon VPC to span AZs. Amazon VPCs are logically isolated from other virtual networks. You can create many subnets in a Amazon VPC, though fewer is recommended to limit the complexity of the network topology, but this is totally up to you. You can configure route tables for your subnets to control the traffic between subnets and the Internet. By default, all subnets within a Amazon VPC can communicate with each other. It should be noted that while a Amazon VPC can span across multiple AZs, a subnet cannot.

Subnets are generally classified as public or private, with **public** having direct access to the Internet and **private** not having direct access to the Internet. For a subnet to be

public, we need to attach an Internet gateway to the Amazon VPC and update the route table of the public subnet to send non-local traffic to the Internet gateway. Amazon EC2 instances also need a public IP address to route to an Internet gateway.

Amazon VPC Components

- **Subnets:** Segment of an Amazon VPC's IP address range where you can launch AWS services.
  - Subnets within a zone cannot span zones → **one** subnet equal **one** availability zone.
  - Can be classified as public, private, or VPN only.
  - Default Amazon VPCs contain one public subnet in every Availability Zone within the region with a netmask of /20.
- **Route Tables:** Used to control traffic going out of the subnets.
- **Security Groups:** A virtual, stateful firewall.
- **Network Access Control Lists (ACLs):** Control access to subnets; and stateless.

You can use the following components to configure networking in your Amazon VPC:

- A **subnet** is a segment of an Amazon IPC address range where you can launch AWS services:
  - CIDR blocks define subnets.
  - AWS reserves the first four IP addresses and the last IP address of every subnet for internal networking purposes.
  - A public subnet is one in which an associated route table direct the subnet's traffic to the Amazon VPC's internet gateway. A private subnet is one in which the associated route table does not direct the subnet's traffic to the internet gateway. A VPN only subnet only directs traffic to the Amazon VPC's virtual private gateway.

- A **route table** contains a set of rules, called **routes**, that are used to determine where network traffic is directed. Each subnet in your Amazon VPC must be associated with a route table; the table controls the routing for the subnet. A subnet can only be associated with one route table at a time, but you can associate multiple subnets with the same route table. Select the link to learn more about route tables. https://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/VPC_Route_Tables.html.

- AWS automatically create and associates a **Dynamic Host Configuration Protocol (DHCP)** option set for your Amazon VPC upon creation and sets two options: domain-name-servers and domain-name.

- A **Security Groups** is a virtual stateful firewall that controls inbound and outbound network

traffic to AWS resources and Amazon EC2 instances. Select the link to learn more about security groups.
https://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/VPC_SecurityGroups.html

- A **Network Access Control List (NACL)** is an optional layer of security for your Amazon VPC that acts as a firewall for controlling traffic in and out of one or more subnets. Select the link to learn more information about NACL.
https://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/VPC_ACLs.html.

Optional Amazon VPC Components

- **Internet Gateway (IGW):** Allows access to the Internet from Amazon VPC.
- **Elastic IP (EIP) Addresses:** Static, public IP address that can be pulled from a pool for use on a temporary basis.
- **Elastic Network Interface (ENI):** Virtual network interface.
- **Endpoints:** Direct connection to another AWS service.
- **Peering:** Allows two Amazon VPCs to communicate.
- **NAT Address Translation (NATs) instances and NAT Gateways:** Accepts, translates, and forwards traffic within a private subnet.

Let's review some some optional Amazon VPC components:

- An **Internet Gateway (IGW)** is a horizontally scaled, redundant, and highly available Amazon VPC component that allows communication between instances in your Amazon VPC and the Internet. Select the link to learn more. .https://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/VPC_Internet_Gateway.html.

- An **Elastic IP (EIP) Address** is a static IPv4 address designed for dynamic cloud computing. An Elastic IP address is associated with your AWS account. Select the link to learn more. https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/elastic-ip-addresses-eip.html

- **Elastic Network Interface (ENI)** is a virtual network interface that you can attach to an instance in an Amazon VPC. Select the link to learn more. https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-eni.html

- An Amazon VPC **endpoint** enables you to create a private connection between your Amazon VPC and another AWS service without requiring access over the Internet or through a NAT instance, VPN connection, or AWS Direct Connect. Select the link to learn more. https://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/vpc-endpoints.html

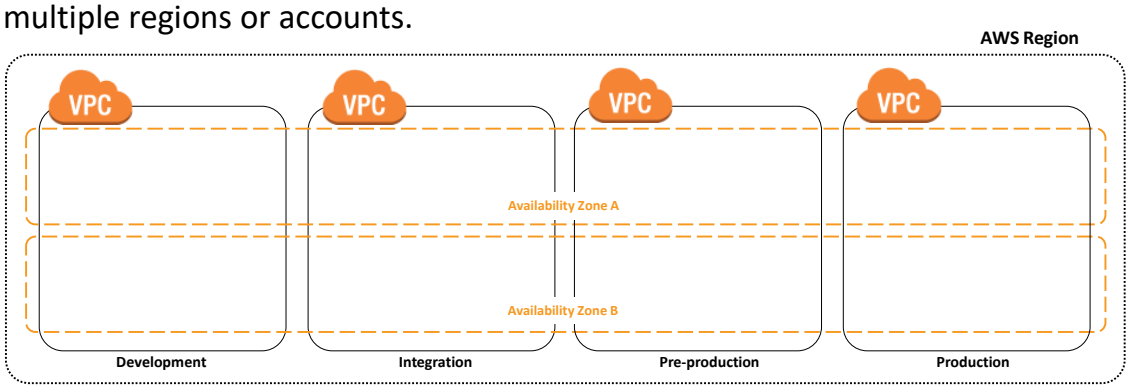
- An Amazon VPC **peering** connection is a networking connection between two Amazon VPCs that enables instances in either Amazon VPC to communicate with each other as if they are within the same network. Select the link to learn more. https://docs.aws.amazon.com/AmazonVPC/latest/PeeringGuide/Welcome.html

- **NAT Address Translation instances** is an Amazon Linux AMI designed to keep traffic from instances

within a private subnet. A **NAT Gateway** is an Amazon managed resources designed to operate just like a NAT instance, but is simpler to manage and highly available within an AZ.

Amazon VPC

- Amazon VPCs can include resources in more than one Availability Zone.
- You can have multiple Amazon VPCs in the same account and region and in multiple regions or accounts.



To summarize, Amazon VPC allows you to create a private network within the AWS cloud that uses many of the same concepts and constructs as an on-premises network.

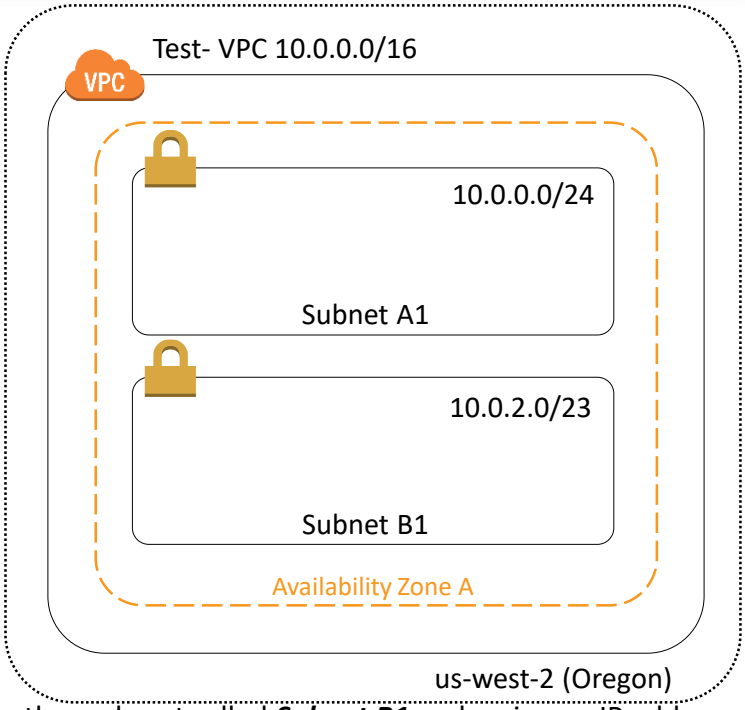
Amazon VPC allows you to:

- Include resources in more than one Availability Zone.
- Have multiple Amazon VPCs in each account or region and VPCs in as many regions as you'd like or in multiple accounts.
- You can connect your Amazon VPC to remote networks using a VPN connection.

Divide Your VPC into Subnets

Now, let’s understand how to divide your VPC into subnets.

Amazon VPC Example

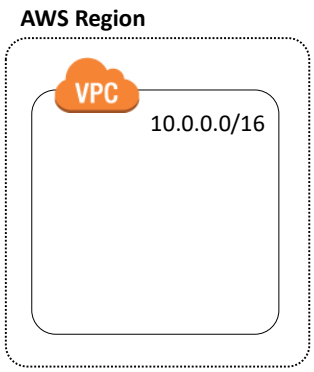


Finally, we create another sub-net called ***Subnet B1*** and assign an IP address space. This subnet contains 512 IP addresses.

Let's make a few more additions that will make ***Subnet A1*** accessible via the Internet.

Amazon VPC Address

- Each Amazon VPC must specify the IPv4 address range by choosing a Classless Inter-Domain Routing (CIDR) block like 10.0.0.0/16:
  - Address range cannot be changed after the Amazon VPC is created.
  - Address range can be large as /16 (65,536 available addresses) or as small as /28 (16 available addresses).
  - Addresses should not overlap addresses of connected networks.



When you create an Amazon VPC, you must specify the IPv4 address range by choosing a **Classless Inter-Domain Routing (CIDR)** block, such as 10.0.0.0/16.

The address range of the Amazon VPC cannot be changed after the Amazon VPC is created. An Amazon VPC address range may be as large as /16 (65,536 addresses available) or as small as /28 (16 addresses available) and should not overlap any addresses of other networks they are connected to.

VPCs and IP Addresses

- When you create your VPC, you specify its set of IP addresses with CIDR notation.
- **Classless Inter-Domain Routing (CIDR)** notation is a simplified way to show a specific range of IP addresses.
  - Example: 10.0.0.0/**16** = all IP addresses from 10.0.0.0 to 10.0.255.255
- **How does that work?** What does the **16** define?

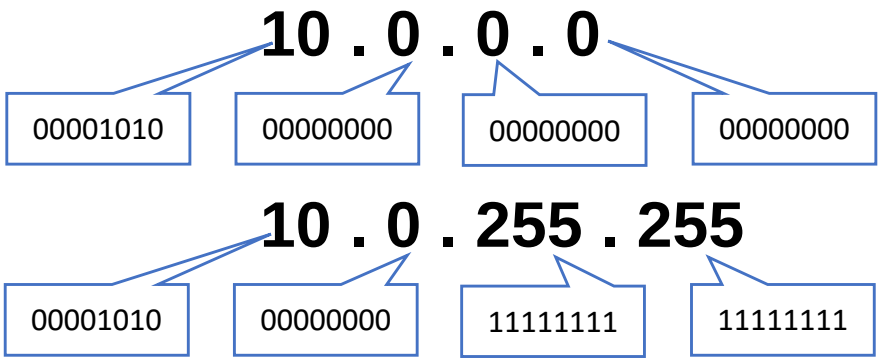
When you create your Virtual Private Cloud, you specify its set of IP addresses with Classless Inter-Domain Routing (or CIDR) notation. CIDR notation is a simplified way to show a specific range of IP addresses.

Displayed is the example: 10.0.0.0/16. This IP gives us all the available IP's displayed in the range from 10.0.0.0 to 10.0.255.255.

So how does that work and what does the 16 tell us? Let's review this in more depth as we continue.

IP Addresses and CIDR

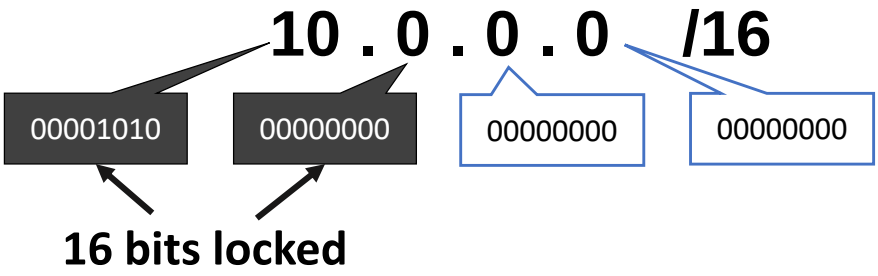
Every set of 3 digits in an IP address represents a set of 8 binary values (8 bits).



Every set of 3 digits in an IP address represents a set of 8 binary values (or 8 bits).

IP Addresses and CIDR: Part I

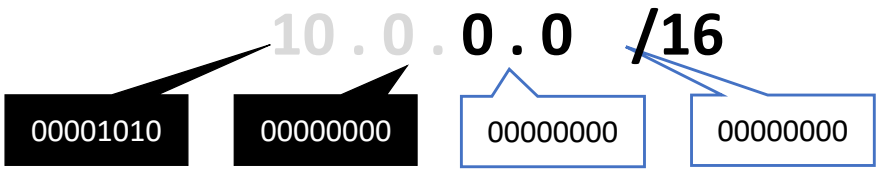
The 16 in the CIDR notation example represents how many of those bits are "locked down" and cannot change.



The 16 in the Classless Inter-Domain Routing notation example represents how many of those bits are "locked down" and cannot change. This means that you only have a certain amount of bits left to use as IP addresses.

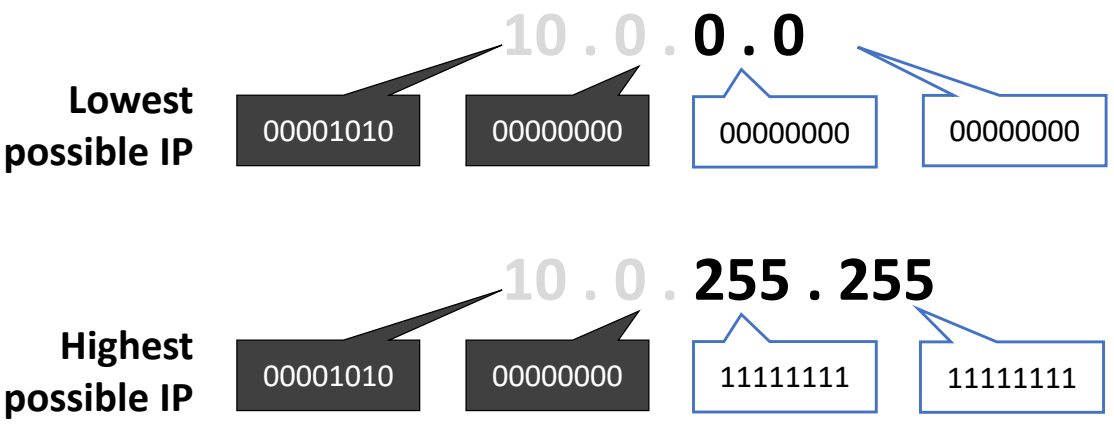
IP Addresses and CIDR: Part II

The unlocked bits can change between 1 and 0, allowing the full range of possible values.



Chiefs

CIDR Example: 10.0.0.0/16



Now we can see the lowest IP address available, all the way up to the highest IP address.

VPCs and IP Addresses

Amazon VPCs can use CIDR ranges between **/16** and **/28**.

For every **one step** a CIDR range increases, the total number of IP addresses is **cut in half**:

Dedicated Network Bits (CIDR)	Bits available to IPs (Total IPs)
/16	**65,536**
/17	**32,768**
/18	**16,384**
/19	**8,192**
/20	**4,096**
…	…
/28	**16**

Amazon VPCs can use CIDR ranges between /16 and /28.

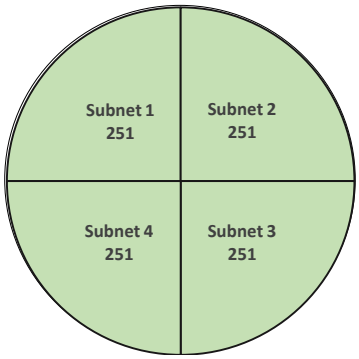
What this means is that under CIDR, you have bits that are dedicated to your network, and bits that are available to IP addresses. The larger the number under CIDR, the fewer the numbers that are left for use by IP addresses because the CIDR numbers are bits that are locked for the network. It's a tradeoff: a larger number means a smaller the number of IP addresses.

For every one step a CIDR range increases, the total number of IP addresses is cut in half.

What Are Subnets?

Subnets are **segments** or **partitions** of a network, divided by **CIDR range**.

Example:
A VPC with **CIDR /22**
includes 1,024 total IPs



Note: In every subnet, the first four and last IP addresses are reserved for AWS use.
- 10.0.0.0: Network address.
- 10.0.0.1: Reserved by AWS for the VPC router.
- 10.0.0.2: Reserved by AWS for mapping to Amazon provided DNS.
- 10.0.0.3: Reserved by AWS for future use.
- 10.0.0.255: Network broadcast address.

Subnets are segments, or partitions of a network, that are divided by the CIDR range.

This example with a CIDR of /22 means that you will have 1,024 total IP addresses. To create subnets of these IP addresses, they can be divided into four groups of 251. These four groups could be set up as two sets of public IP addresses, and two sets of private IP addresses.

The first four IP addresses and the last IP address in each subnet aren't available for you to use because they are reserved for Amazon Web Services use, and can't be assigned to an instance. For example, in a subnet with CIDR block 10.0.0.0 /22, the following five IP addresses are reserved:
- 10.0.0.0 is reserved for the network address.
- 10.0.0.1 is reserved by AWS for the VPC router.
- 10.0.0.2 is reserved by AWS for mapping to the Amazon provided Domain Name System, or DNS.
- 10.0.0.3 is reserved by AWS for future use.
- 10.0.0.255 is the network broadcast address. AWS does support broadcast in a VPC, so therefore this address is reserved.

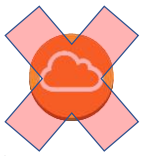
Note that these five IP addresses are automatically reserved on every network range.

Public and Private Subnets

Internet gateway

Public subnets

- If a subnet's traffic is routed to an internet gateway, the subnet is a *public subnet.*



internet gateway

Private subnets

- If a subnet's traffic does not have a route to an internet gateway, the subnet is a *private subnet.*

Learn more.

What is the difference between a public and a private subnet?

If a subnet's traffic is routed to an internet gateway, the subnet is a public subnet.

If a subnet's traffic is does not have a route to an internet gateway, the subnet is a private subnet.

For more information on public and private subnets, go to the VPCs and subnets page in the Amazon VPC documentation.
https://docs.aws.amazon.com/vpc/latest/userguide/VPC_Subnets.html

How to Use Subnets

Recommendation: Use subnets to define Internet accessibility.

Public subnets

- Include a routing table entry to an **Internet gateway** to support inbound/outbound access to the public Internet.

Private subnets

- Do not have a routing table entry to an Internet gateway and are **not directly accessible** from the public Internet.
- Typically use a "jump box" (NAT/proxy/bastion host) to support restricted, **outbound-only** public Internet access.

Subnets should be used to define which parts of the network are accessible to the internet, and which parts are not. Rather than defining your subnets based on application or functional tier-such as web, application, data, etc.-it's recommended that you organize subnets based on internet accessibility. This practice allows you to define clear, subnet-level isolation between public and private resources.

Public subnets include a routing table entry to an **internet gateway** to support inbound or outbound access to the public Internet.

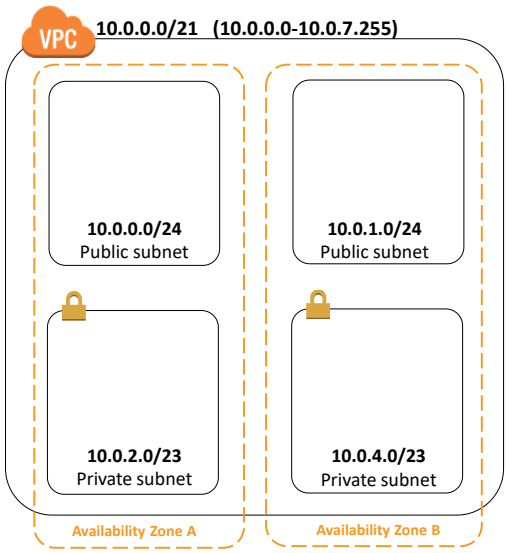
Private subnets do not have a routing table entry to an internet gateway, and are **not directly accessible** from the public internet.

If you have a private subnet that requires internet access-to download security patches for applications, for example-you'd need to use a "jump box" to support the restricted, **outbound-only** public internet access. A "jump box" is a NAT, proxy, or bastion host.

Subnets

Recommendation:

Start with **one public** and **one private** subnet per Availability Zone.



Displayed is an example of how to divide subnets. Subnets should be used to define internet accessibility, so there might not be a good reason to have more than one public and one private subnet per Availability Zone. In this environment, all of your resources that require direct access to the internet-including public-facing load balancers, NAT instances, bastion hosts, etc.-would go into the public subnet, while all other instances would go into your private subnet. An exception would be resources that require absolutely no access to the internet, either directly or indirectly. These resources would go into a separate private subnet.

Some environments try to use subnets to create layers of separation between "tiers" of resources, such as putting your backend application instances and your data resources into separate private subnets. This practice requires you to more accurately predict how many hosts you will need in each subnet, making it more likely that you will either run out of IP addresses more quickly, or leave too many IP addresses unused when they could be used elsewhere.

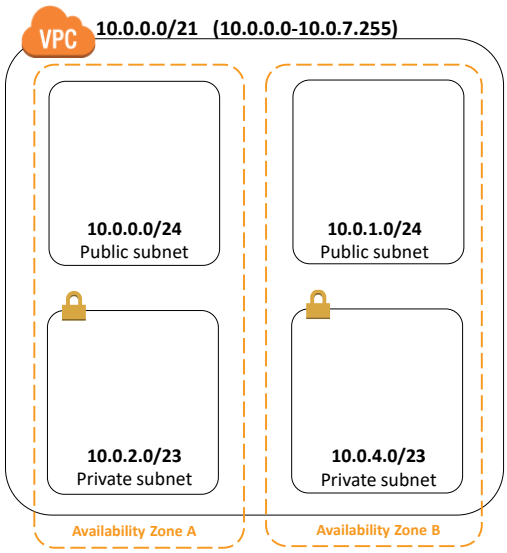
Subnets can provide a very basic element of segregation between resources by using a network access control list, or network ACL, rules. However, security

groups can provide an even more fine-grained level of traffic control between your resources, without the risk of overcomplicating your infrastructure and wasting or running out of IP addresses. With this approach, you just need to anticipate how many public and how many private IP addresses your VPC needs, and use other resources to create segregation between resources within a subnet.

Subnets: Part II

Recommendation:

Allocate substantially **more IP addresses for private subnets** than for public subnets.



The majority of resources on AWS can be hosted in private subnets, using public subnets for controlled access to and from the internet as necessary. You'll always need more private IP addresses than public IP addresses because the more resources you expose to the internet, the more vulnerable you become. You can protect your IP resources by placing them in a private subnet.

When you plan your architecture, it's important to try to anticipate how many hosts your VPC might need, and how many of those hosts can be placed in private subnets. The course will discuss strategies for placing public-facing resources in private subnets in more detail.

Subnet Sizes

Recommendation:
Consider larger subnets over smaller ones (/24 and larger).

Simplifies workload placement:

- Choosing where to place a workload among 10 small subnets is more complicated than with one large subnet.

Less likely to waste or run out of IP addresses:

- If your subnet runs out of available IP addresses, you can't add more to that subnet.
  - Ex.: If you have 251 IP addresses in a subnet that's using only 25, you can't share the unused 226 IP addresses with another subnet that's running out.

It’s recommended to consider larger subnets over smaller ones, such as slash 24 and larger. This simplifies your workload placement. Choosing where to place a workload among 10 small subnets is more complicated than choosing where to place the same workload with one large subnet.

You’re less likely to waste or run out of IP addresses. If your subnet runs out of available IP addresses, you can't add more IP addresses to that subnet. For example, if you have 251 IP addresses in a subnet that's using only 25 of them, you can't share the unused 226 IP addresses with another subnet that's running out. Consider where you will be five years down the road because this will save you a lot of time. Choose a larger range of IP ranges, rather than a smaller range.

Note that it’s no longer necessary to limit Address Resolution Protocol, or ARP, broadcast domains because this is solved by the VPC.

Select the Subnet Types

Which subnet type (public or private) should you use for these resources?

Data store instances

- **Private**

Batch processing instances

- **Private**

Backend instances

- **Private**

Web application instances

- **Public or private***

Which subnet type should you use for these resources?

While you can put web-tier instances into a public subnet, we actually recommend that you place your web-tier instances inside of private subnets that are behind a load balancer placed in a public subnet. Some environments require web application instances to be attached to Elastic IP addresses directly, even though you can also attach an Elastic IP address to a load balancer. In those cases, web application instances would need to be in a public subnet. We will talk more about load balancers in a later module.

Default VPCs and Default Subnets

What are default VPCs and default subnets, and when should you use them?

What is a Default VPC?

Details about default VPCs:

- **Each Region** in your account has a default VPC.
- Default CIDR is **172.31.0.0/16.**
- If you create a VPC-based resource (Amazon EC2, Amazon RDS, Elastic Load Balancing, etc.) but **don't specify a custom VPC**, it will be placed in your default VPC in that region.
- Includes a default **subnet**, **IGW**, main **route table** connecting default subnet to the IGW, default **security group**, and default **NACL**.
- **Configurable** the same as other VPCs; e.g., adding more subnets.

The following list includes details about default VPCs:

- Each Region in your account has a default VPC.
- The default CIDR range is 172.31.0.0/16.
- If you create a VPC-based resource (such as Amazon EC2, Amazon RDS, Elastic Load Balancing, etc.) but don't specify a custom VPC, it will be placed in your default VPC in that Region. That includes the default internet gateway. This scenario means that the VPC set up as the default has internet-routable traffic, the default security group, and the default network ACLs. In general, this situation is never a good idea because many people know the default CIDR range, and that those ranges are automatically connected to internet gateways by default unless they are disabled.
- Default VPCs includes a default subnet, an internet gateway, a main route table that connects the default subnet to the internet gateway, a default security group, and a default network ACL.
- Default VPCs are configurable like other VPCs. For example, you can add more subnets.

What Is a Default Subnet?

Default subnets in default VPCs:

- Created **within each Availability Zone** for each default VPC.
- **Public** subnet with a CIDR block of **/20** (4,096 IP addresses).
- You can convert it (and any public subnet) into a **private** subnet by removing its route to the IGW.
- When a new Availability Zone is added to a region, your default VPC in that region gets a subnet placed in the new Availability Zone (unless you've made modifications to that VPC).

Default subnets in default VPCs:

• Are created within each Availability Zone for each default VPC.

• Have a public subnet with a CIDR block of /20 (with 4,096 IP addresses). The default subnets are created within each Availability Zone. There is a public subnet with a CIDR block range of /20 with over 4,000 IP addresses.

• Can be converted, like any public subnet, into a private subnet by removing its route to the internet gateway. Using the default subnet and the default VPC is like leaving the router login and password set to admin. Many people might know how to use these default credentials, and it’s not very secure.

• When a new Availability Zone is added to a region, your default VPC in that

region gets a subnet placed in the new Availability Zone (unless you've made modifications to that VPC).

Default VPCs and Subnets

Recommendation: Use default VPCs and their subnets only for experimenting in your AWS account.

- Default VPCs are a quick start solution.
  - They provide an easy way to test launching instances of your VPC-based resources, without having to set up a new VPC.
- For real-world applications, create your own VPCs and subnets.
  - You'll have greater control and knowledge of their configurations.
  - Possible to re-establish default VPC if accidentally deleted.

We recommended that you use default VPCs and their subnets only for experimenting in your AWS account.

Default VPCs are a quick start solution. They provide an easy way to test launching instances of your VPC-based resources without having to set up a new VPC.

For real-world applications, create your own VPCs and subnets. You'll have greater control and knowledge of their configurations. It’s possible to re-establish a default VPC if it’s accidentally deleted.

This week – Network Design

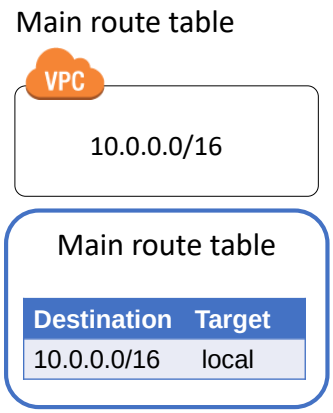
- Choosing a Region and Selecting Availability Zones
- Creating a Virtual Private Cloud (VPC) and Subnets
  - VPC components and network address - CIDR
  - Private and Public Subnets
  - Default VPCs and Default Subnets
- **Controlling VPC Traffic**
  - Route tables, Security groups, Network ACLs, Internet gateways, NATs, Bastion Hosts
- Multiple VPCs and AWS Accounts

SWINBURNE UNIVERSITY OF TECHNOLOGY

Directing Traffic Between VPC Resources

Route tables:

- Determine where network traffic is routed
- Main (default) and custom route tables
- All route tables include a local route entry
  - The local route entry cannot be deleted
- Only one route table per subnet



Best practice: Use custom route tables for each subnet.

As we have discussed, your Amazon VPC is your own logically isolated part of the AWS Cloud. Every VPC has a default route table.

A route table is a map that tells you how to enter and leave your network. It contains a set of rules, called routes, which are used to determine where network traffic is directed. You can have main route tables, which is the default that's displayed, and custom route tables. For example, you can use custom route tables if you need infrastructure within your VPC that can connect back to your on-premises environment.

All route tables include a local route entry. When you create a VPC, it automatically has a main route table—and every route table in a VPC —contains only a single route: a local route that enables communication within the VPC. You can't delete the local route in a route table. When you launch an instance in the VPC, the local route automatically covers that instance. You don't need to add the new instance to a route table. You can create additional custom route tables for your VPC.
Each subnet in your VPC must be associated with a route table, which controls the routing for the subnet. If you don't explicitly associate a subnet with a particular route table, the subnet is implicitly associated with—and uses—the main route table. A subnet can be associated with only one route table at a time, but you can associate multiple subnets with the same route table.

A best practice is to use custom route tables for each subnet, which enables

Securing VPC Traffic with Security Groups

Security groups:

- Are stateful applications that act as virtual firewalls controlling inbound and outbound traffic for one or more instances.
- Deny all incoming traffic by default and use allow rules that can filter based on network protocols (TCP, UDP, and ICMP protocols).
- Use a CIDR block or security group to create layers of security to define access to assets.
- If your inbound request is allowed, the outbound response is allowed automatically.

Learn more.

Security groups are stateful applications that act as virtual firewalls controlling inbound and outbound traffic for one or more instances.

By default, a security group will deny all incoming traffic. You can use rules to control that traffic that filter based network protocols such as Transmission Control Protocol, or TCP, User Datagram Protocol, or UDP, and Internet Control Message Protocol, or ICMP.

You can also use an entire CIDR block or another security group to create layers of security to define who or what has access to your assets.
This is the first layer of protection around instances.

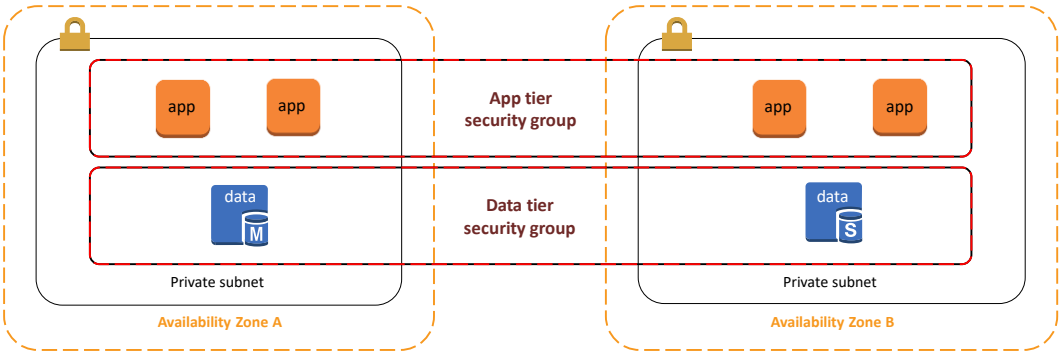
Because a security group is stateful, if your inbound request is allowed, the outbound response is allowed automatically. For example, if you initiate an HTTP request to your instance from your home computer, and your inbound security group rules allow HTTP traffic, information about the connection including the source IP address and port number) is tracked. The HTTP response from your instance to your home computer is recognized as part of an established connection and allowed through the security group, even if the security group

rules restrict outbound HTTP Traffic.

For further information, select the link.
http://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/VPC_Networking.html

Security Groups

Use security groups to control traffic
into, out of, and between resources.



Use security groups to control traffic into, out of, and between resources. Displayed is an example of what security groups might look like, that spans two Availability Zones. One is set up for the application tier and a second is set up for the data tier.

In the example, the application and data tiers both exist in the private subnets of this VPC. To provide component isolation, the application servers share one security group, and the Amazon RDS instances share another.

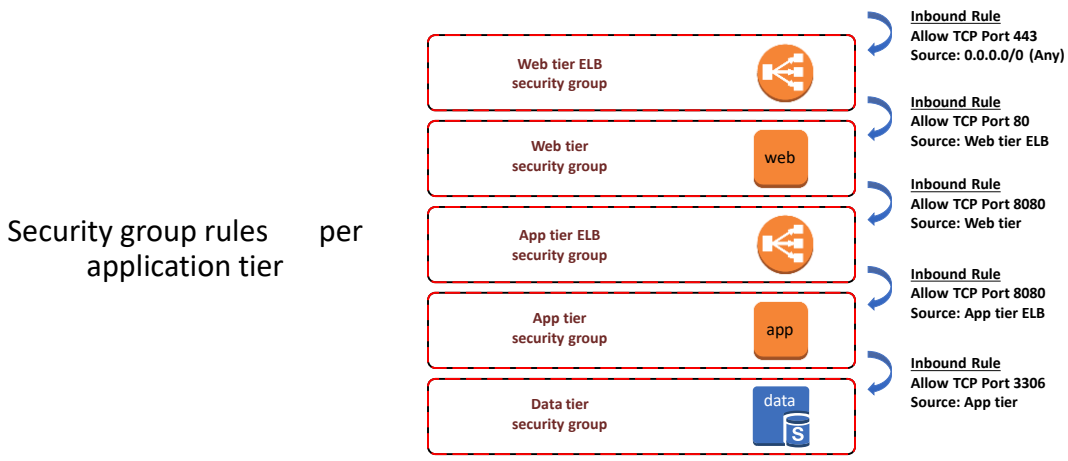
How Security Groups are Configured

- By default, all newly created security groups **allow all outbound traffic** to all destinations.
  - Modifying the default outbound rule on security groups **increases complexity** and is not recommended unless required for compliance.
- Most organizations create security groups with inbound rules for **each functional tier** (web/app/data) within an application.

By default, all newly created security groups allow all outbound traffic to all destinations. Be careful not to make these security groups too complex. Modifying the default outbound rule on security groups increases complexity, and it's not recommended unless it's required for compliance.

Most organizations create security groups with inbound rules for each functional tier—including web, application, and data—within an application.

Security Group Chaining Diagram



Displayed is an example of a chaining diagram that shows a chain of security groups. The inbound and outbound rules are set up so that traffic can only flow from the top tier to the bottom tier, and back up again. The security groups act as firewalls that prevent a security breach in one tier from automatically providing subnet-wide access to all resources in the compromised client.

This diagram also has a web tier Elastic Load Balancing security group. It's allowed to talk to the web tier security group over port 80, but that traffic has to come from the web tier elastic load balancer. The traffic can't come directly from the internet and access the web tier security group and its servers. The third tier is the application tier elastic load balancer. That tier only accepts traffic that comes from the web tier security group over port 8080. The application tier security group servers will only accept traffic that comes from the application tier elastic load balancer group from port 8080. Finally, when you get to the data tier, it will only accept inbound traffic from the application tier over port 3306.

With this kind of security chaining, someone from the internet can't get beyond the web tier load balancer security group.

Network ACLs

- A network ACL is a **virtual firewall** that controls traffic in and out of a subnet.
- **Allow all** incoming/outgoing traffic by default and use **stateless** rules to allow or deny traffic.
- "Stateless rules" inspect all inbound and outbound traffic and do not keep track of connections.
- An allow rule must be explicitly created.

A network access control list, or network ACL, is a virtual firewall that controls access to a subnet. Each subnet can have only one network ACL assigned. If you don’t create a network ACL for a subnet, a “default network ACL” will be assigned.

Network ACLs use inbound and outbound rule definitions. Each of these rules either “allow” or “deny” the traffic that’s defined by source, destination, port, and protocol.

You can define multiple rules for both inbound and outbound traffic. Rules are evaluated in numerical order. When you create a network ACL, it’s automatically configured to DENY ALL traffic, and you need to configure the allowed traffic accordingly.

The default network ACL-which is automatically assigned to a subnet when you create it-is configured to ALLOW ALL traffic. We recommended that you evaluate your security requirements and define each network ACL accordingly.

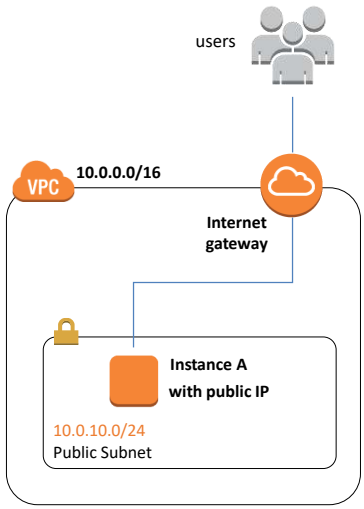
Network ACLs are “stateless.” "Stateless rules" inspect all inbound and outbound

traffic, and they don’t keep track of connections. Because they don’t keep track of sessions, when you allow traffic based on an incoming rule, network ACLs don’t automatically allow the corresponding outgoing traffic. An allow rule must be explicitly created.

Directing Traffic to Your VPC

Internet gateways:

- Allow communication from internet into VPC.
- Are horizontally scaled, redundant, and highly available by default.
- Provide a target in your subnet route tables for internet-routable traffic.



Internet gateways allow communication from the internet into your VPC.

They are horizontally scaled out, redundant, and highly available by

default. They provide a way for you to get access to the internet, and they allow traffic on the internet to come to you by providing a target in your subnet route tables for internet-routable traffic.

In the example, we have an internet setup connected to a VPC. Because the instance has a public IP address, the internet can access the public instance with the public IP address.

Directing Traffic to Your VPC

aws academy

To enable access to or from the internet for instances in a VPC subnet, you must:

- Attach an internet gateway to your VPC.
- Ensure that your subnet's route table points to the internet gateway.
- Ensure that instances in your subnet have public IP addresses or Elastic IP addresses.
- Ensure that your network ACLs and security groups allow the relevant traffic to flow to and from your instance.

If you want to enable access to or from the internet for instances in a VPC subnet, you must:

- Attach an internet gateway to your VPC.
- Ensure that your subnet's route

table points to the internet gateway.

- Ensure that instances in your subnet have public IP addresses or Elastic IP addresses.

- And ensure that your network ACLs and security groups allow the relevant traffic to flow to and from your instance.

Outbound Traffic from Private Instances

Network Address Translation services:

- Enable instances in the private subnet to initiate outbound traffic to the internet or other AWS services.
- Prevent private instances from receiving inbound traffic from the internet.
- Two primary options:

1. Amazon EC2 instance set up as a NAT in a public subnet
2. NAT Gateway

Note: Not Free-tier!

Network Address Translation, or NAT, services enable instances in the private subnet to initiate outbound traffic to the internet or to other AWS services. NAT services also prevent private instances from receiving inbound traffic from the Internet.

For example, you would use a NAT service if you have a database that you want to keep in the private subnet, but still let it access database patches. The NAT service allows your instance to reach out onto the internet to download patches without letting traffic come back in and access the instance.

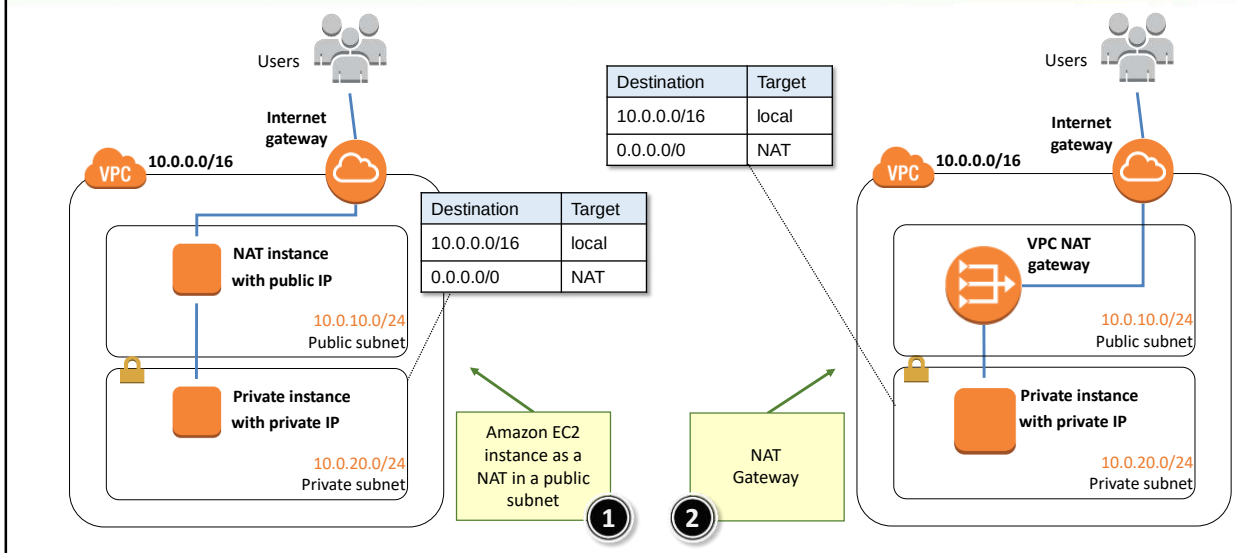
AWS offers two primary options for using NAT services:

- An Amazon EC2 instance that's set up as a NAT service in a public

subnet

- And a NAT Gateway

NAT Gateway Service



The example displayed shows the NAT instance, with the private instance on the left. The right side of the diagram shows the VPC NAT gateway. The NAT gateway service is fully scaled, redundant and highly available.

For customers who require a private subnet on their IPv6-enabled VPCs, we are introducing a new resource within the VPC called the Egress-only Internet Gateway, which can be set up to allow one-way access to internet resources. With the Egress-only Internet Gateway, outgoing traffic to the internet will be allowed. However, incoming traffic that's initiated from the internet will be blocked. There is no additional charge to use the Egress-only Internet Gateways. However, data transfer charges apply.

VPC NAT Gateways vs. NAT Instances on Amazon EC2

	VPC NAT gateway	NAT instance
Availability	Highly available by default	Use script to manage failover
Bandwidth	Bursts to 10 Gbps	Based on bandwidth of instance type
Maintenance	Managed by AWS	Managed by you
Security	NACLs	Security groups and NACLs
Port forwarding	Not supported	Supported
Scope	Availability Zone	Availability Zone

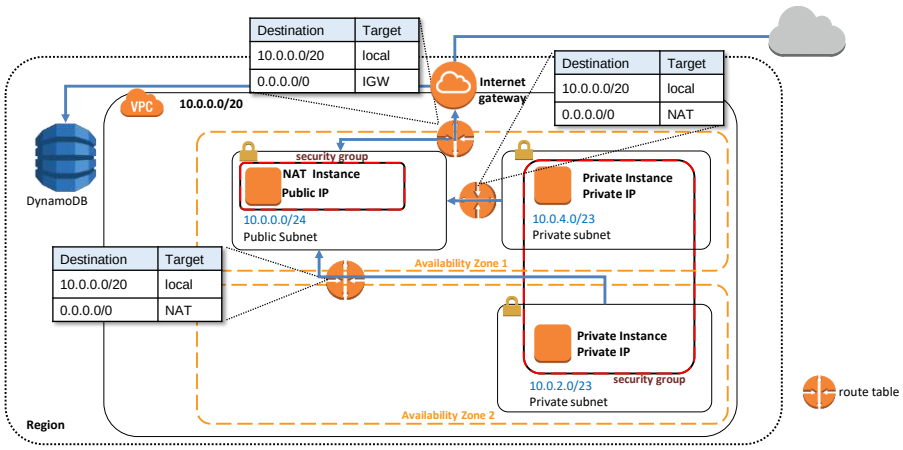
It's important to carefully consider your decision on which NAT solution you will use, whether it's a gateway or an instance. While VPC NAT Gateways offer all the advantages of a service that's managed by AWS—such as being inherently highly available—they might not provide the exact level of control that your application needs.

One example where a NAT gateway might not work as a solution is when you need more than 10GB of bandwidth, which is the maximum amount of bandwidth that the NAT gateway can handle.

Another important difference between the VPC NAT gateway and a NAT instance is port forwarding. The VPC NAT gateway does not support Port 40.

There are also cost differences to consider between the two options.

Subnets, Gateways, and Routes



Let's look at how subnets, gateways, and routes work together.

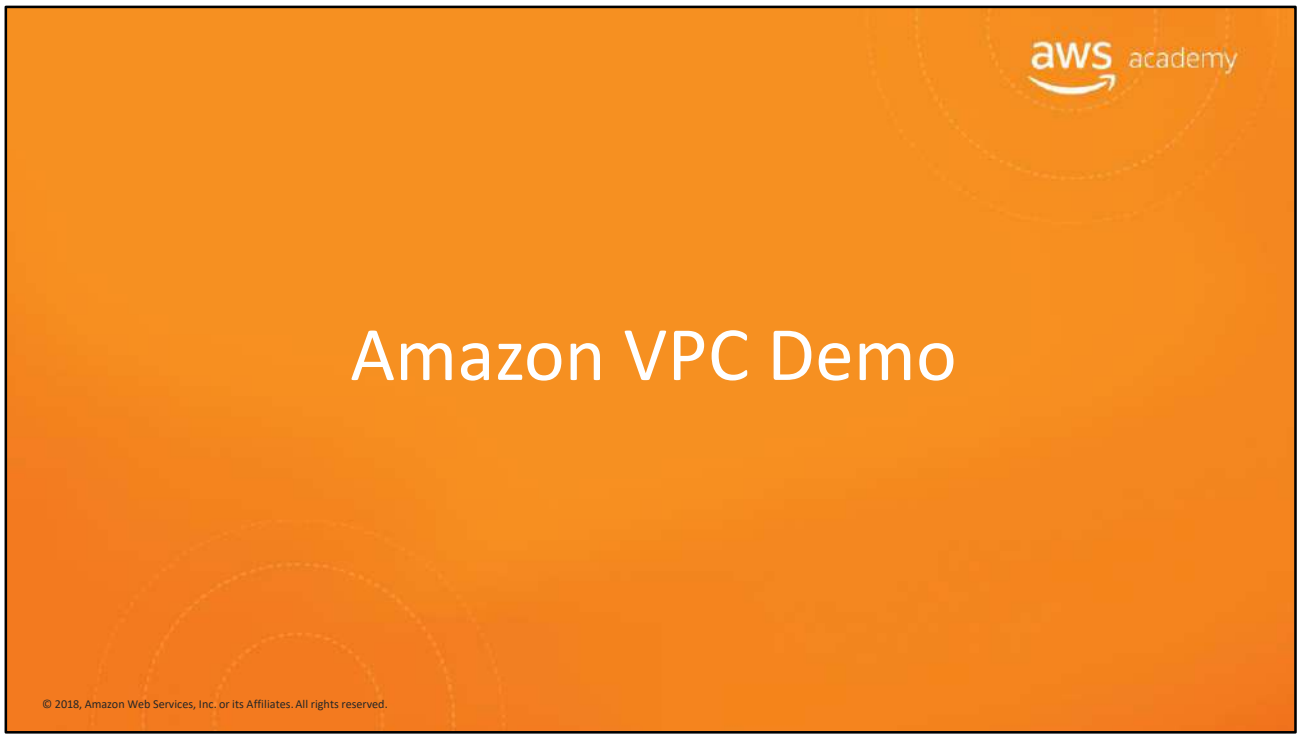
This diagram, we have two Availability Zones, two instances and private subnets, the NAT instance with the public IP address, and the public subnet. We also have an internet gateway connected to the VPC.

We have a NAT instance, but it needs a route to the internet gateway so that the instance can talk to the internet. You can see that the internet is in the route table, as indicated by the 0.0.0.0/0 address. The targets that use the internet gateway have the local route group, as indicated by the 10.0.0.0/20 address. If you want the private subnet to be able connect to the internet, it has to go through the NAT instance. The 0.0.0.0 address indicates that it must route to the NAT. The NAT will route the request to the Internet.

The second private instance currently has no route. If that instance needs to go to the internet, a route needs to be created to the NAT instance with the public IP address 0.0.0.0/0 with the target of NAT. This will enable the private instances to access the internet.

You can further tighten security using security groups. There is a security group for the two private instances. There is also a security group for the public NAT group. The security groups and the route tables helps control traffic within the VPC.

The diagram also contains a DynamoDB instance that sits outside of the VPC, but in the same Region as the VPC. Traffic could be routed to it via the internet gateway. Remember that there will be some services that reside outside of your VPC. One of those services is DynamoDB.



Please review the Amazon VPC demonstration: M2_S1_CoreSVCS.mp4.

This video demonstration can be found in the learning management system.

This week – Network Design

- Choosing a Region and Selecting Availability Zones
- Creating a Virtual Private Cloud (VPC) and Subnets
  - VPC components and network address - CIDR
  - Private and Public Subnets
  - Default VPCs and Default Subnets
- Controlling VPC Traffic
  - Route tables, Security groups, Network ACLs, Internet gateways, NATs, Bastion Hosts
- **Multiple VPCs and AWS Accounts**

SWINBURNE UNIVERSITY OF TECHNOLOGY

Using One VPC

There are **limited** use cases where one VPC could be appropriate:

- High-performance computing environments
- Microsoft active directory for identity management
- Small, single applications managed by one person or very small team

For **most** use cases, there are two primary patterns for organizing your infrastructure:

Multi-VPC or Multi-Account

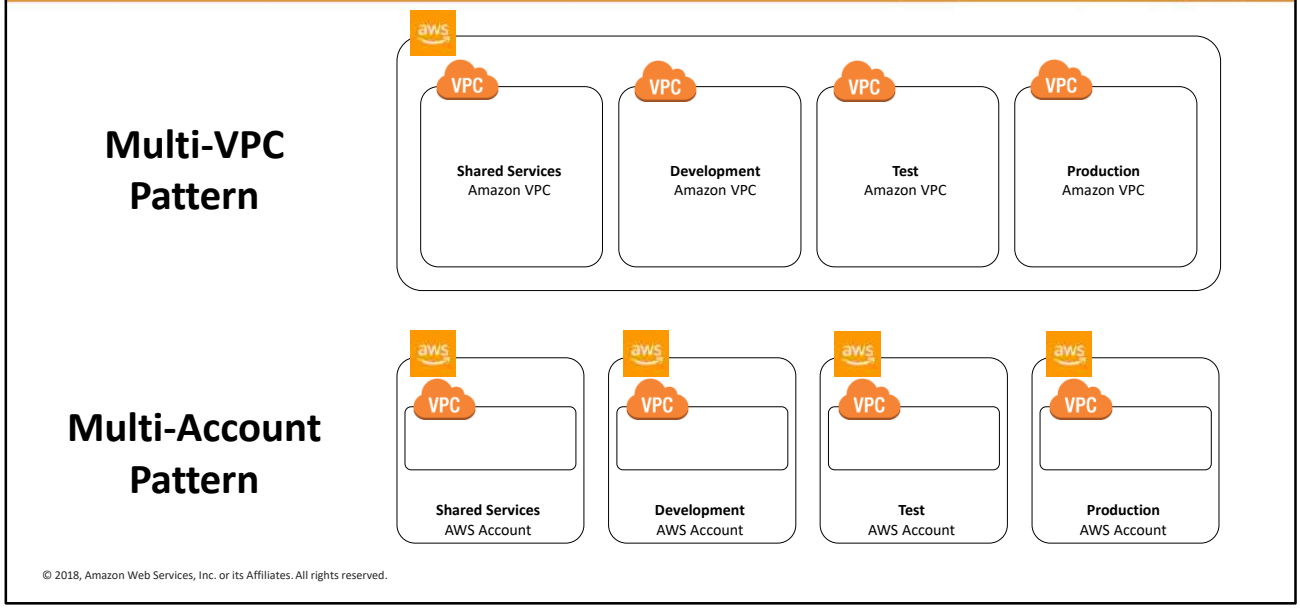
There are limited use cases where one Virtual Private Cloud could be appropriate. High-performance computing environments might work best entirely within a single VPC, as a single VPC environment will have lower latency than one that's spread across multiple VPCs.

The use of Microsoft Active directory for identity management might best be limited to one VPC for the strongest security measures.

For small, single applications that are managed by one person or a very small team, it might be easiest to use one Virtual Private Cloud.

In most cases, there are two primary patterns for organizing your infrastructure: Multi-VPC or Multi-Account.

Multi-VPC and Multi-Account Patterns



The Multi-VPC pattern has one Region with four different VPCs—shared services, development, test, and production each have their own VPC.

With the Multi-Account pattern, you can have multiple Amazon web services accounts with the same information—such as shared services, development, test and production—instead of having multiple VPCs.

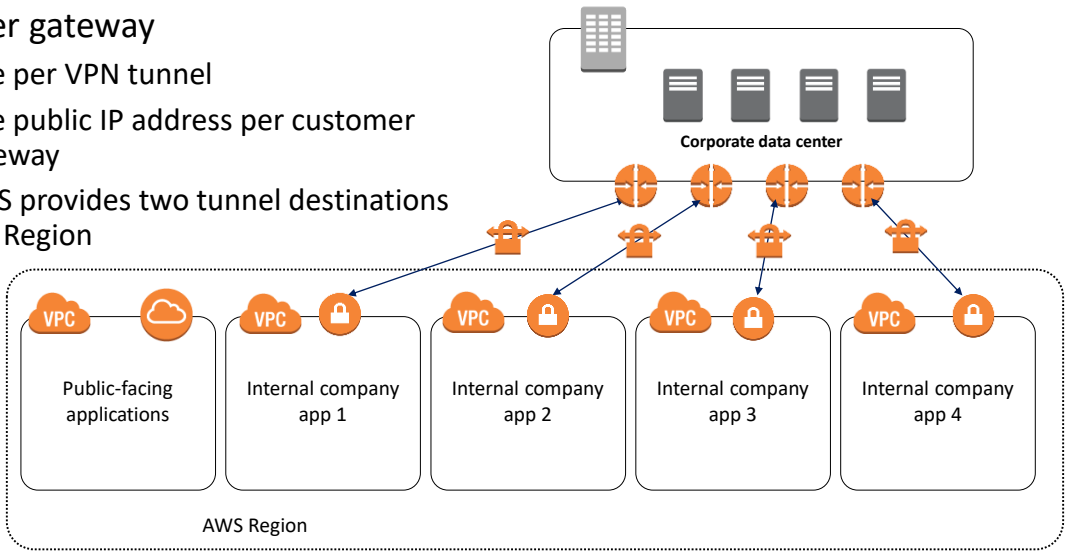
Can you connect multiple VPCs to each other?

Now, let’s discover how to connect multiple VPCs to each other.

VPN Hub And Spoke Architecture

Customer gateway

- One per VPN tunnel
- One public IP address per customer gateway
- AWS provides two tunnel destinations per Region



If multiple VPCs need to connect to the data center, you can see the challenge of using a customer gateway.

There can be one customer gateway per VPN tunnel, and one public IP address per customer gateway. AWS provides two tunnel destinations per Region.

In this situation, you would have to maintain unique IP addresses on your end and on your route. You can have a VPN hub-and-spoke architecture to work around this situation. However, is it really an efficient solution?

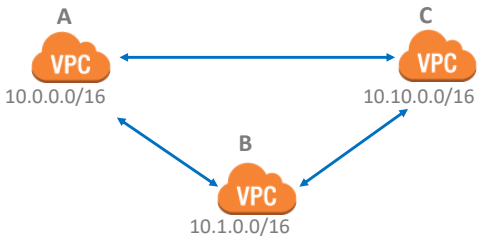
With a VPN hub-and-spoke architecture:

- There are Amazon EC2 VPN instances to the central customer gateway.
- There are two Amazon EC2-based VPN endpoints in each spoke to support high availability.
- There is a central VPC (hub) that contains common services for all application VPCs.
- There is a dynamic routing protocol between the spokes and the hub.

Resolution: VPC Peering

To establish a VPC peering connection:

- Owner sends a request to create VPC
- Owner of peer VPC accepts connection
- To enable flow of traffic, add a route
- Update security group rules
- VPC peering connection is one-to-one relationship between two VPCs



Amazon EC2 now allows peering relationships between VPCs.

In the diagram, VPCs A and B are peered, which does not mean that C can talk to A. By default, VPC peering does not allow C to connect to A unless they are explicitly established as peers. Therefore, you control which VPCs can talk to each other.

To establish a VPC peering connection, the owner of the requester VPC—or local VPC—sends a request to the owner of the peer VPC to create the VPC peering connection. The peer VPC can be owned by you or another AWS account, and can't have a CIDR block that overlaps with the requester VPC's CIDR block. The owner of the peer VPC has to accept the VPC peering connection request to activate the VPC peering connection. To enable the flow of the traffic between the peer VPCs using private IP addresses, add a route to one or more of your VPC's route tables that points to the IP address range of the peer VPC. The owner of the peer VPC adds a route to one of the VPC's route tables that points to the IP address range of your VPC. You might also need to update the security group rules that are associated with your instance to ensure that traffic to and from the peer VPC is not restricted.

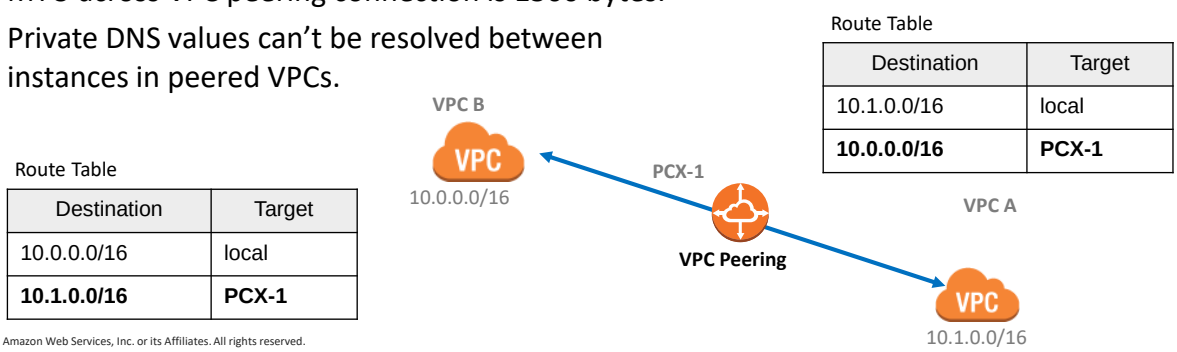
A VPC peering connection is a one-to-one relationship between two VPCs. You can create multiple VPC peering connections for each VPC that you own, but transitive peering relationships are not supported: you will not have any peering relationship with VPCs that your VPC is not directly peered with. You can create a VPC peering connection between your own VPCs, or with a VPC in another AWS account within a single Region.

Amazon EC2 now allows peering relationships to be established between virtual private clouds, or VPCs, across different Regions. Inter-Region VPC peering allows

VPC resources that run in different Regions—like Amazon EC2 instances, Amazon RDS databases, and AWS Lambda functions—to communicate with each other. This inter-Region communication occurs without requiring gateways, VPN connections, or separate network services. Data that's transferred across inter-Region VPC peering connections is charged at the standard inter-Region data transfer rates.

How Does VPC Peering Work?

- There is a limit on the number of active and pending VPC peering connections
- VPC peering does not support transitive peering relationships.
- You can't have more than one VPC peering connection between the same two VPCs at the same time.
- MTU across VPC peering connection is 1500 bytes.
- Private DNS values can't be resolved between instances in peered VPCs.



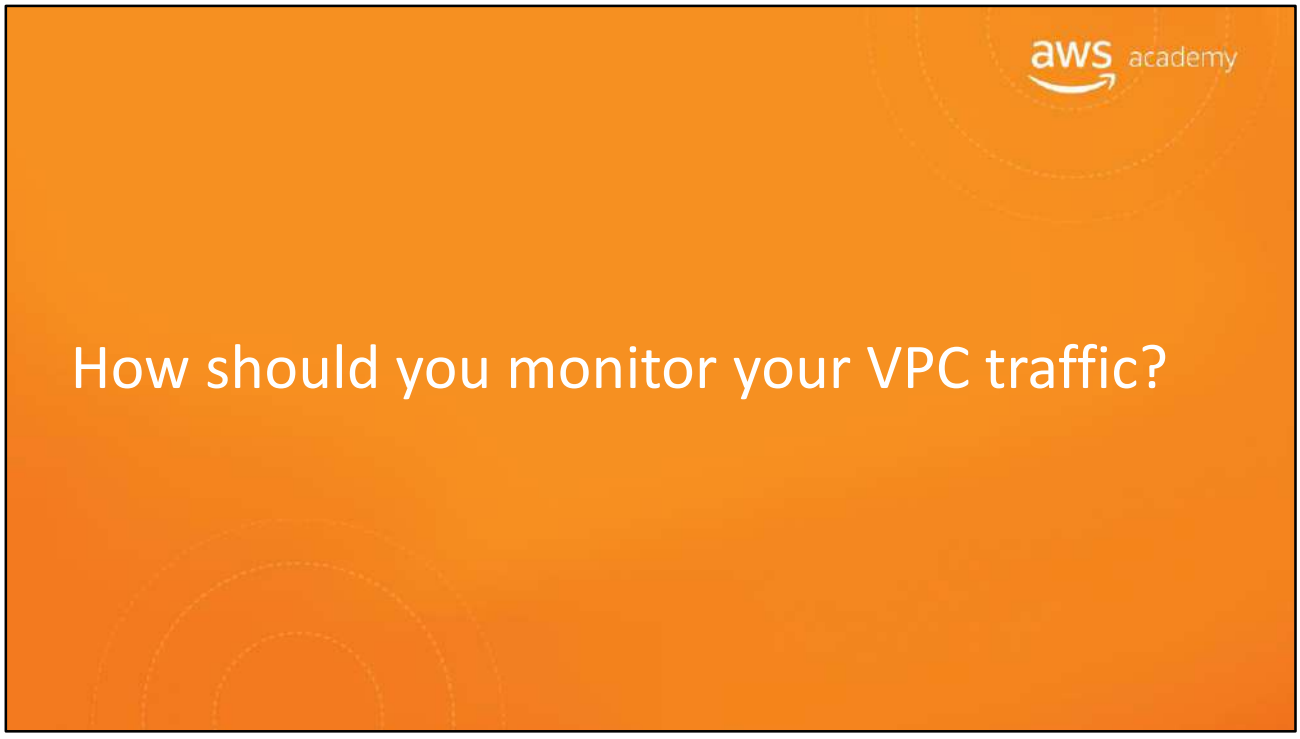
In this example, an entire CIDR block is opened, but you can specify an IP address on the route table.

To create a VPC peering connection with another VPC, you need to be aware of the following limitations and rules:
• There is a limit on the number of active and pending VPC peering connections that you can have per VPC.
• VPC peering does not support transitive peering relationships. In a VPC peering connection, your VPC will not have access to any other VPCs that the peer VPC might be peered with. This includes VPC peering connections that are established entirely within your own AWS account.
• You can't have more than one VPC peering connection between the same two VPCs at the same time.
• The Maximum Transmission Unit (MTU) across a VPC peering connection is 1500 bytes.
• A placement group can span peered VPCs. However, you will not get full-bisection bandwidth between instances in peered VPCs.
• Unicast reverse path forwarding in VPC peering connections is not supported.
• Private DNS values can't be resolved between instances in peered VPCs.

Traffic using inter-Region VPC peering always stays on the global AWS backbone, and traffic never traverses the public internet, which reduces threat vectors, such as common exploits and distributed denial of service, or DDOS, attacks.

You can now reference security groups in a peered VPC in both inbound and outbound rules. This functionality is supported cross-account, so the two VPCs can be in different

can reference security groups from a peered VPC by using the AWS Management Console, the AWS CLI, and through SDKs.



How should you monitor your VPC traffic?

Amazon VPC Flow Logs

- Captures traffic flow details in your VPC
  - Accepted and rejected traffic
- Can be enabled for VPCs, subnets, and ENIs
- Logs published to CloudWatch Logs

Use cases:

- Troubleshoot connectivity issues.
- Test network access rules.
- Monitor traffic.
- Detect and investigate security incidents.

How can you verify that the configured network access rules are working as expected? Many organizations typically collect, store, monitor, and analyze network flow logs for various purposes, including troubleshooting connectivity and security issues, and testing network access rules.

VPC Flow Logs is a feature that enables you to capture information about the IP traffic that goes to and from network interfaces in your VPC. The flow log captures accepted and rejected traffic flow information for all network interfaces in the selected resource. The information that's captured by the flow logs can help you with a number of tasks. For example, you can troubleshoot why specific traffic is not reaching an instance, which in turn can help you diagnose overly restrictive security group rules. You can also use flow logs as a security tool to monitor the traffic that is reaching your instance. You can create alarms to notify you if certain types of traffic are detected, and you can also create metrics to help you to identify trends and patterns.

You can create a flow log for a VPC, a subnet, or a network interface. If you create a flow log for a subnet or VPC, each network interface in the VPC or subnet is monitored. Flow log data is published to a log group in CloudWatch Logs, and each network interface has a unique log stream. Log streams contain flow log records, which are log events that consist of fields that describe the traffic for that network interface. Amazon CloudWatch and CloudWatch Logs are covered later.

You can analyze the data that's captured from flow logs with your own applications or with solutions that are available from AWS Marketplace.

SWINBURNE UNIVERSITY OF TECHNOLOGY

COS80001
Cloud Computing Architect

Lecture 03 Networks

- **Caching with CloudFront**
- **Routing with Route53**

includes material from
ACA Modules 3 and 7

This week – Networks outside the VPC

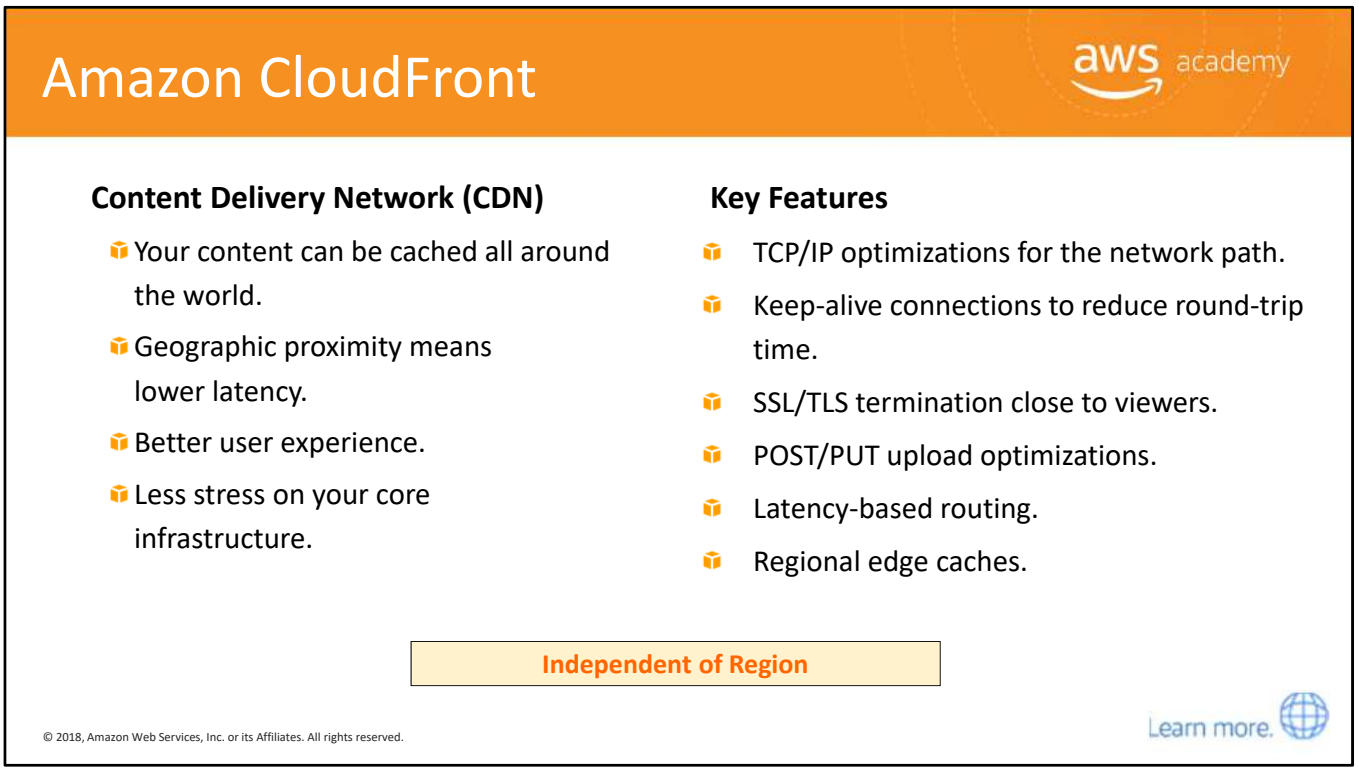
- **Web Caching**
  - **CloudFront**
- **Routing across Regions**
  - **Route53**

SWINBURNE UNIVERSITY OF TECHNOLOGY

Caching with Amazon CloudFront

SWINBURNE UNIVERSITY OF TECHNOLOGY

Now, let's discuss caching with Amazon CloudFront.



Amazon CloudFront is a web service that speeds the distribution of static and dynamic web content to users,—such as Hypertext Markup Language, or HTML, files; CSS files; JavaScript files; and image files. CloudFront is a content delivery network—or CDN—that delivers cached content through a worldwide network of data centers that are called edge locations. This geographic proximity means lower latency when you serve content to users. This makes it more cost effective and faster for a better user experience, as well as putting less stress on your core infrastructure.

Key features for CloudFront include:

- TCP/IP optimizations for the network path.
- Keep-alive connections to reduce round-trip time.
- SSL/TLS termination that is close to viewers.
- POST and PUT upload optimizations.
- Latency-based routing.
- And regional edge caches.

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. This process means that content is delivered to the user with the best possible performance. If

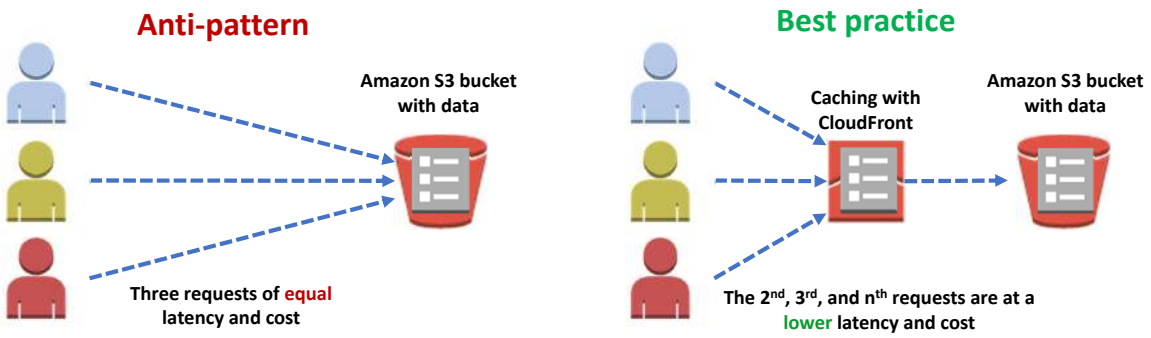
the content is already in the edge location with the lowest latency, CloudFront delivers it immediately. If the content is not currently in that edge location, CloudFront retrieves it from an Amazon S3 bucket or an HTTP server—for example, a web server—hat you have identified as the source for the definitive version of your content.

In addition, Amazon CloudFront has another type of edge location, which is called the regional edge cache. Regional edge caches further improves performance for your viewers. They can also help reduce the load on your origin resources, which minimizes the operational burden and the costs that are associated with scaling your origin resources. Regional edge caches are turned on by default for your CloudFront distributions, and you do not need to make any changes to your distributions to take advantage of this feature. There are also no additional charges to use this feature.

To learn more, go to the following page about CloudFront:
https://aws.amazon.com/cloudfront/details/

Best Practice: Caching

Implement caching at **multiple layers** of an architecture. It can **reduce cost and latency** and **increase performance** of applications.



It is an architectural best practice is to implement caching at multiple layers of an architecture to reduce cost and latency, and to increase the performance of applications. It is more cost-effective to distribute files from CloudFront than from an Amazon S3 bucket.

Caching is the process of temporarily storing data or files in an intermediary location between the requester and the permanent storage. The purpose of caching is to make responding to future requests faster and reduce network throughput.

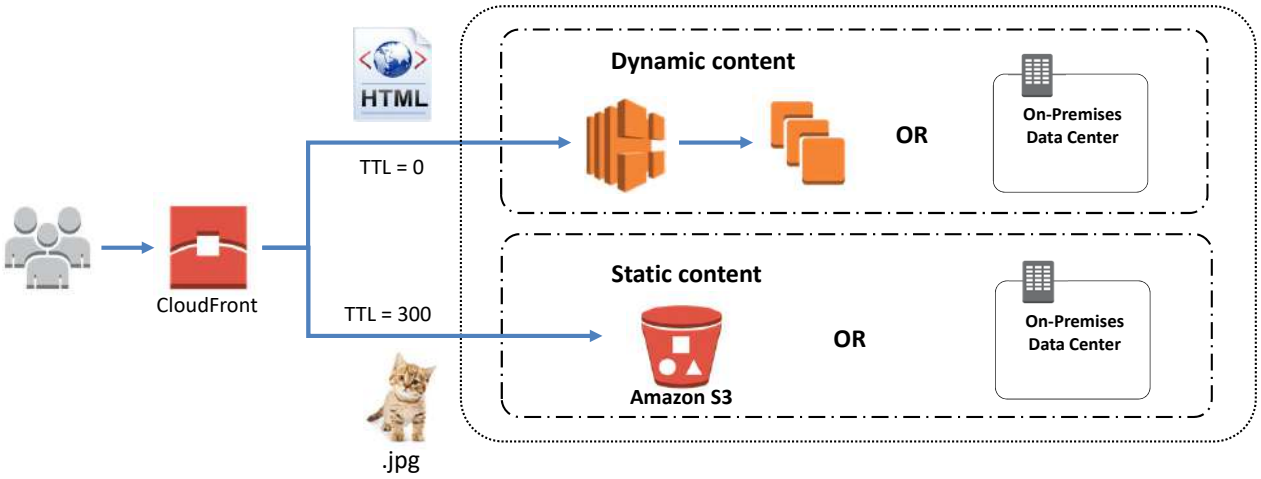
For instance, in the anti-pattern that is shown on the slide, your Amazon S3 bucket does not use a caching service. Three users request a file from one of your Amazon S3 buckets, one at a time. The file is delivered to each user in the same way. The result is that each request takes the same amount of time to complete. This process also incurs costs for the three separate times that the file is delivered for each request.

Let’s compare the process in the anti-pattern with a better pattern. In the best practice pattern, your infrastructure places Amazon CloudFront—which offers caching—in front of Amazon S3. In this scenario, the first request checks for the file in CloudFront. If the request does not find the file in CloudFront, it pulls the file from Amazon S3, and stores a copy of the file in CloudFront at the edge location that is closest to the user. The request then sends a copy of the file to the user who made the request. Now, when any other users request that file, it's

retrieved from the closer edge location in CloudFront. The request does not have to go to Amazon S3 to get the file.

The best practice pattern reduces both latency and cost. After the first request is complete, you no longer pay for the file to be transferred out of Amazon S3.

Cache Static and Reusable Content

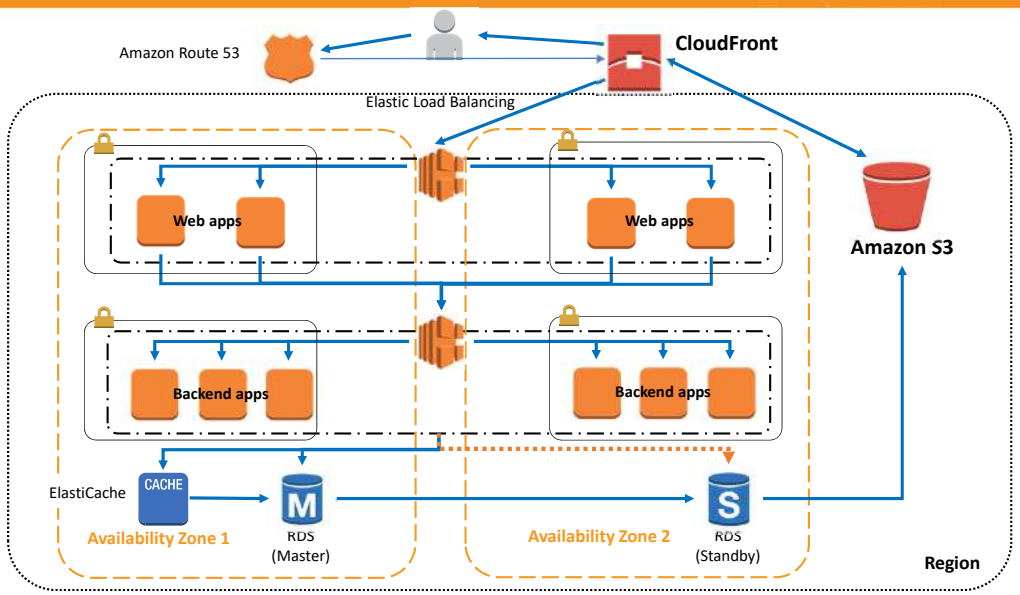


This slide shows another way to use Amazon CloudFront. In general, you only cache static content. However, dynamic or unique content affects the performance of your application. Depending on the demand, you might still get some performance gain by caching the dynamic or unique content in Amazon S3.

For example, if there's something that needs to be personalized, the time to live is zero. A time to live of zero tells CloudFront that each and every time it sees the dynamic content, it needs to go back to its origin because it will change a lot.

At the same time, the time to live can be set to 5 minutes or 24 hours, depending on how long the content is good for. CloudFront can pull content from Amazon S3 so that it can save the load back to an on-premises data center. When you save a load, you can have smaller instances and save money, and resources can perform more efficiently.

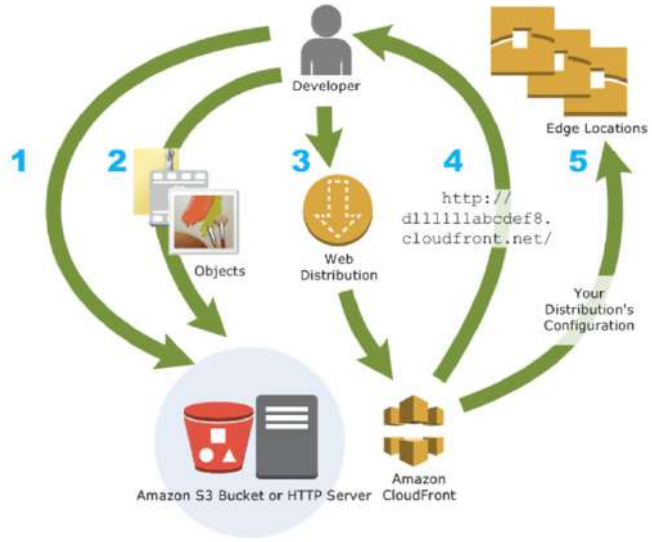
AWS Cloud Architecture: Web Hosting



Let's take a look at cloud architecture for web hosting. We have our Elastic Load Balancing load balancers in place with CloudFront. End users are directed to CloudFront via Amazon Route 53. The load balancers pull content and data from the Amazon S3 bucket, Amazon RDS, or Amazon ElastiCache. This can serve as a read replica if content is cached there.

This architecture diagram shows how you can use CloudFront in front of your hosting architecture to decrease the number of times CloudFront must redirect requests to the load balancer, and pull content.

How to Enable CloudFront?



How can you enable Amazon CloudFront?

First, you specify origin servers, like an Amazon S3 bucket or your own HTTP server. CloudFront gets your files from the origin servers, and the files will then be distributed from CloudFront edge locations all over the world.

Second, you upload your files to your origin servers. Your files, which are also known as objects, typically include webpages, images, and media files.

Third, you create a CloudFront distribution, which tells CloudFront which origin servers to get your files from when users request the files through your website or application. At the same time, you specify details, such as whether you want CloudFront to log all requests, and whether you want the distribution to be enabled as soon as it's created.

Fourth, CloudFront assigns a domain name to your new distribution. You can see the domain name in the CloudFront console. The domain name can also be returned in the response to a programmatic request, like a request from an application programming interface, or API.

Fifth, CloudFront sends your distribution's configuration—but not your content—to all of its edge locations. -collections of servers in geographically dispersed data centers where CloudFront caches copies of your objects.

CloudFront Enablement Options

Use a separate CNAME for static content.

- Static content cached, dynamic content straight from origin.
- Most efficient.
- More effort to set up and manage.

Point entire URL to CloudFront.

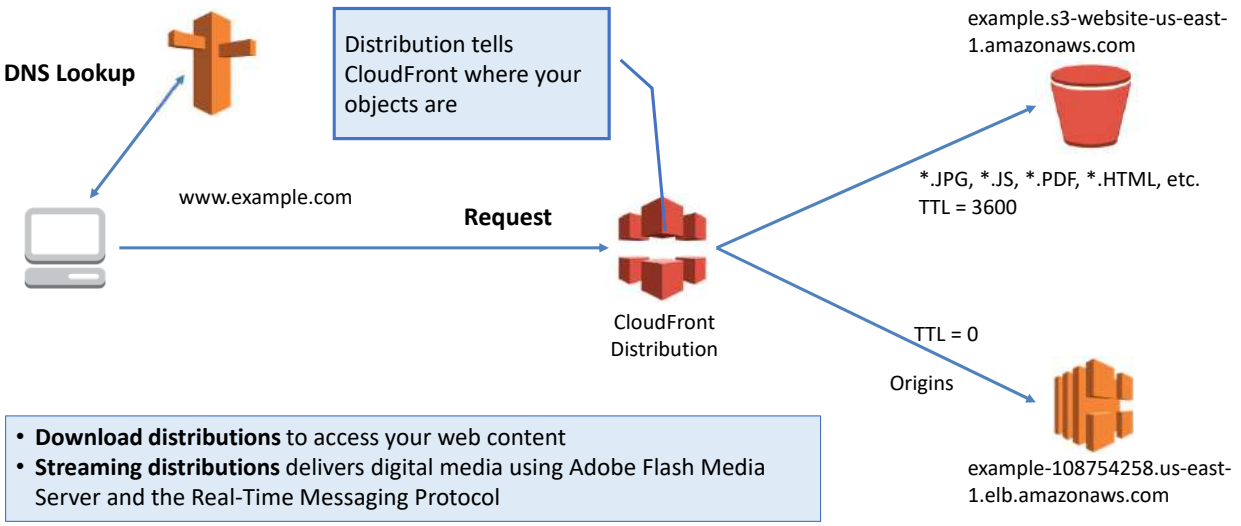
- Easiest to manage.
- Use URL patterns to stage dynamic content.
- ALL content goes through edge locations.

Amazon CloudFront has several options for enablement.

If you choose to use a separate Canonical Name Record—or CNAME—for static content, the static content is cached straight from the origin server. This process is the most efficient, but it takes more effort to set up and manage.

If you point the entire uniform resource locator—or URL—to CloudFront, it’s easier to manage. You can use URL patterns to stage dynamic content. All of the content goes through edge locations.

CloudFront Distributions



- **Download distributions** to access your web content
- **Streaming distributions** delivers digital media using Adobe Flash Media Server and the Real-Time Messaging Protocol

Now let's look at how the CloudFront distributions work. In this example, Amazon Route 53 resolves example.com on behalf of the client. A request is made to the CloudFront distribution. CloudFront looks at the Amazon S3 bucket for the content that it identifies as being stored statically. For any content that has a time to live of zero, CloudFront will go directly to the Elastic Load Balancing load balancer because it has to go back to the origin server to pull the content. In this way, CloudFront delivers both static and dynamic content.

There are two distribution types:

- Web distribution, which lets you access your web content in any combination of up to 10 Amazon S3 buckets or custom origin servers.
- And Real Time Messaging Protocol—or RTMP—distribution, which is always an Amazon S3 bucket.

CloudFront Speeds Up a Website

Use **cache control headers**.

- Cache control header set on your files identifies **static** and **dynamic** content.
- Delivering all your content using a single Amazon CloudFront distribution helps to ensure performance optimization on your entire website.

How does CloudFront speed up a website? Amazon CloudFront reads cache control headers to determine how frequently it needs to check the origin server for an updated version of that file. For an expiration period set to 0 seconds, Amazon CloudFront will revalidate every request with the origin server.

The cache control header that is set on your files identifies both static and dynamic content. The cache control header can even have custom headers within the CloudFront distribution graphical user interface—or GUI— within the console.

Delivering all your content by using a single Amazon CloudFront distribution helps to ensure performance optimization on your entire website.

Expiration Period

The expiration period is set by you.

- If your files don’t change often, set a long expiration period.

How long is a file kept at the edge location?

- Set **expiration period** by setting the cache control headers on your files in your origin.

What about an expiration period? You can set one. If your files don’t change very often, the best practice is to set a long expiration period, and implement a versioning system to manage updates to your files.

By default, if no cache control header is set, each edge location checks for an updated version of your file when it receives a request more than 24 hours after the previous time it checked the origin server for changes to that file.

How long is a file kept at the edge location?
You can set the expiration period by setting the cache control headers on your files in your origin server.

How to Expire Contents

Time to live (TTL)

- Fixed period of time (expiration period).
- Time period is set by you.
- GET request to origin from CloudFront will use **`If-Modified-Since header`**.

Change object name

- **`Header-v1.jpg becomes Header-v2.jpg`**.
- New name forces refresh.

Invalidate object

- Last resort: very inefficient and very expensive.

Learn more.

How can you set content to expire? There are three ways to retire cached content:
- Use time to live, or TTL
- Change the object name
- Or invalidate the object

The most preferred options are to use time to live and to change the object name. Using time to live is the easiest option if the replacement does not need to be immediate.

If you set the time to live for a particular origin server to 0, CloudFront will still cache the content from that origin server. It will then make a GET request with an If-Modified-Since header. This header allows the origin server to signal that CloudFront can continue to use the cached content if the content has not changed at the origin server.

Changing the object name requires more effort, but its effects are immediate. There might be some support for this option in some content management systems, or CMSs. Although you can update existing objects in a CloudFront distribution and use the same object names, it is not recommended. CloudFront distributes objects to edge locations only when the objects are requested, not when you put new or updated objects in your origin server. If you update an existing object in your origin server with a newer version that has the same name, an edge location won't get that new version from your origin server until the object is updated and requested.

Invalidating an object should be used sparingly for individual objects. It is a bad solution because the system must forcibly interact with all edge locations.

To learn more, go to the following page:
http://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/RequestAndResponseBehaviorS3Origin.html