

THE ORION PAPERS PRO

AWS Certified Solutions Architect - Professional
Course Manual

ENTER

ADRIAN CANTRILL, TRAINING ARCHITECT



Linux Academy



Linux Academy

Dear students,

Welcome to Linux Academy's updated AWS Certified Solutions Architect - Professional exam prep course! This course will help you prepare for the AWS Certified Solutions Architect - Professional exam.

To help you on your journey, we've provided an interactive learning tool known as *The Orion Papers*. This resource is meant to be a non-linear, interactive guide to the course material. You can use it to enhance your learning experience and learn about the required AWS services at your own pace. This interactive guide can be used independently of the video lessons, but we are also going to use it in conjunction with them. Each hands-on lab will also provide a static chart for you to reference.

This course contains a variety of labs that give you hands-on exposure to AWS in live environments. You won't just learn the material; you'll get to practice it until you're confident in your skills! Successfully completing this course and practicing the hands-on labs will put you in a great position for taking the certification exam.

We hope you enjoy the course.

Now let's get certified!

Adrian Cantrill

Craig Arcuri



Global Infrastructure

Account & Services Layer

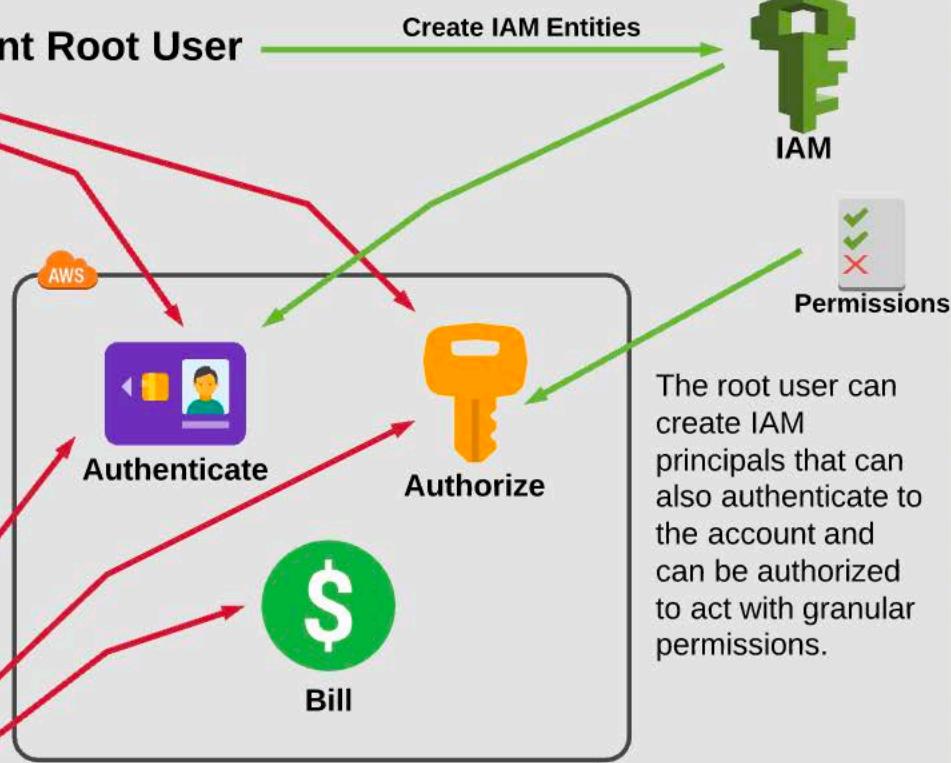
Physical & Networking Layer

Regions, AZs,
& Edge

Account Root User

The root user is initially the only principal that can authenticate to the account and the only principal authorized to do so.

The root user is specific to an account.



Account Domains

AWS accounts consist of three discrete domains: Authentication, Authorization, and Billing. By default, every AWS account has separate billing, users, and permissions. If an account is exploited, the "blast radius" is limited to that specific account.



JSON (JavaScript Object Notation)

Key/Value Pairs:

```
{ "Name" : "LinuxAcademy",
  "Type" : "Company" }
```

Lists of Things:

```
[ 1, 2, 3]
["Penny", "Roffle", "Winkie", "Truffles"]
```

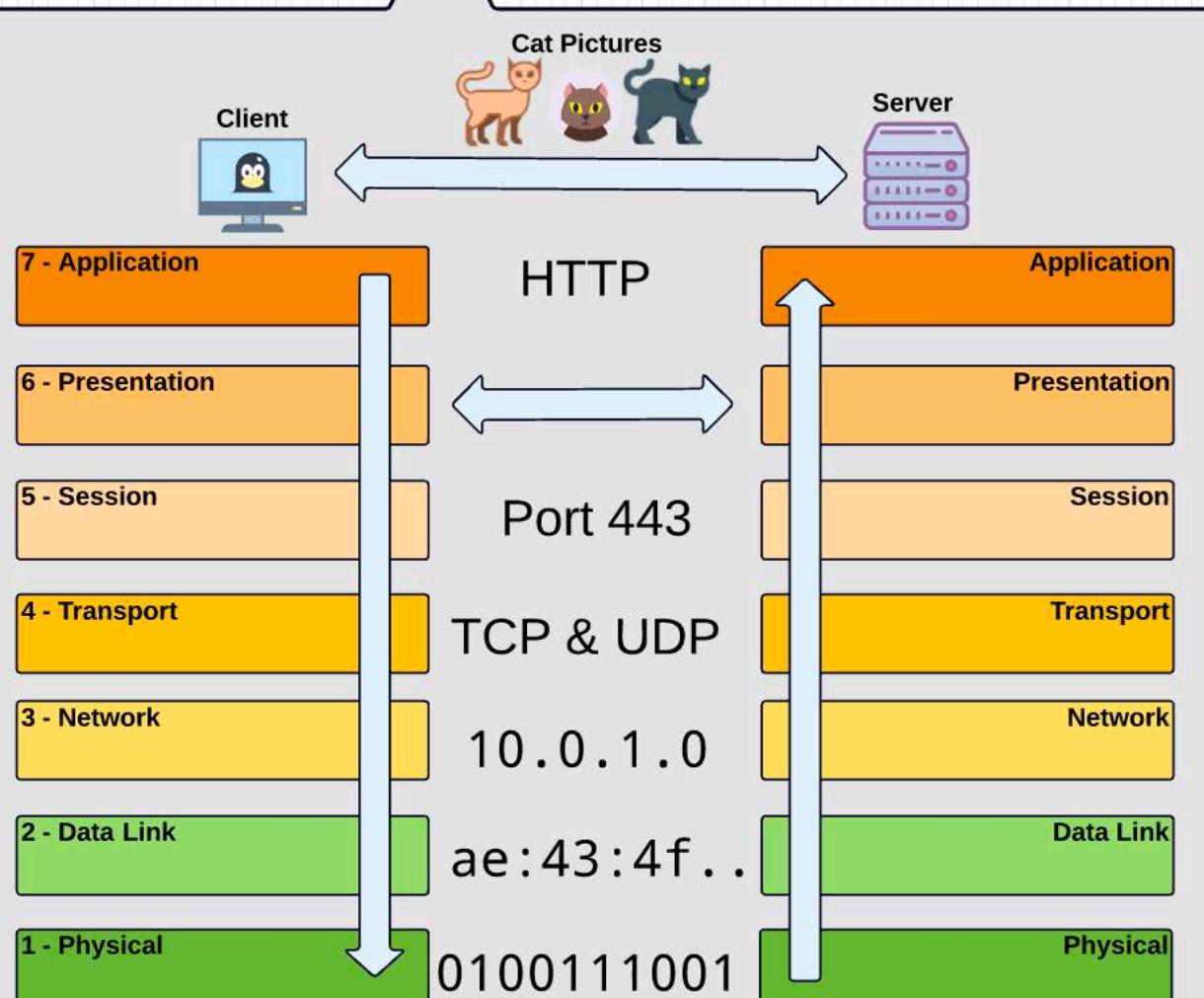
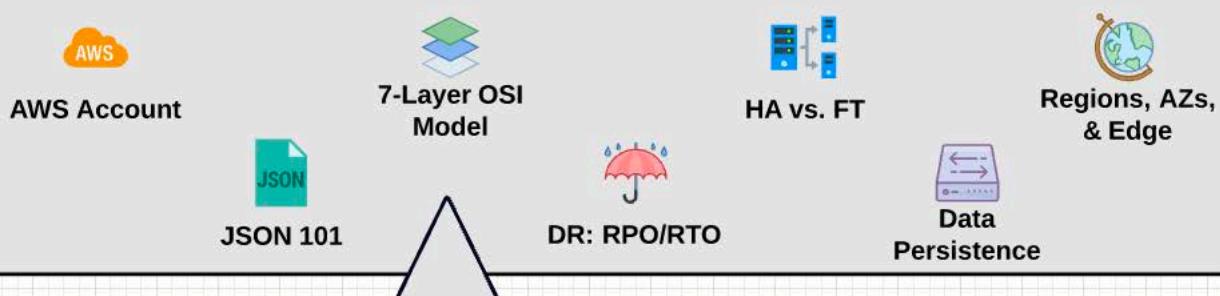
Key/Value Pairs & Lists:

```
{ "Name" : "Adrian",
  "Pets" : ["Penny", "Roffle", "Winkie", "Truffles"] }
```

Nesting and Complex Structures:

```
{"Team" : [ {"Name" : "Tom"}, {"Name", "Tia"}, {"Name" :
"Adrian", "Pets" : ["Winkie", "Roffle", ... ] } ] }
```

```
{"Team" : [ {"Name" : "Tom"}, {"Name", "Tia"}, {"Name" :
"Adrian", "Pets" : ["Winkie", "Roffle", ... ] } ] }
```



The OSI 7-layer networking model splits network communication into isolated layers. The protocols at any given layer can be changed or updated without affecting any of the other layers. Communication can occur between hosts as long as both hosts are using the same protocol at the same layer.



Global Infrastructure

Account & Services Layer

Physical & Networking Layer



Everything is fine.
It's 6:00 PM Monday.



Nightly Backup
It's 11:00 PM Monday.



RPO

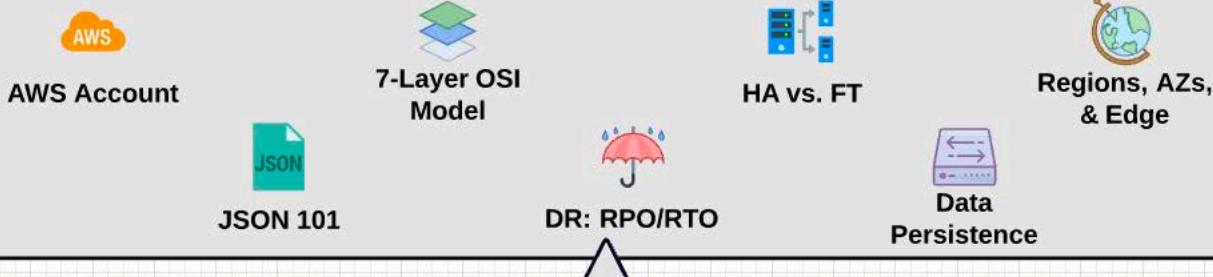
Oh, no!! It's broken!!
It's now 3:00 AM Tuesday.



Everything is "fine" again.
It's 10:00 AM Tuesday.



RTO



Everything is fine.
It's 6:00 PM Monday.



Nightly Backup

It's 11:00 PM Monday.



RPO: Recovery Point Objective

The time between when a disaster occurs and the last recoverable copy of key business data was created.

Minimize the length of this time period (via regular backups, snapshots, and transaction logs) to avoid business disruptive data loss.



Oh, no!! It's broken!!

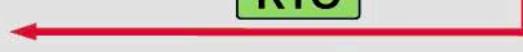
It's now 3:00 AM Tuesday.



Everything is "fine" again.
It's 10:00 AM Tuesday.



RTO





Global Infrastructure

Account & Services Layer

Physical & Networking Layer



Regions, AZs, & Edge



Everything is fine.
It's 6:00 PM Monday.



Nightly Backup
It's 11:00 PM Monday.



RTO: Recovery Time Objective

The time between when a disaster occurs and when the system can be restored to an operational state and handed over to the business for testing.

Improve this by decreasing restore time, having hot spares tested and ready to use, and enforcing efficient processes.



Everything is "fine" again.
It's 10:00 AM Tuesday.



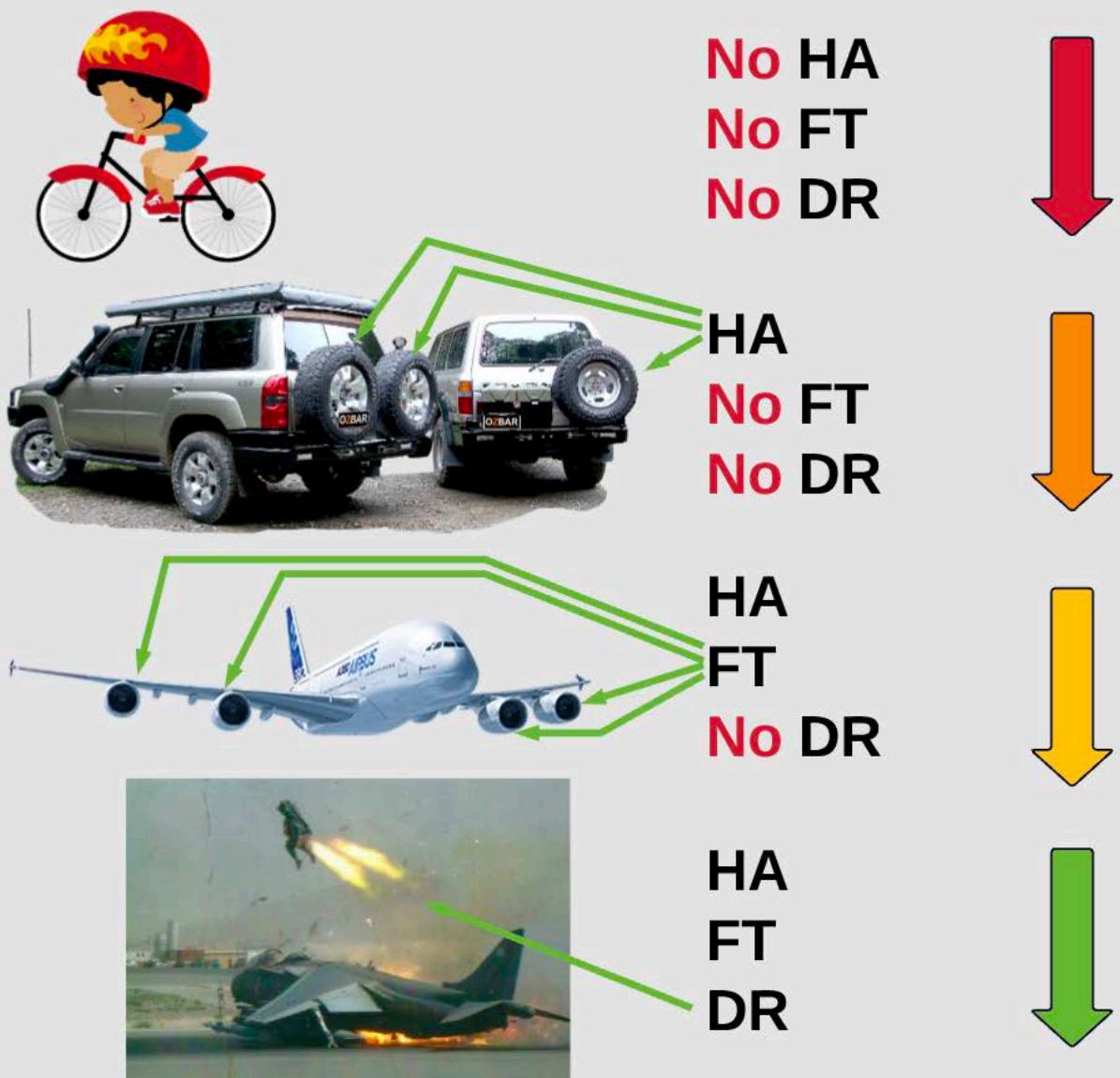
Oh, no!! It's broken!!
It's now 3:00 AM Tuesday.



RPO



RTO

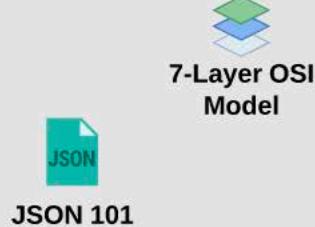




Global Infrastructure

Account & Services Layer

Physical & Networking Layer



Ephemeral

Data that is generally local to a resource and is lost when that resource is powered down



Instance
Store
Volume



Amazon
ElastiCache

Transient

Data that exists in a form while it's passed between sources



Queue

Persistent

Data that is durable and survives power events (Start, Stop, Restart)



Amazon
EBS



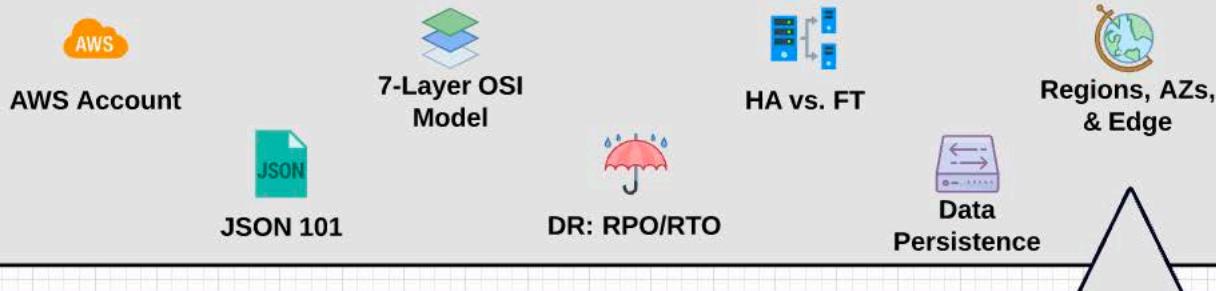
Amazon
EFS



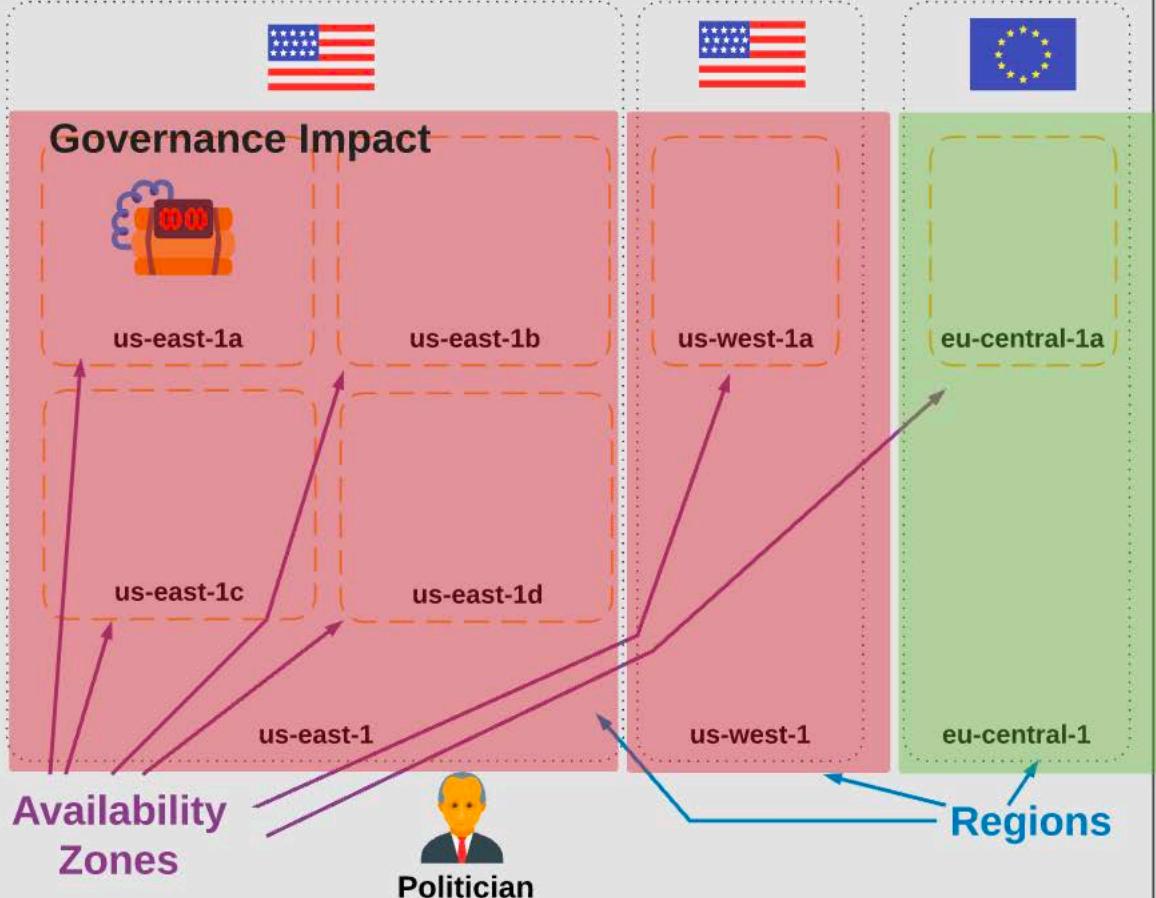
Global Infrastructure

Account & Services Layer

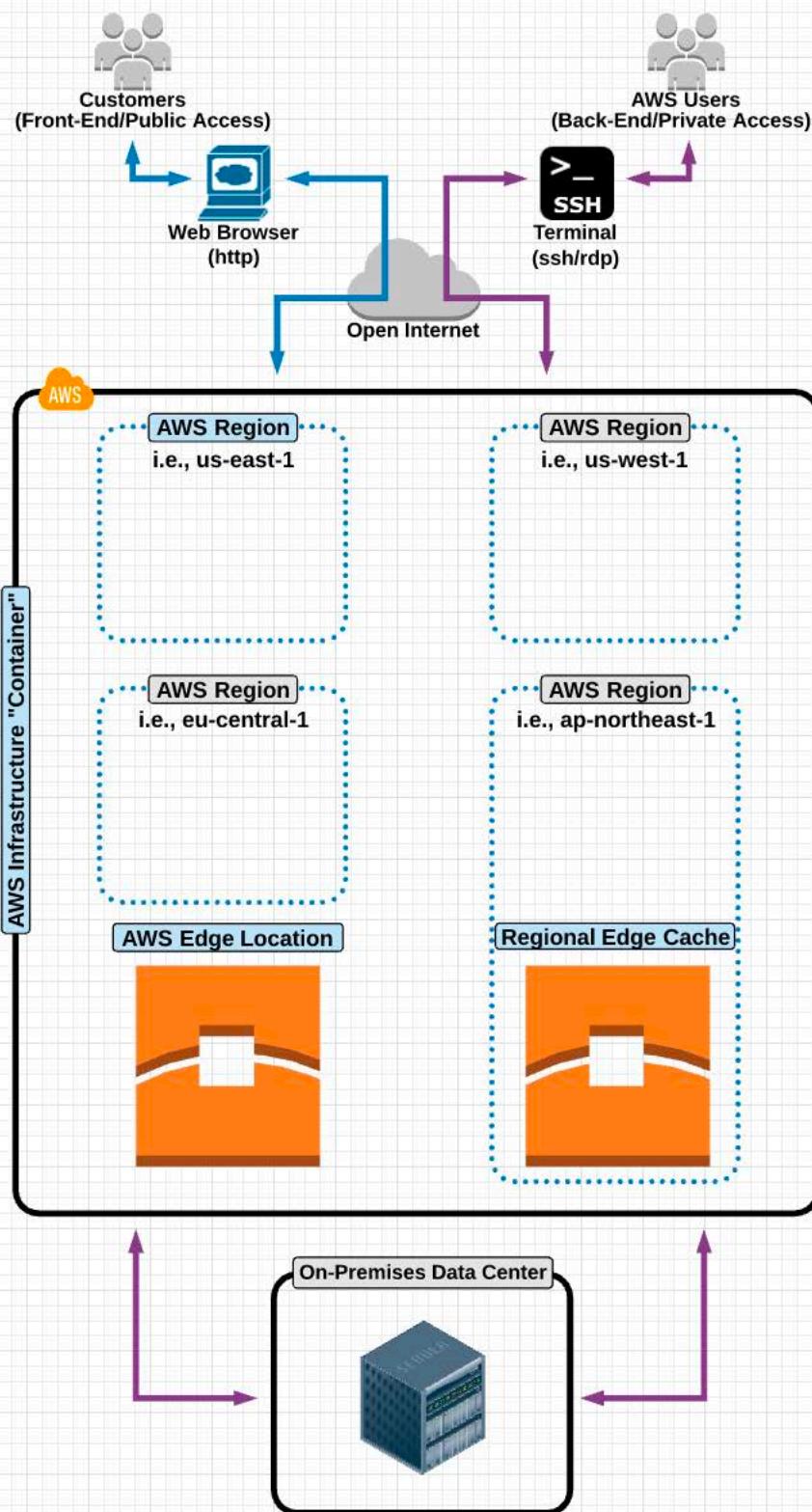
Physical & Networking Layer

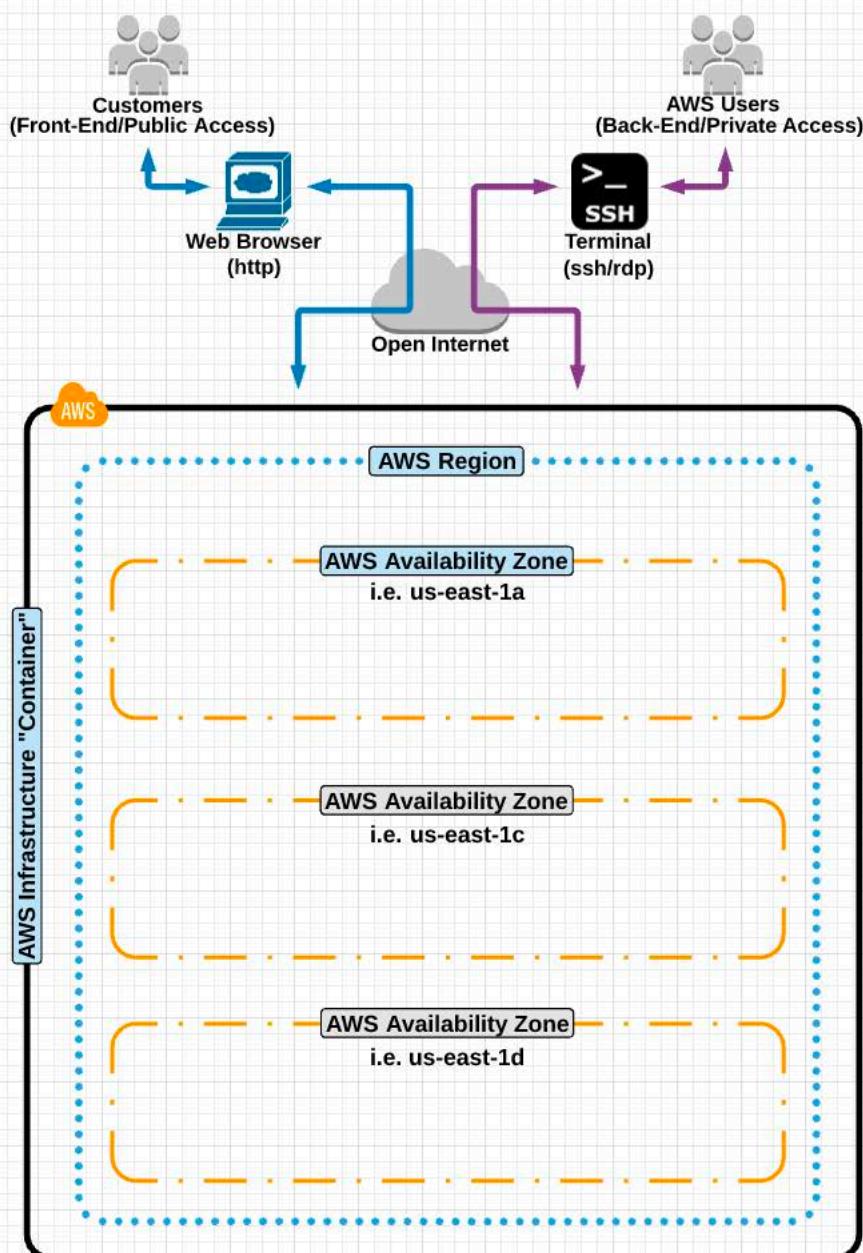


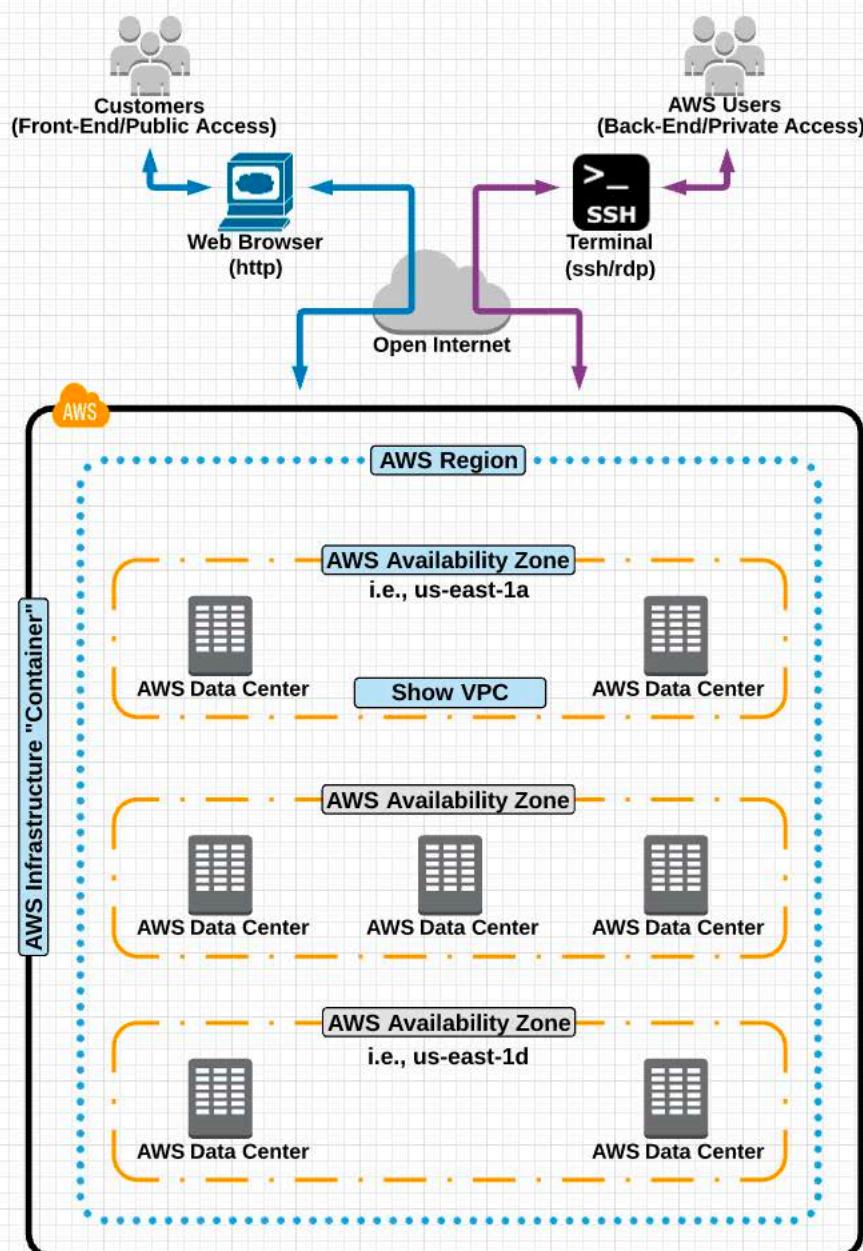
AWS



Edge locations are smaller pockets of infrastructure, distributed globally in major cities. These locations are used to distribute content and operate other "edge" services.



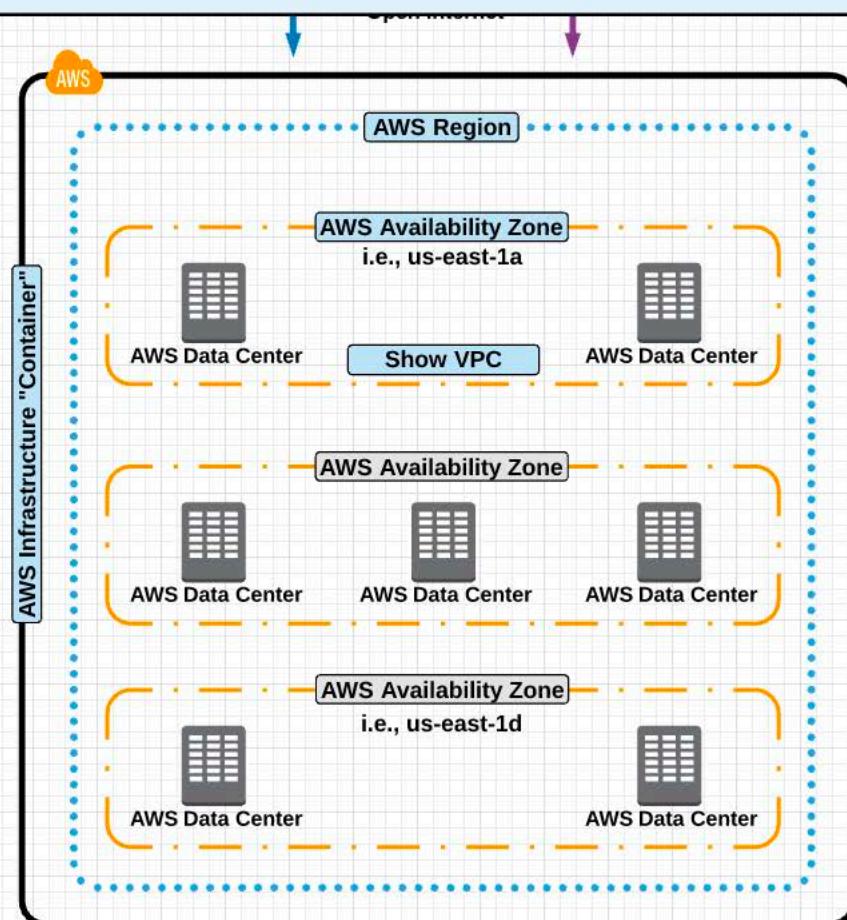






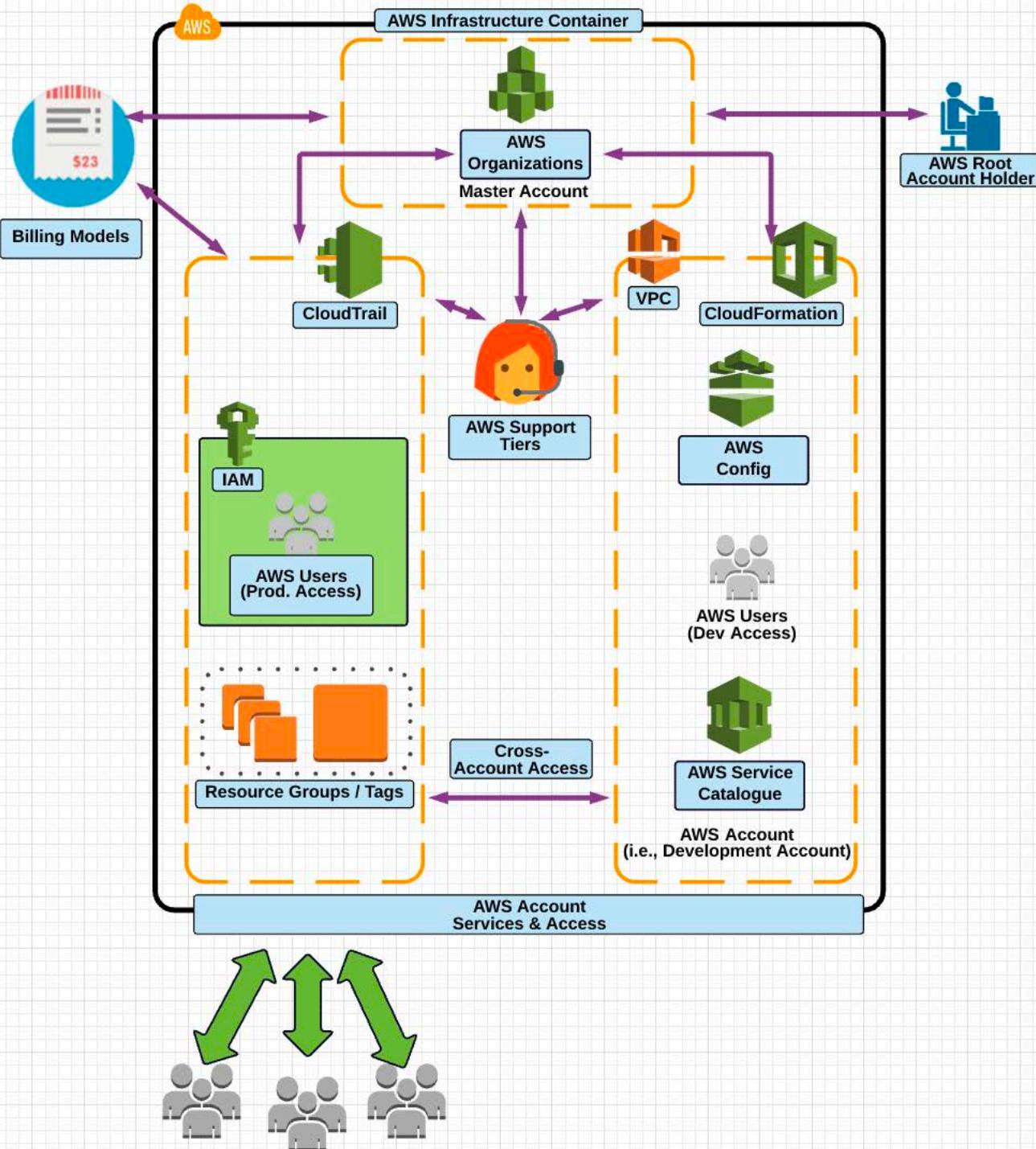
AWS Infrastructure Container

- This represents the boundaries of AWS.
- Everything inside is part of AWS's infrastructure, including all of its physical networking components and services.
- Everything outside represents items that are external to AWS, which either connect to AWS or belong to you or your company (e.g., on-premises servers, the open internet, or your personal computer).





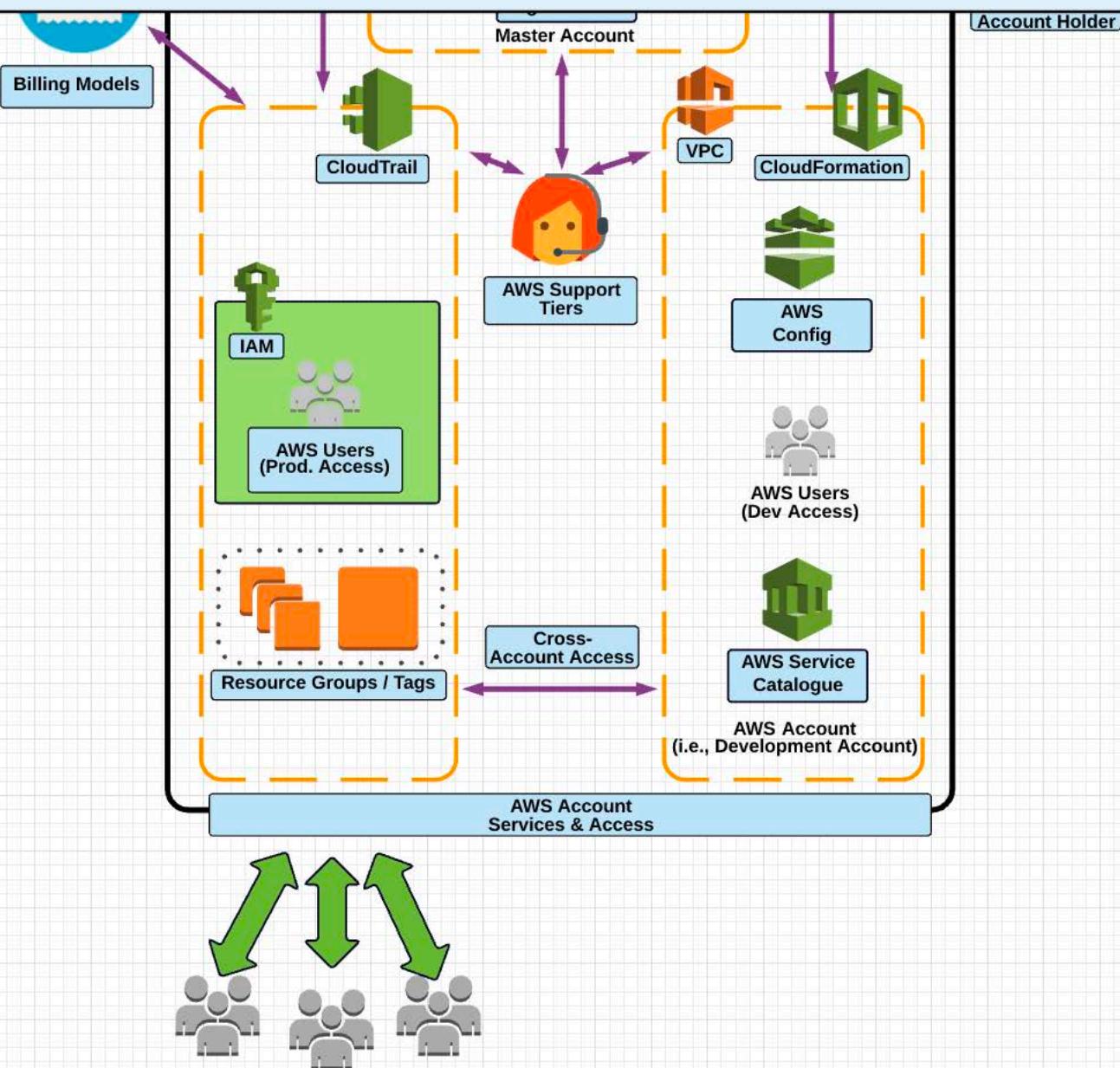
The Account and Services layer represents how we create, access, and manage an AWS account and its services, from interacting with an AWS account and managing user rights, to accessing and using various AWS services and features.





AWS Infrastructure Container

- The AWS infrastructure container represents the boundaries of AWS.
- Everything inside is part of AWS's infrastructure, including all of its physical networking components and services.
- Everything outside represents items that are external to AWS, that either connect to AWS or belong to you or your company (i.e., on-premises servers, the internet, or your personal computer).



[Close](#)

AWS Organizations

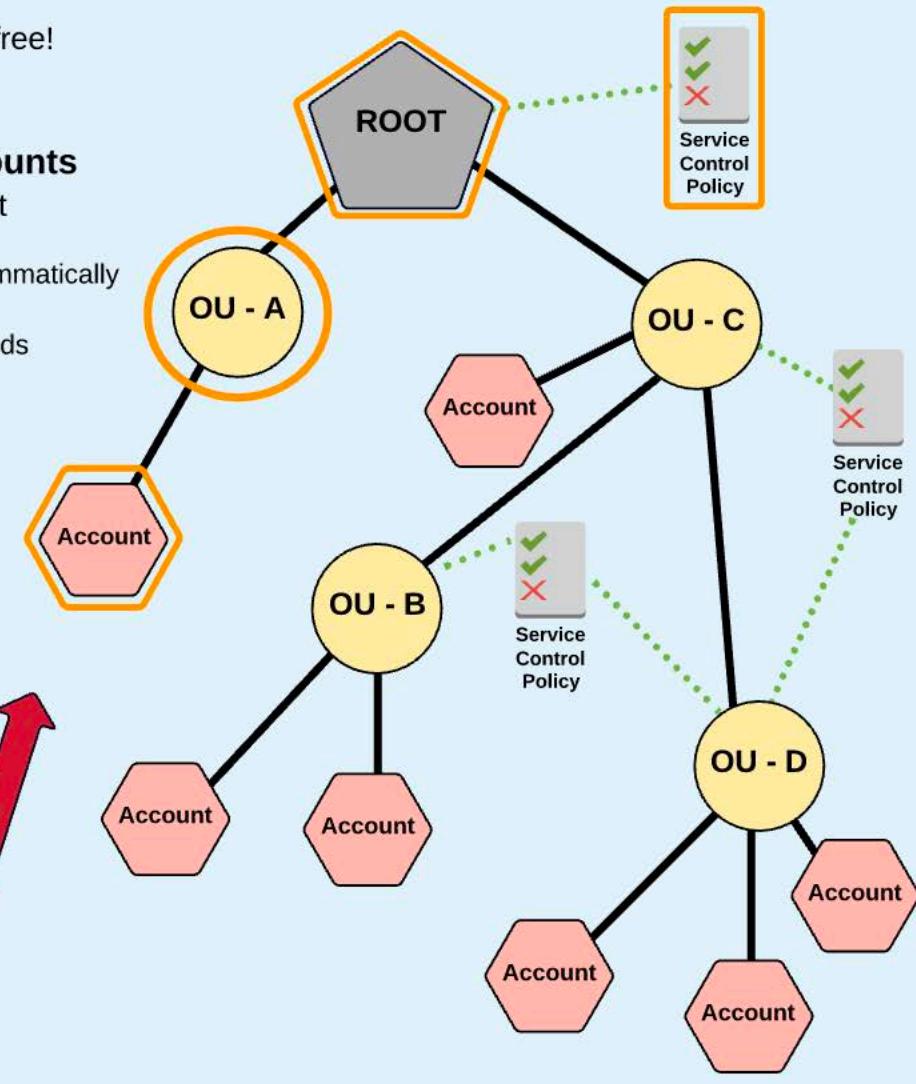
AWS Organizations is a service that lets you consolidate multiple AWS accounts into a single organization. This allows you to manage all accounts within the organization in one place.

AWS Organizations makes billing and permissions easier and allows for the creation of managed accounts. Accounts are organized hierarchically, which provides better security and compliance controls.

Best of all, this service is free!

Manage Multiple Accounts

- IAM policy management
 - Account groups
 - Create accounts programmatically
- Consolidated billing
 - Manage payment methods



Organizations operate in either "Consolidated Billing" or "All Features" mode. This can be modified at any time.

[Consolidated Billing](#)[All Features](#)

[Close](#)

AWS Organizations

AWS Organizations is a service that lets you consolidate multiple AWS accounts into a single organization. This allows you to manage all accounts within the organization in one place.

AWS Organizations makes billing and permissions easier and allows for the creation of managed accounts. Accounts are organized hierarchically, which provides better security and compliance controls.

Best of all, this service is free!

Manage Multiple Accounts

- IAM policy management
 - Account groups
 - Create accounts program
- Consolidated billing
 - Manage payment method

Consolidated billing is a feature in Organizations that allows the master account to pay for all charges incurred by the member accounts.

You get one bill for *all* accounts.

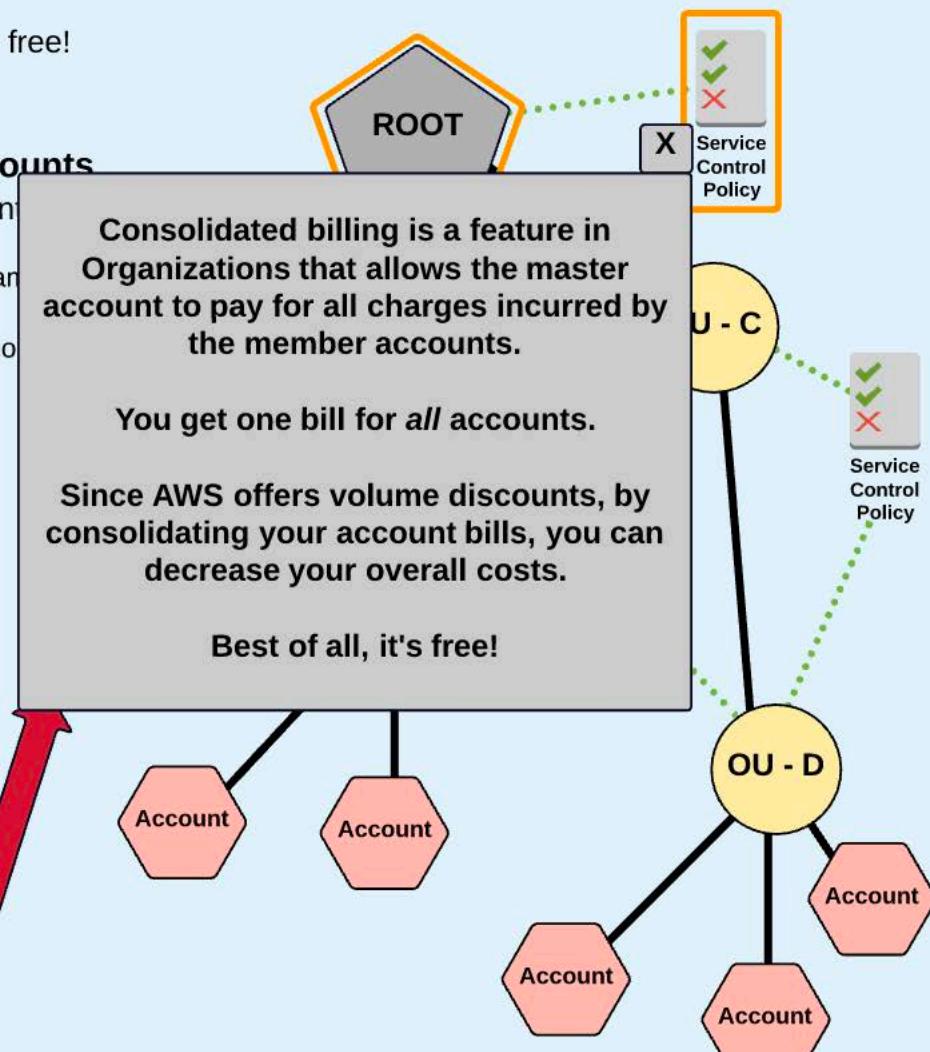
Since AWS offers volume discounts, by consolidating your account bills, you can decrease your overall costs.

Best of all, it's free!

Organizations operate in either "Consolidated Billing" or "All Features" mode. This can be modified at any time.

[Consolidated Billing](#)

[All Features](#)





Close

AWS Organizations

AWS Organizations is a service that lets you consolidate multiple AWS accounts into a single organization. This allows you to manage all accounts within the organization in one place.

AWS Organizations makes billing and permissions easier and allows for the creation of managed accounts. Accounts are organized hierarchically, which provides better security and compliance controls.

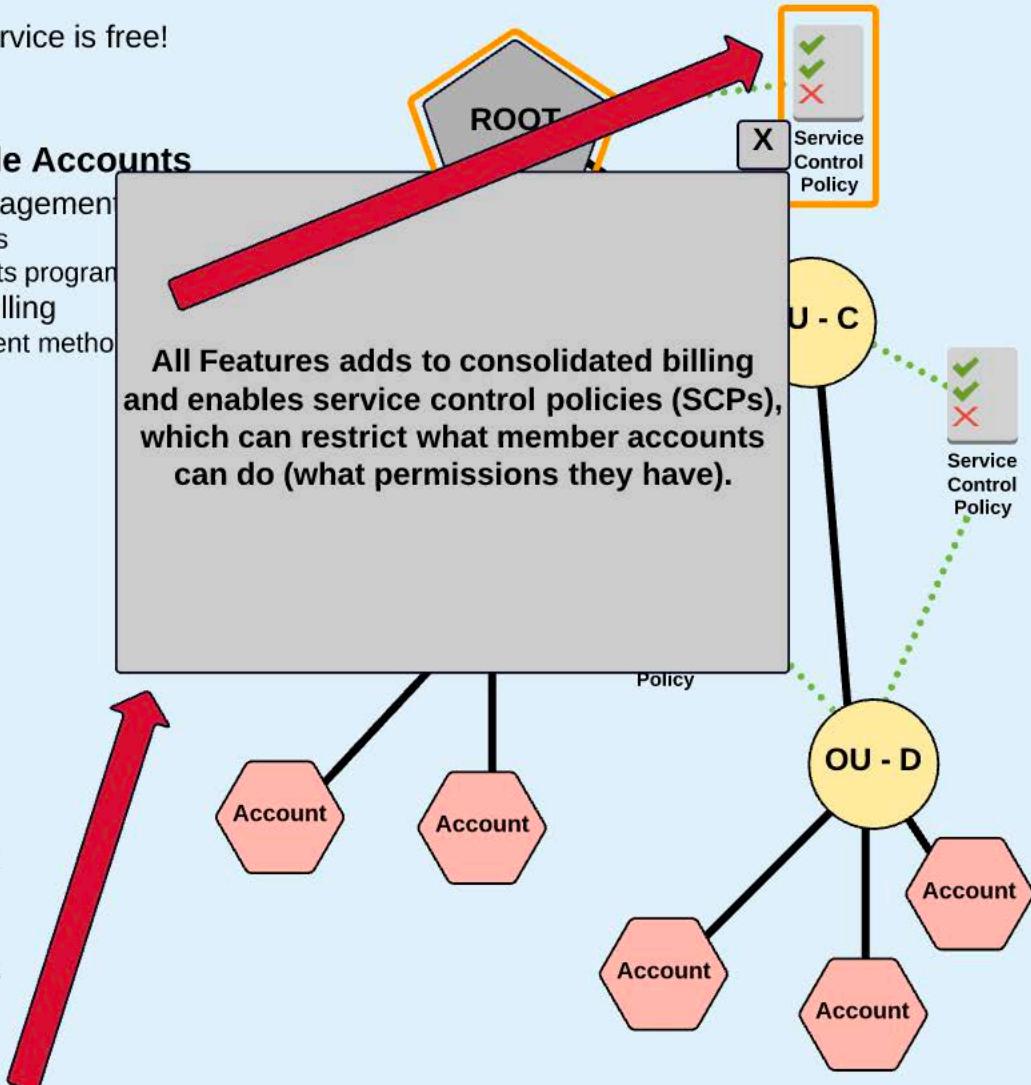
Best of all, this service is free!

Manage Multiple Accounts

- IAM policy management
 - Account groups
 - Create accounts program
- Consolidated billing
 - Manage payment method

All Features adds to consolidated billing and enables service control policies (SCPs), which can restrict what member accounts can do (what permissions they have).

Organizations operate in either "Consolidated Billing" or "All Features" mode. This can be modified at any time.



Consolidated Billing

All Features



CloudTrail

CloudTrail is the primary service we use for logging in AWS.

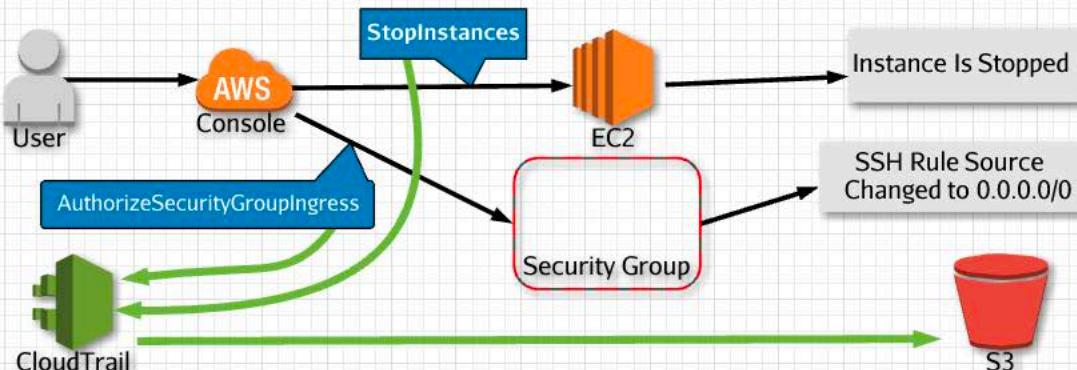
Important Features:

- It logs all the API calls in an AWS account (includes Console, CLI, API/SDK calls)
- It's enabled when your account is created
- Entries can be viewed using the **Event History** (past 90 days)
- **Trail:** A configuration allowing for logs to be sent to an S3 bucket:
 - Single region or multi-region trails can be configured
 - Trails can make multi-account logging possible (more on this later)

Trails have several configuration options:

- **Management events:** Enabling will log control plane events, such as:
 - User login events
 - Configuring security
 - Setting up logging
- **Data events**, which include:
 - Object-level events in S3
 - Function-level events in Lambda
- **Encryption flexibility:**
 - Encrypted in S3 server-side by default but can be changed to KMS
 - The logs can be sent to an **S3 bucket** of choice and even prefixed (folders)

Scenario: An IAM user logs in to the AWS Management Console. That user then proceeds to stop an EC2 instance and edit a security group. CloudTrail will log all of these actions. Here is the workflow:





AWS Support Tiers

[Close](#)

AWS Support provides three different paid support tiers: Developer, Business, and Enterprise. Each of these tiers has a different cost structure and provides a different feature set, which for a Solutions Architect, is important to understand at a high level.

	Developer	Business	Enterprise
	<i>Recommended if you are experimenting or testing in AWS.</i>	<i>Recommended if you have production workloads in AWS.</i>	<i>Recommended if you have business and/or mission critical workloads in AWS.</i>
AWS Trusted Advisor Best Practice Checks	7 Core checks	Full set of checks	Full set of checks
Enhanced Technical Support	Business hour email access to Support Engineers Unlimited cases / 1 primary contact	24x7 phone, email, and chat access to Support Engineers Unlimited cases / unlimited contacts (IAM supported)	24x7 phone, email, and chat access to Support Engineers Unlimited cases / unlimited contacts (IAM supported)
Architectural Guidance	General	Contextual to your use-cases	Consultative review and guidance based on your applications
Programmatic Case Management		AWS Support API	AWS Support API
Third-Party Software Support		Interoperability & configuration guidance and troubleshooting	Interoperability & configuration guidance and troubleshooting
Proactive Programs		Access to Infrastructure Event Management for additional fee.	Infrastructure Event Management Well-Architected Reviews Operations Reviews Technical Account Manager (TAM) coordinates access to programs and other AWS experts as needed.
Technical Account Management			Designated Technical Account Manager (TAM) to proactively monitor your environment and assist with optimization.
Training			Access to online self-paced labs
Account Assistance			Concierge Support Team
Pricing	Starts at \$29 / month. See pricing detail and example.	Starts at \$100 / month. See pricing detail and example.	Starts at \$15K / month. See pricing detail and example.

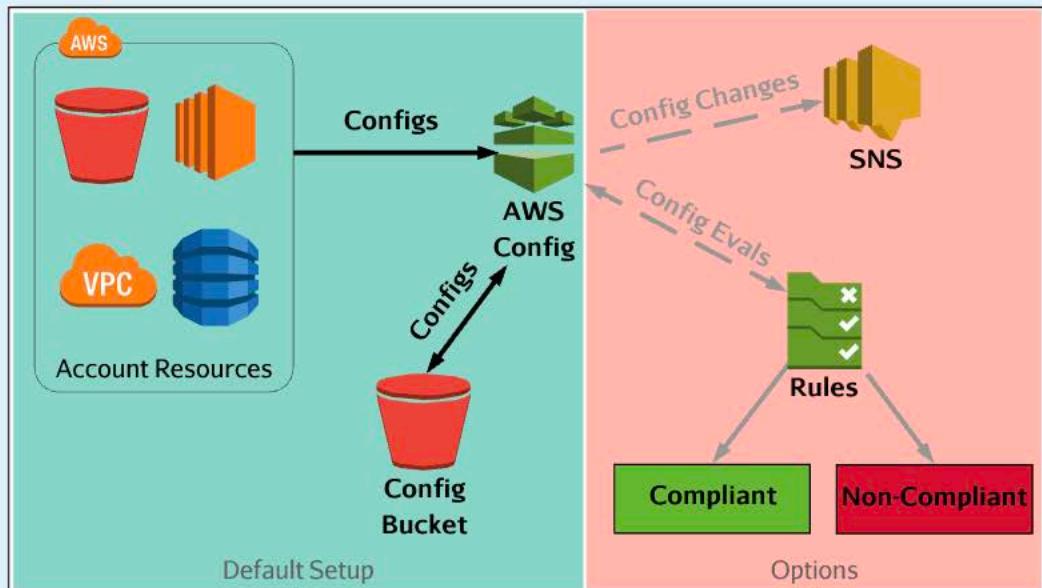
[Close](#)

AWS Config

A detailed view of the configuration of AWS resources (EC2, EBS, security groups, VPC, etc.)

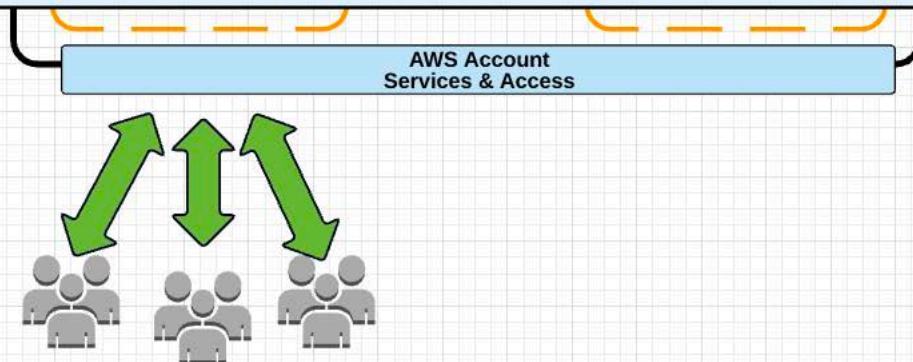
With AWS Config, you can:

- Evaluate resource configurations for desired settings
- Get a snapshot of the current configurations associated with your account
- Retrieve the current or historical configurations of resources in your account
- Receive notification of creations, deletions, and modifications
- View the relationships between resources (e.g., members of a security group)



Uses of AWS Config:

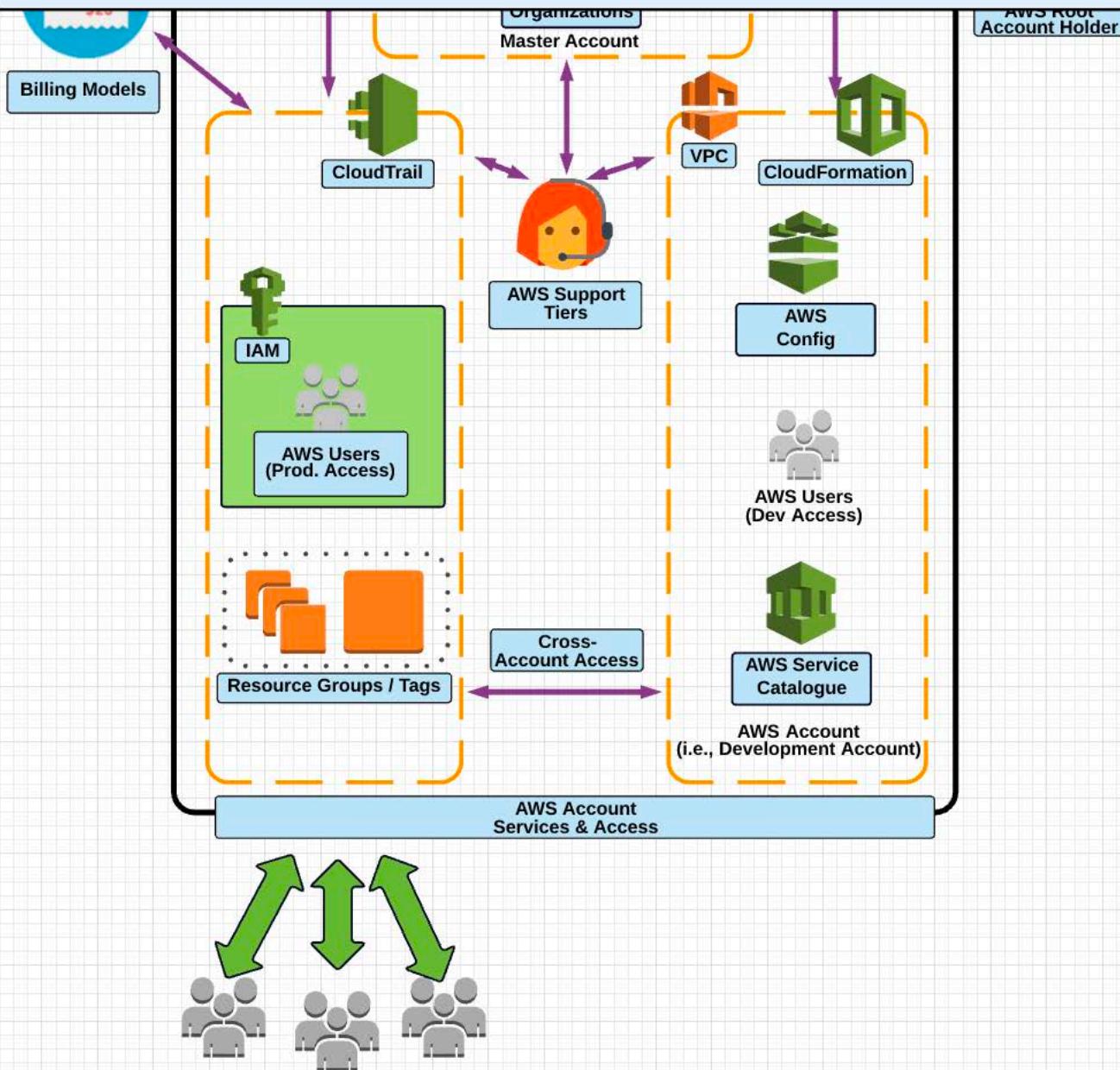
- Can be used to identify usage/changes in config over time (current and historical)
- Helps with architecture planning: shows relationships between products and services
- Can generate events related to changes
- Can help keep an implementation in a compliant state
- Assists security teams by showing changes to SGs and NACLs over time
- Fault finding and investigation: last known good config





AWS Users

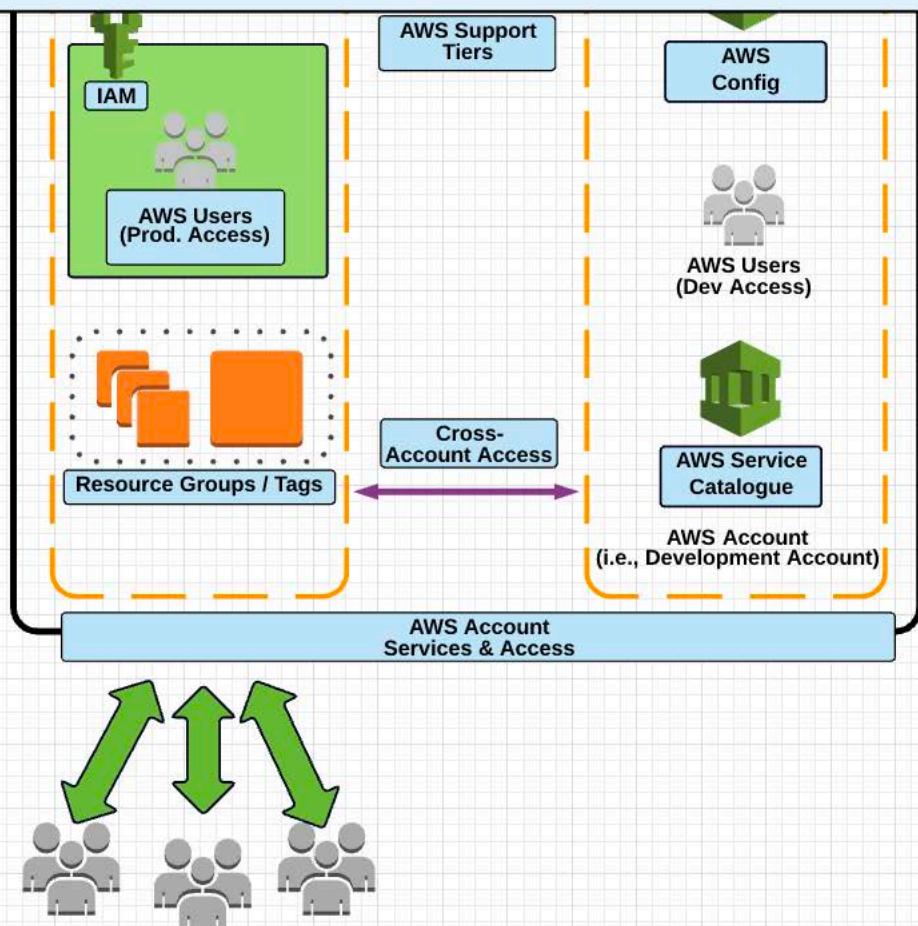
- AWS users are users that you can create (in IAM) who have varying degrees of access (see the example below).
- Permissions are controlled with IAM policies.





Cross-Account Access

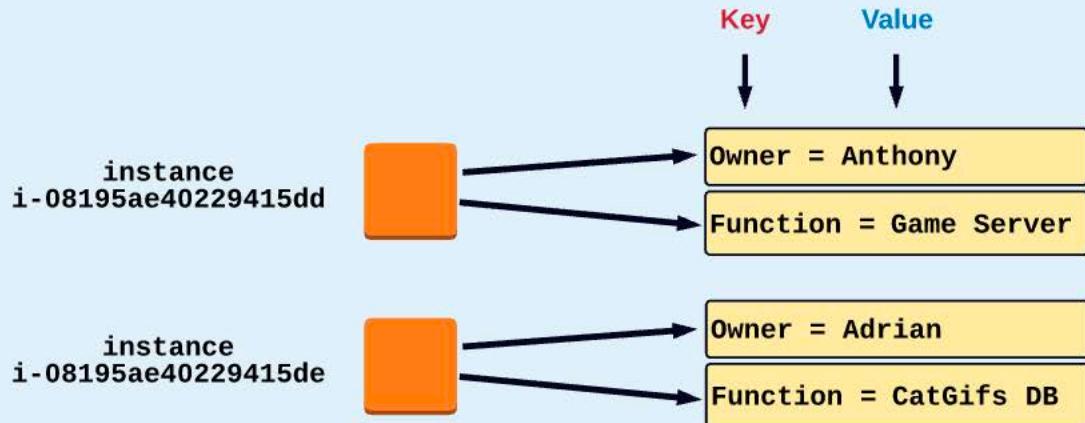
- **Cross-account access** allows IAM users in one account to access resources that are in different AWS accounts that you own (i.e., separate Production and Development accounts; see diagram below).
- Cross-account access is handled through the use of **IAM roles**.
- Users in one account assume a role that grants access to resources in another.
- The benefits of using roles for cross-account access include:
 - No need to create individual IAM users in each account
 - Users don't have to sign out of one account and into another in order to access resources that are in different AWS accounts



[Close](#)

Tags and Resource Groups

In AWS, tags are key and value pairs that can be assigned to resources. A tag must contain a key and can optionally have an associated value.



Tags can be used in the UI console under Resource Groups. Both the UI and CLI/API allow you to search and filter by tags. You can also use tags for cost allocation reporting — this has to be enabled globally for your account/organization and then each tag enabled for billing. Tags can also be referenced within IAM policy documents and used to grant/restrict access to resources for IAM users, groups, and roles.

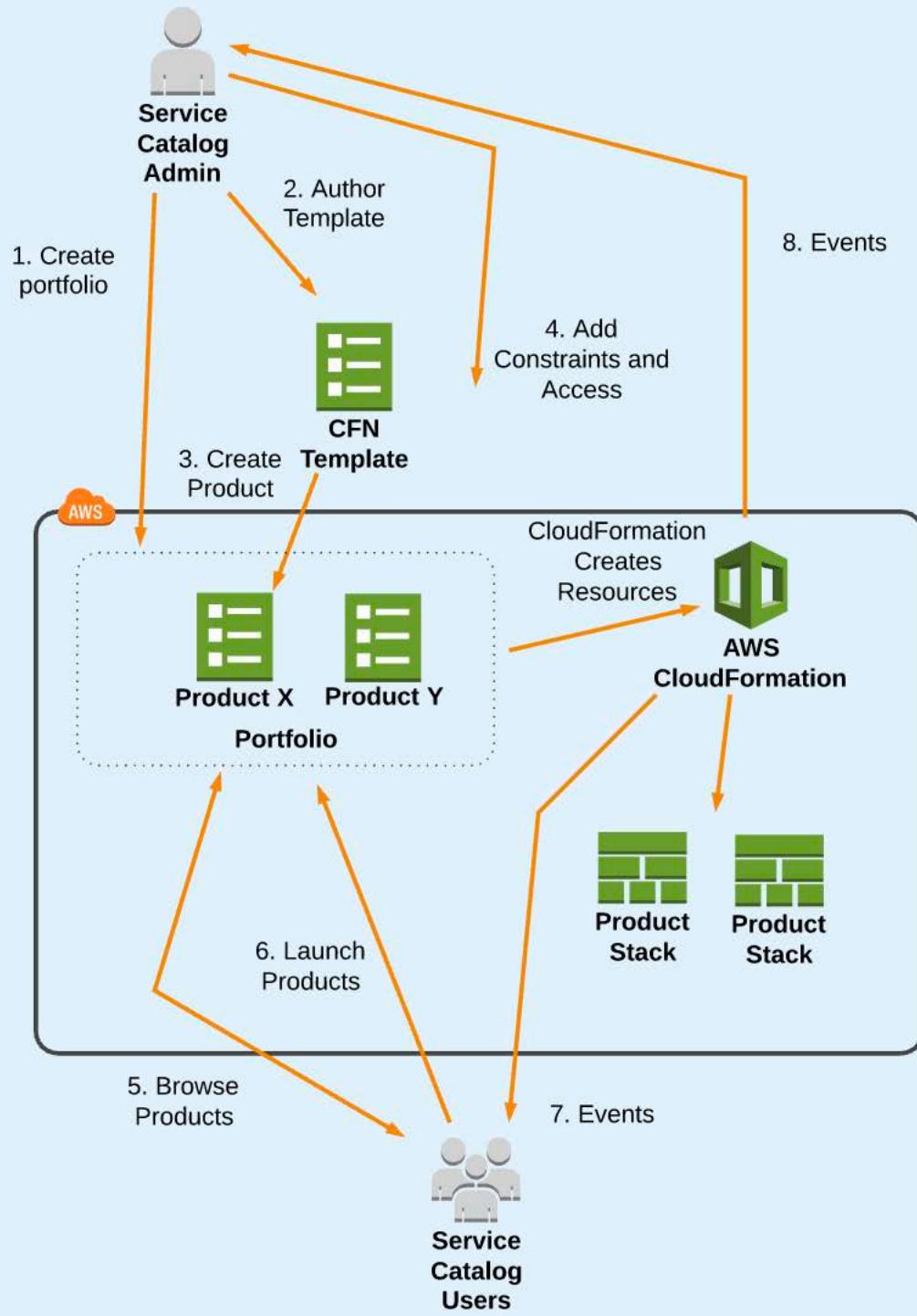
Think carefully about your tagging strategy and categories. All tags should support the business in terms of billing, security, and automation.

Technical	Automation	Business	Security
Name: Identification Application ID: System components Application Role: The function of the application Cluster Name/ID: Node identification of cluster members Environment: Production, Staging, Test, and Development Version	Backup Type Date/Time: Schedules for a resource Exclude: Exclude from automation Security: Requirements for encryption, logging, and access restrictions	Owner Department Cost Center Customer or Client Project Expiry/Project Close SLAs	Confidentiality Compliance Access Restrictions Region Restrictions



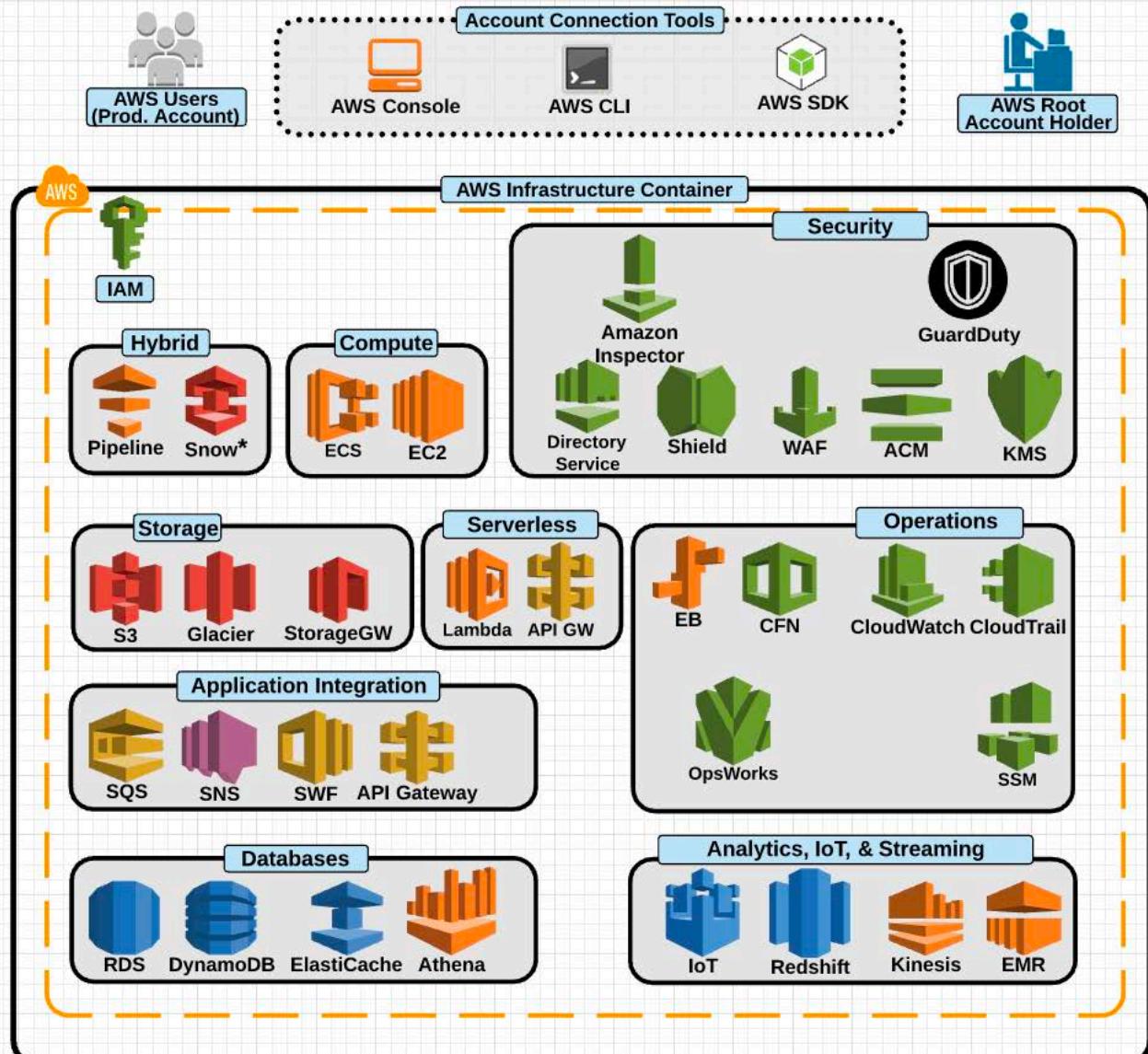
AWS Service Catalog

Service Catalog is a product that allows administrators to define products and portfolios (groups of products and configurations). Users can be allowed to self-service deploy these products without the usual IAM permissions required to do so directly with AWS services.



[?](#) [Back](#)[Global Infrastructure](#)[Account & Services](#)[Networking](#)[Well-Architected Appendix](#)

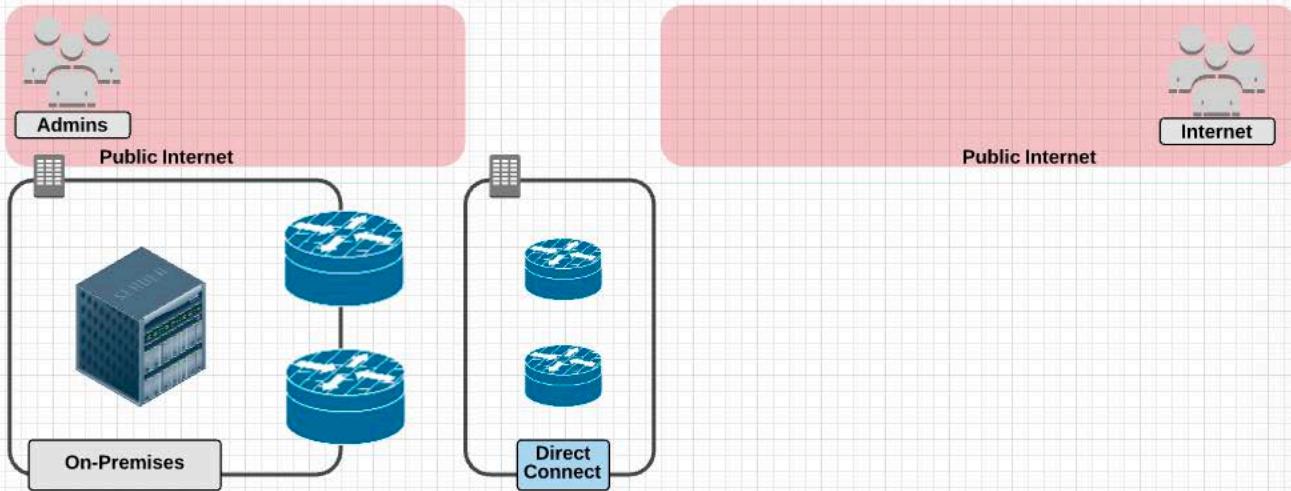
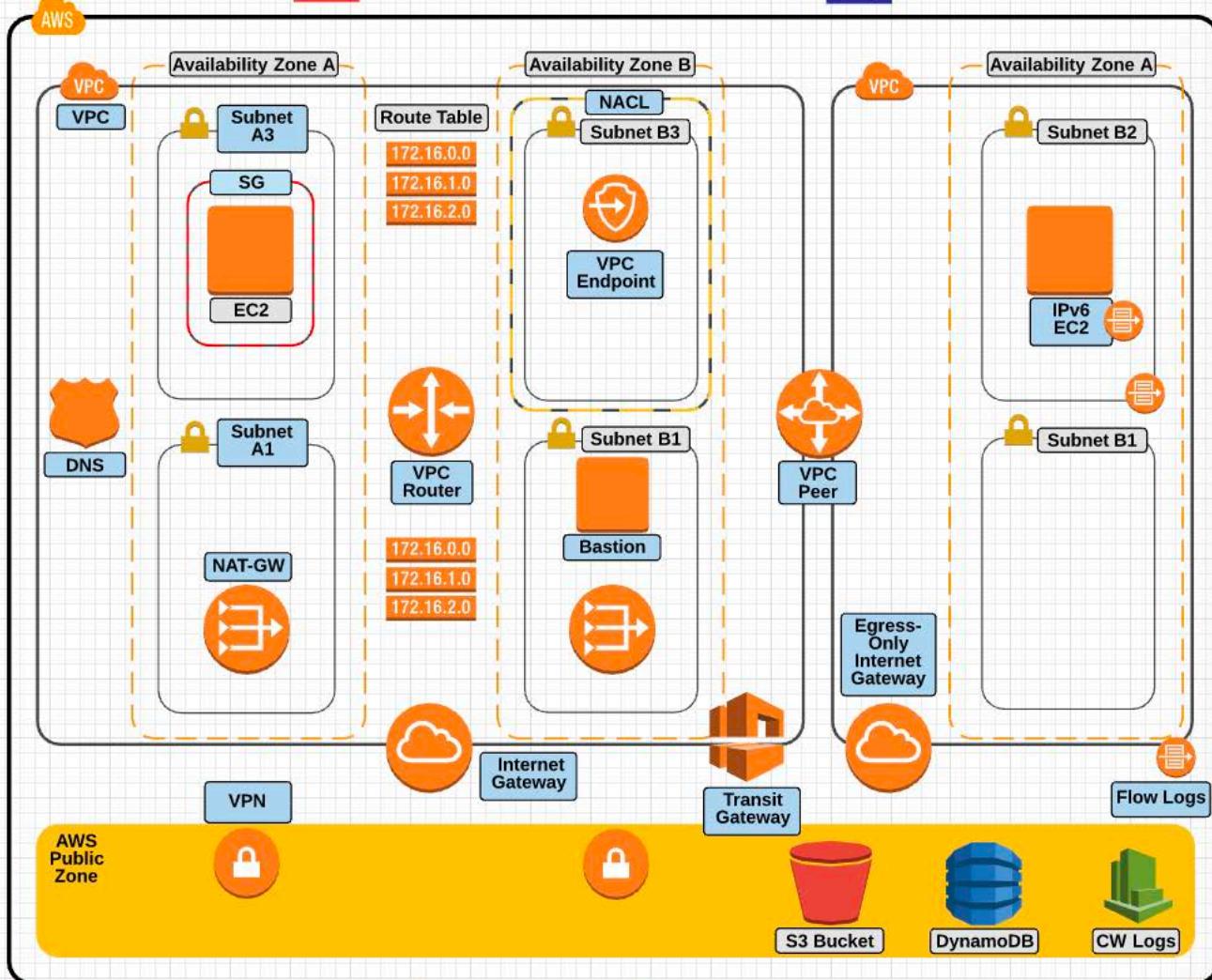
The Account and Services layer represents how we create, access, and manage an AWS account and its services, from interacting with an AWS account and managing user rights, to accessing and using various AWS services and features.

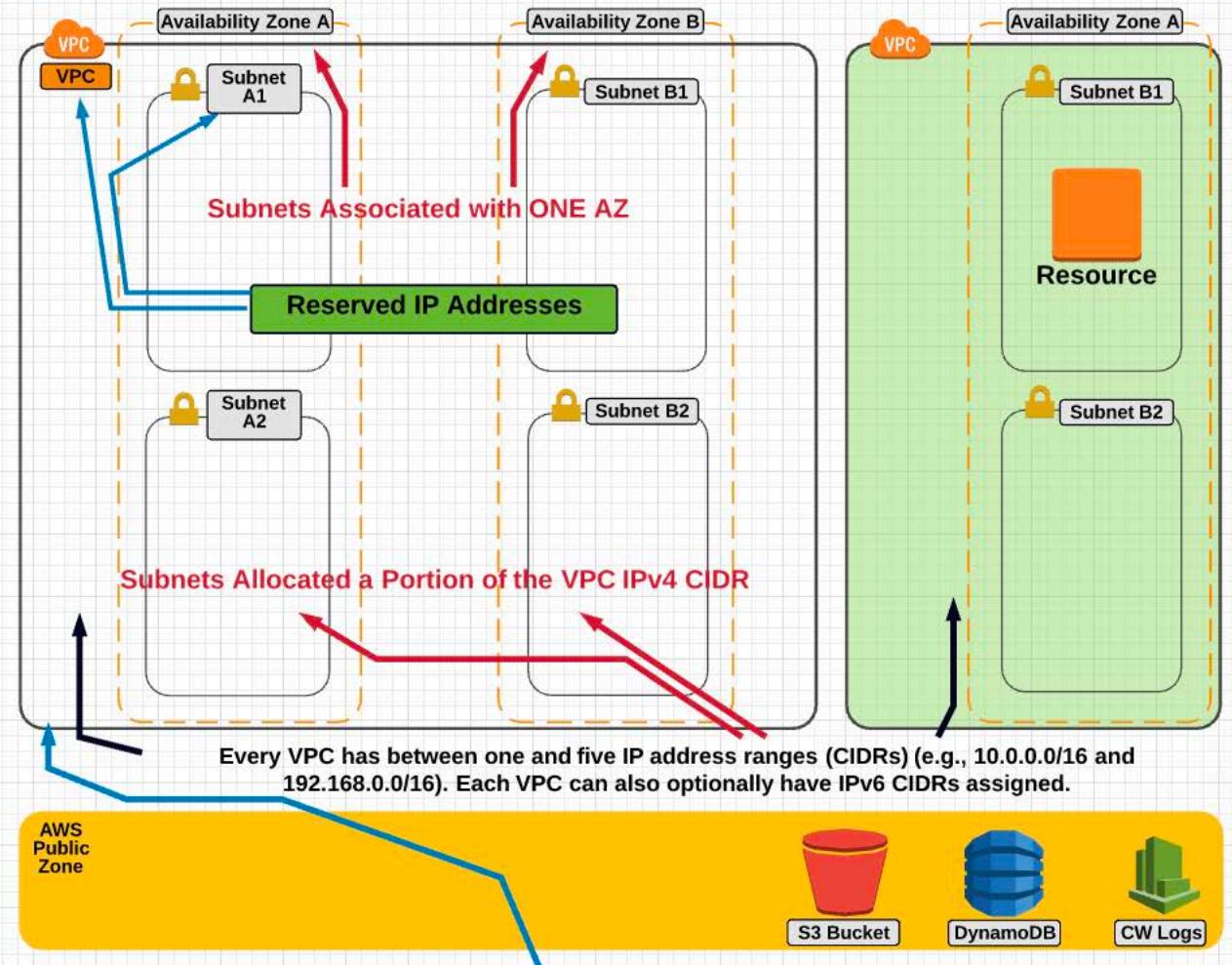




us-east-1

ap-southeast-2





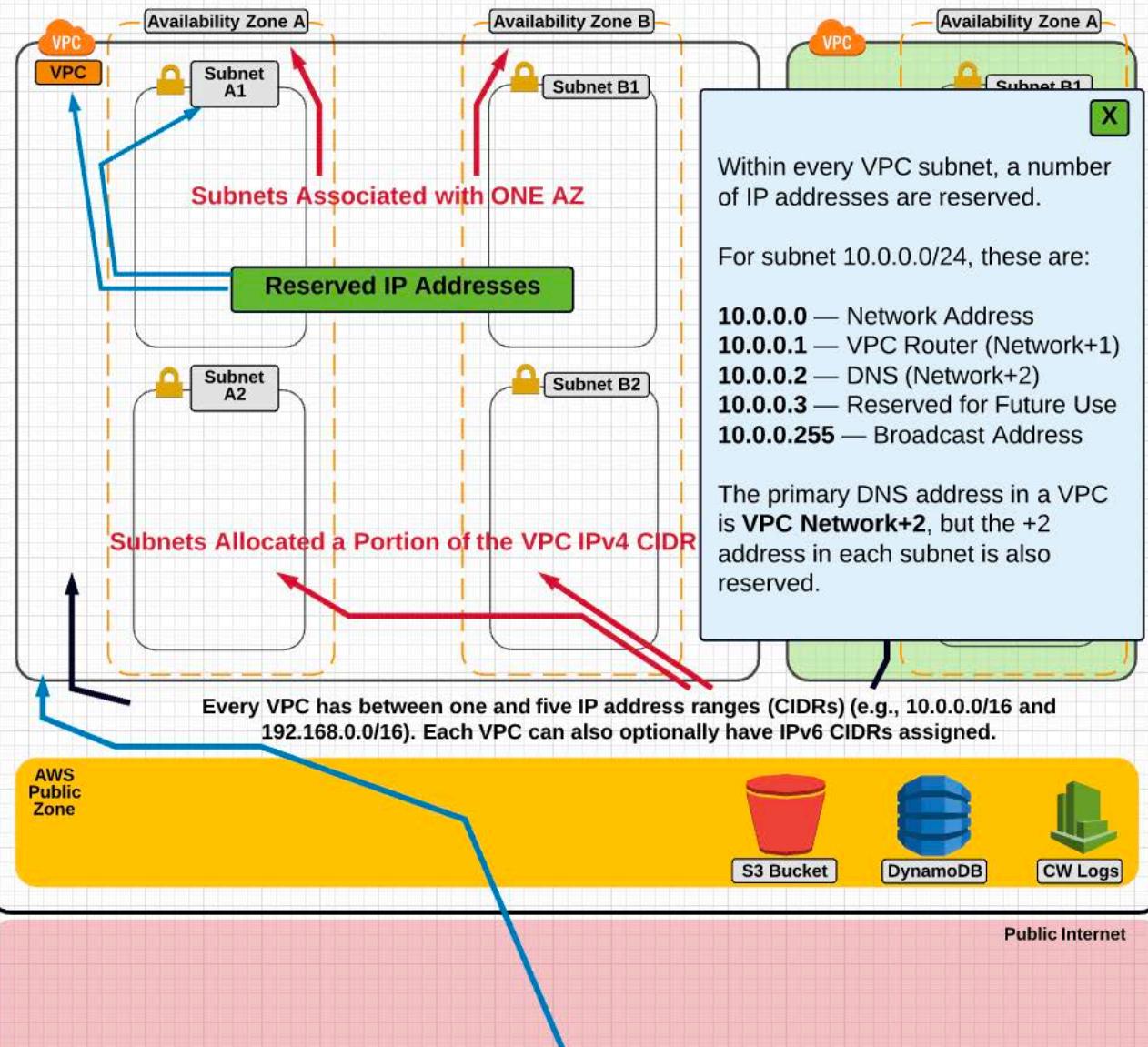
Network Zones: Private (VPC), Public Internet, AWS Public Services

VPC Tenancy Modes: Default and Dedicated

Default mode lets you select any of the tenancy options for VPC resources later if you need to. Dedicated mode limits your future tenancy options.



AWS



Network Zones: Private (VPC), Public Internet, AWS Public Services

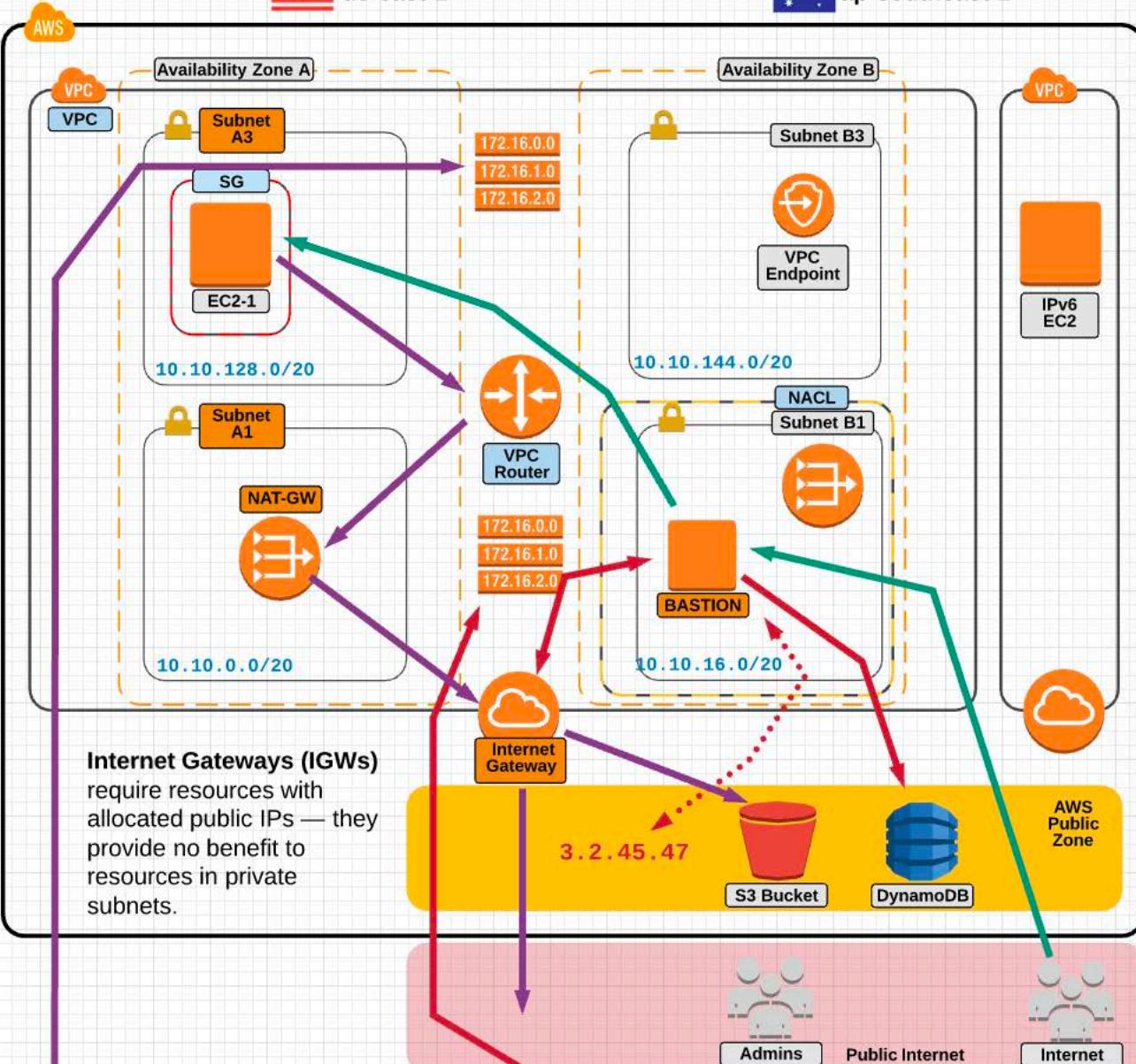
VPC Tenancy Modes: Default and Dedicated

Default mode lets you select any of the tenancy options for VPC resources later if you need to. Dedicated mode limits your future tenancy options.



us-east-1

ap-southeast-2



Destination	Target
10.10.0.0/16	local
0.0.0.0/0	nat-0d12780b15c2f8c86

Destination	Target
10.10.0.0/16	local
0.0.0.0/0	igw-997a6be1



Admins



Public Internet



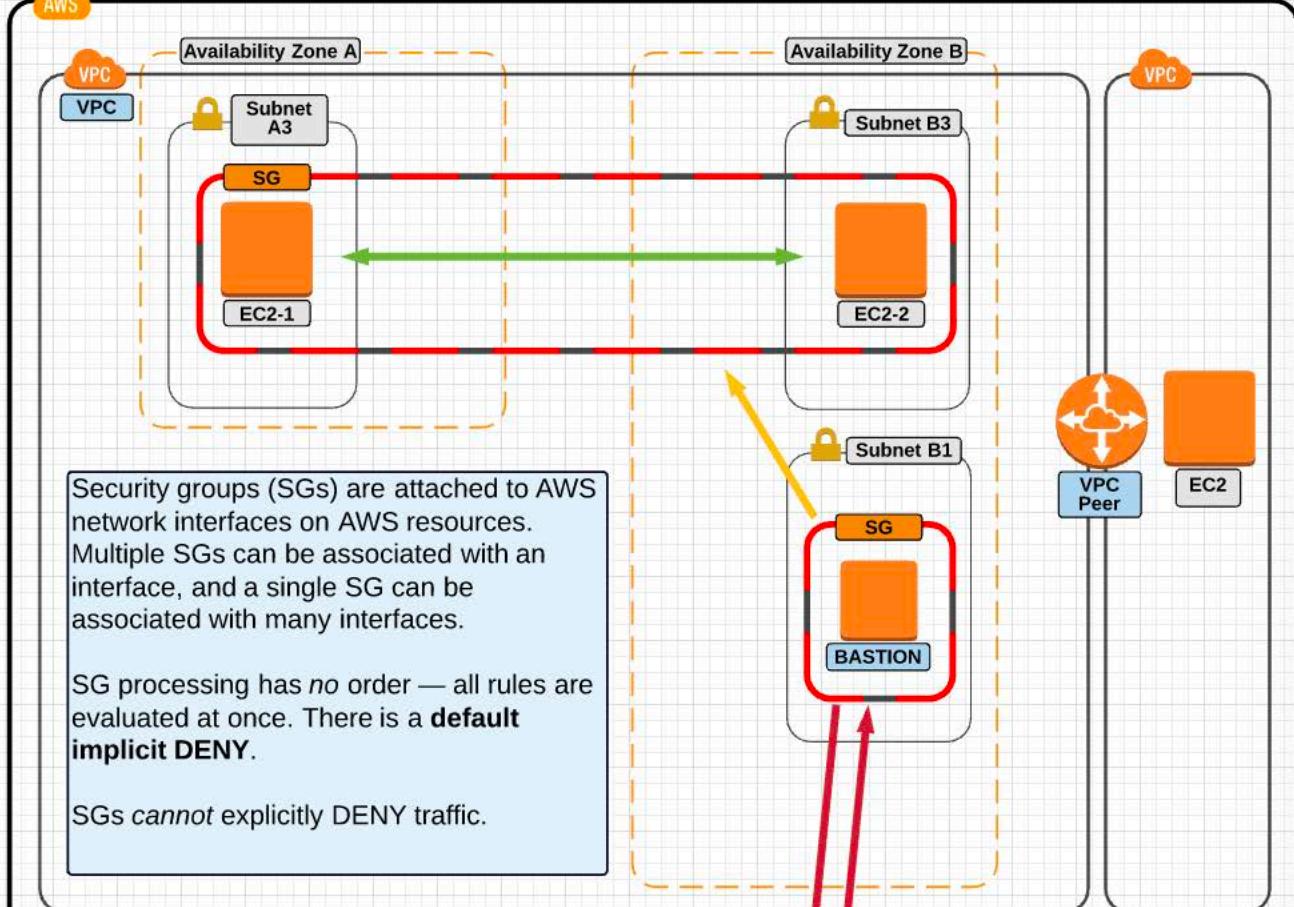
Internet



us-east-1

ap-southeast-2

AWS

AWS
Public
Zone

Public Internet

Inbound

Stateful

Type <i>i</i>	Protocol <i>i</i>	Port Range <i>i</i>	Source <i>i</i>
SSH	TCP	22	Custom 0.0.0.0/0

Type *i*Protocol *i*Port Range *i*Destination *i*

All traffic

All

All

Custom

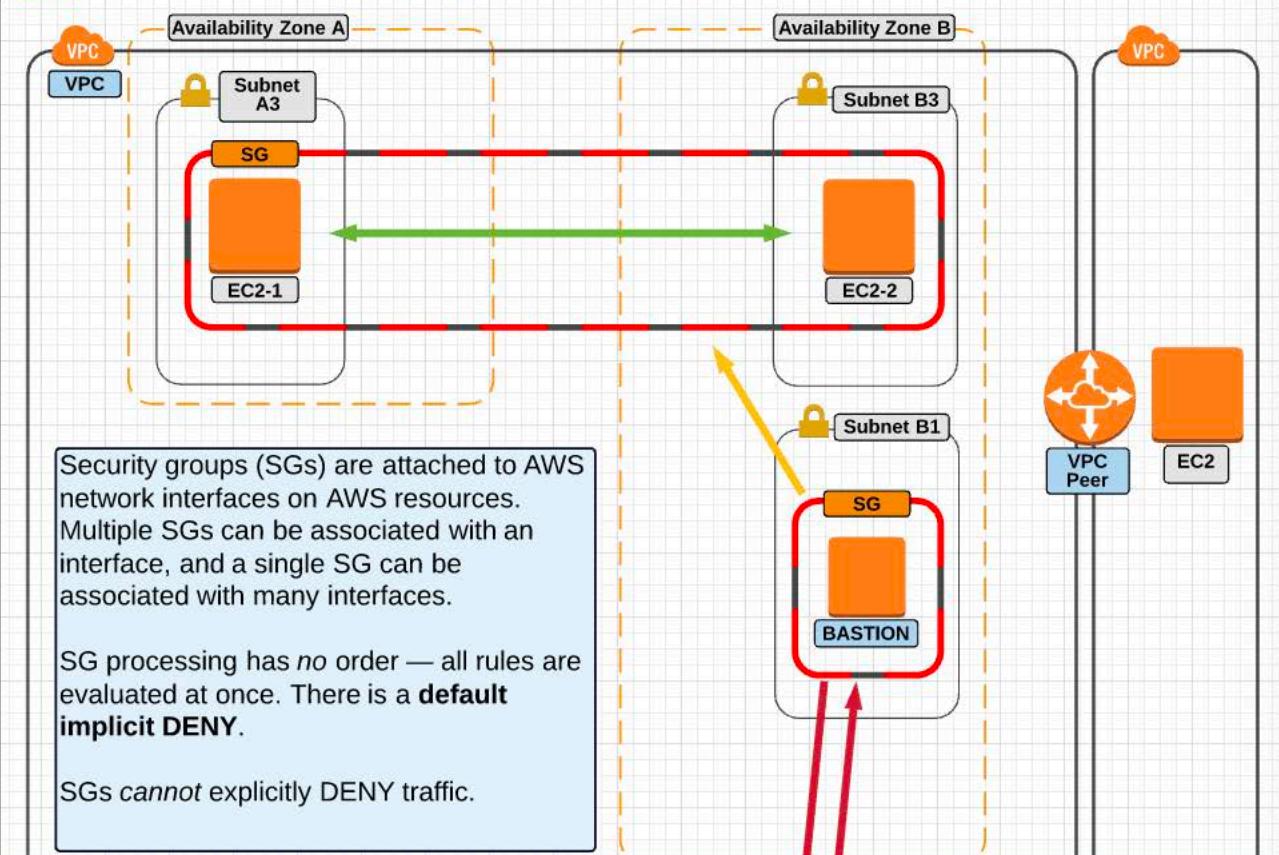
0.0.0.0/0

Outbound



us-east-1

ap-southeast-2



Security groups are *stateful*. If traffic is allowed in, the corresponding return traffic is automatically allowed. This cannot be changed. The same logic applies for outgoing traffic — its return communications are automatically permitted.

This means you don't have to worry about adding ephemeral ports.



Inbound

Stateful

Type <i>i</i>	Protocol <i>i</i>	Port Range <i>i</i>	Source <i>i</i>
SSH	TCP	22	Custom
0.0.0.0/0			

Type *i*Protocol *i*Port Range *i*Destination *i*

Outbound

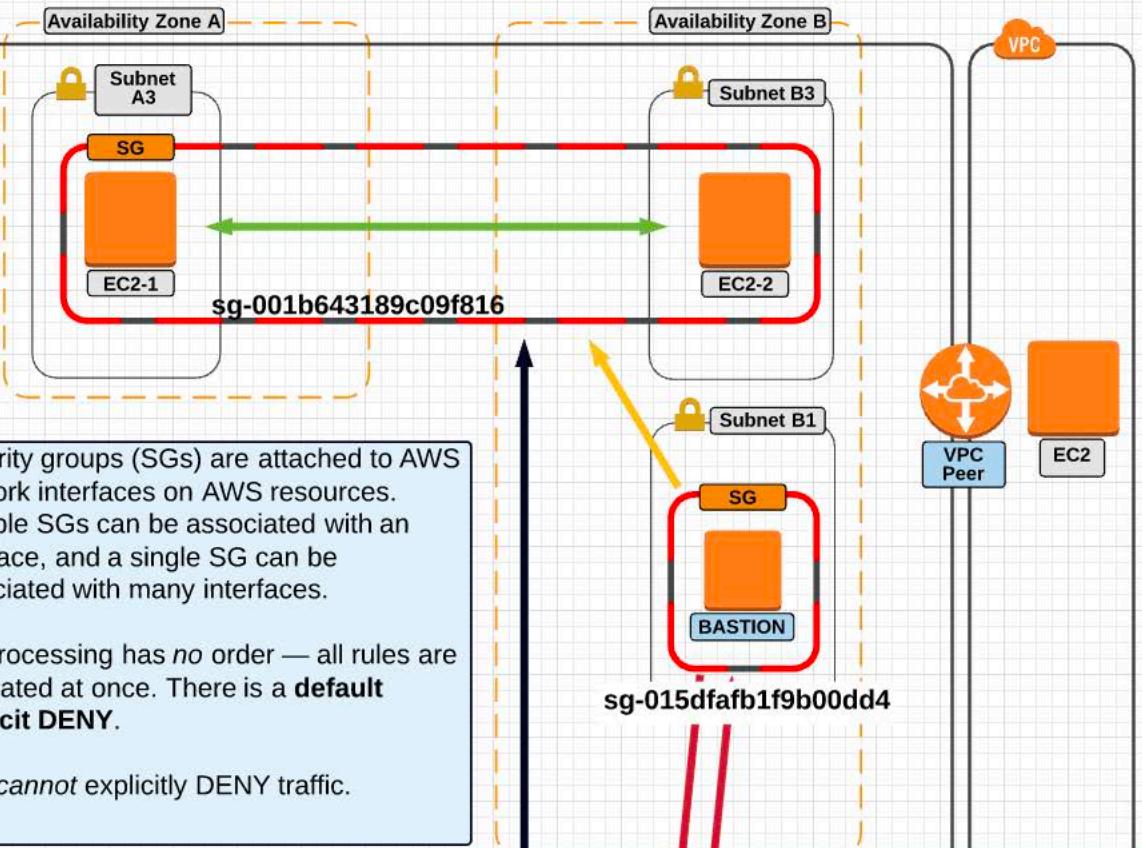
All traffic

All

All

Custom

0.0.0.0/0



Security groups are aware of AWS logical resources. This means a security group's *Source* or *Destination* field can reference CIDRs or logical resources including EC2 instances, other security groups, or even itself.

Inbound

Source	Custom	sg-001b643189c09f816
Custom	sg-015dfaf1f9b00dd4	X

Type	Protocol	Port Range	Source
Stateful	TCP	22	Custom
SSH		Custom	0.0.0.0/0

Outbound

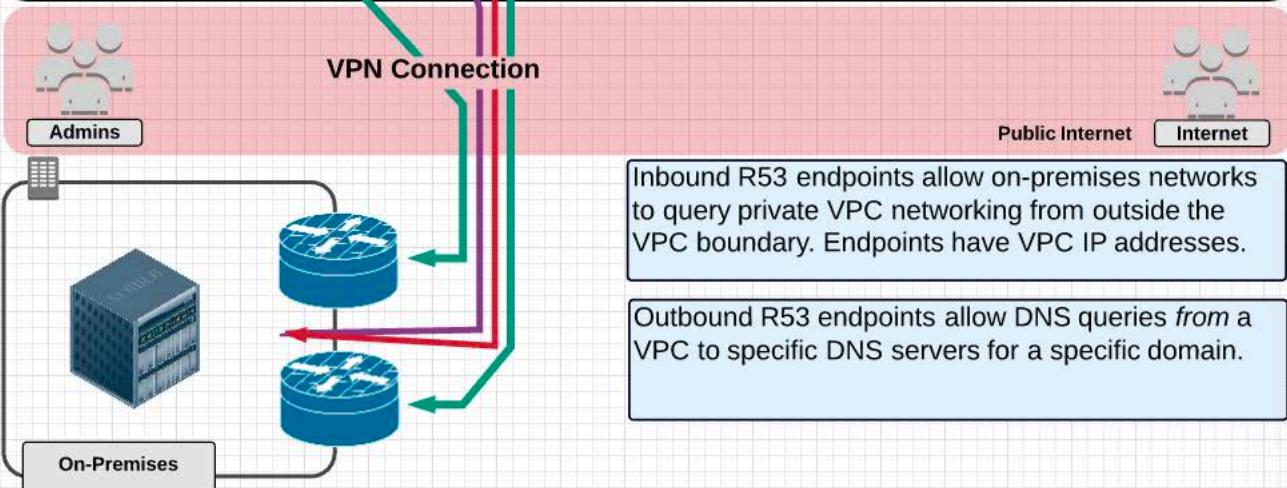
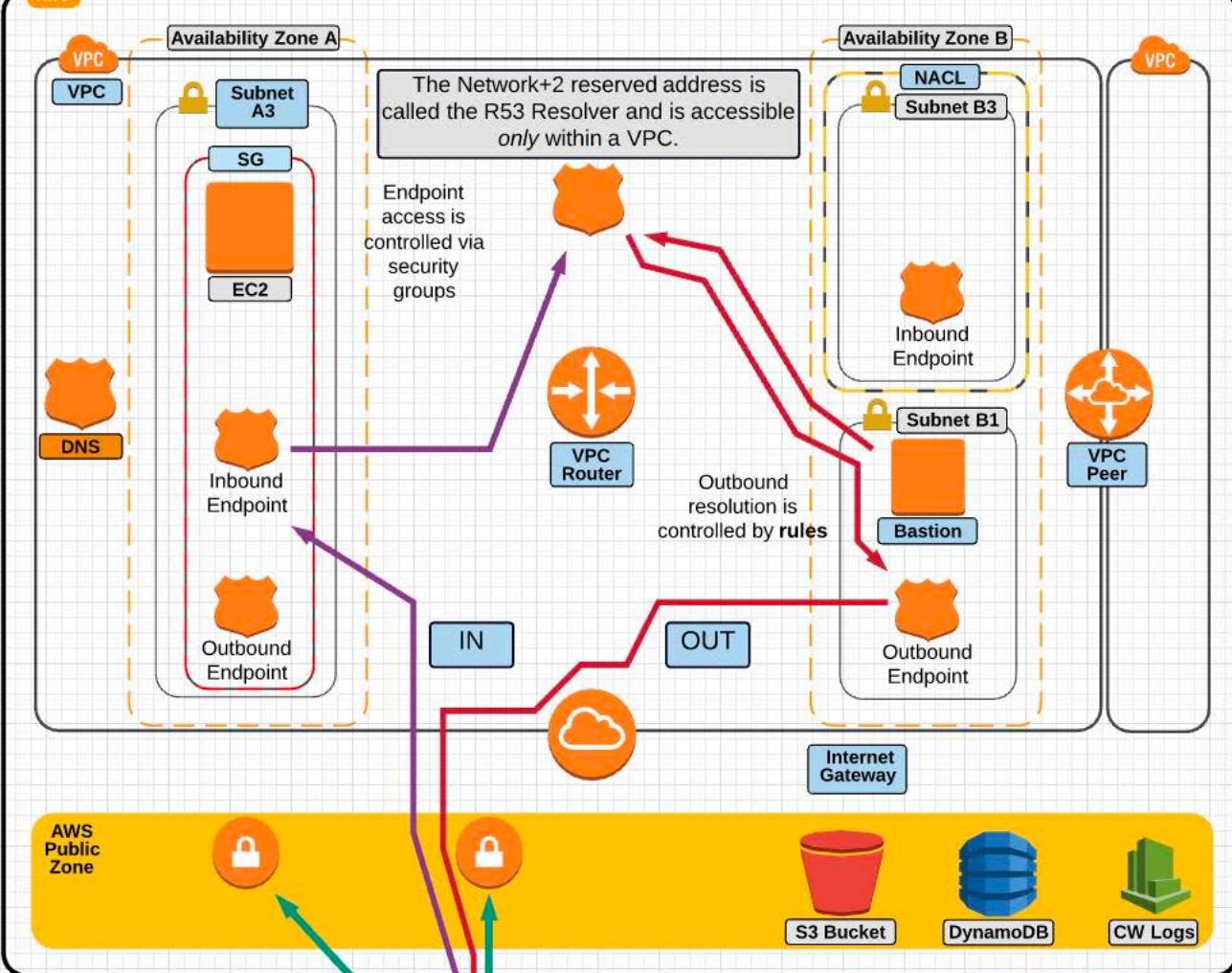
Type	Protocol	Port Range	Destination
All traffic	All	All	Custom
			0.0.0.0/0



us-east-1

ap-southeast-2

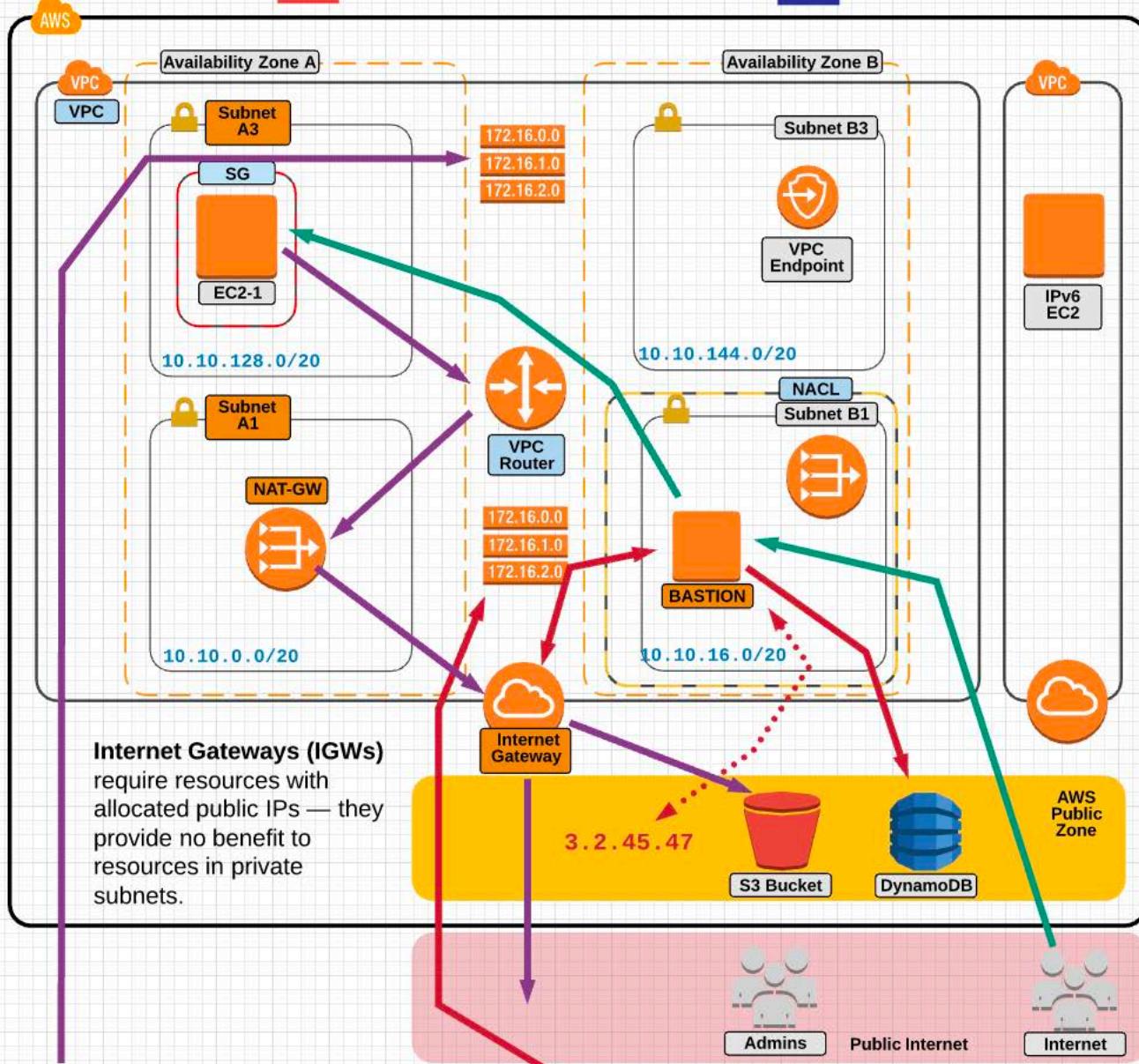
AWS





us-east-1

ap-southeast-2



Destination	Target
10.10.0.0/16	local
0.0.0.0/0	nat-0d12780b15c2f8c86

Destination	Target
10.10.0.0/16	local
0.0.0.0/0	igw-997a6be1



Admins

Public Internet



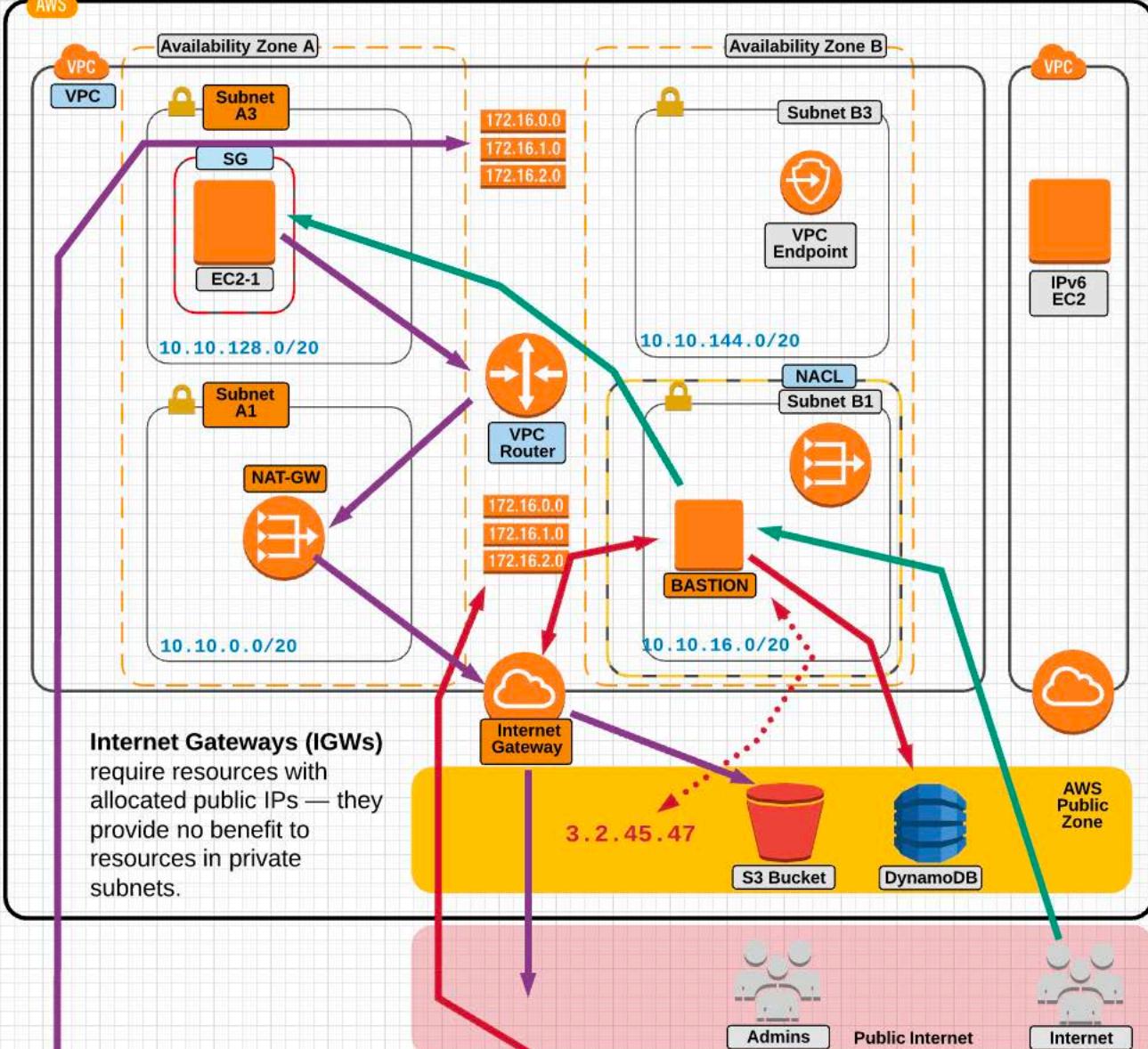
Internet



us-east-1

ap-southeast-2

AWS

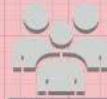


Destination	Target
10.10.0.0/16	local
0.0.0.0/0	nat-0d12780b15c2f8c86

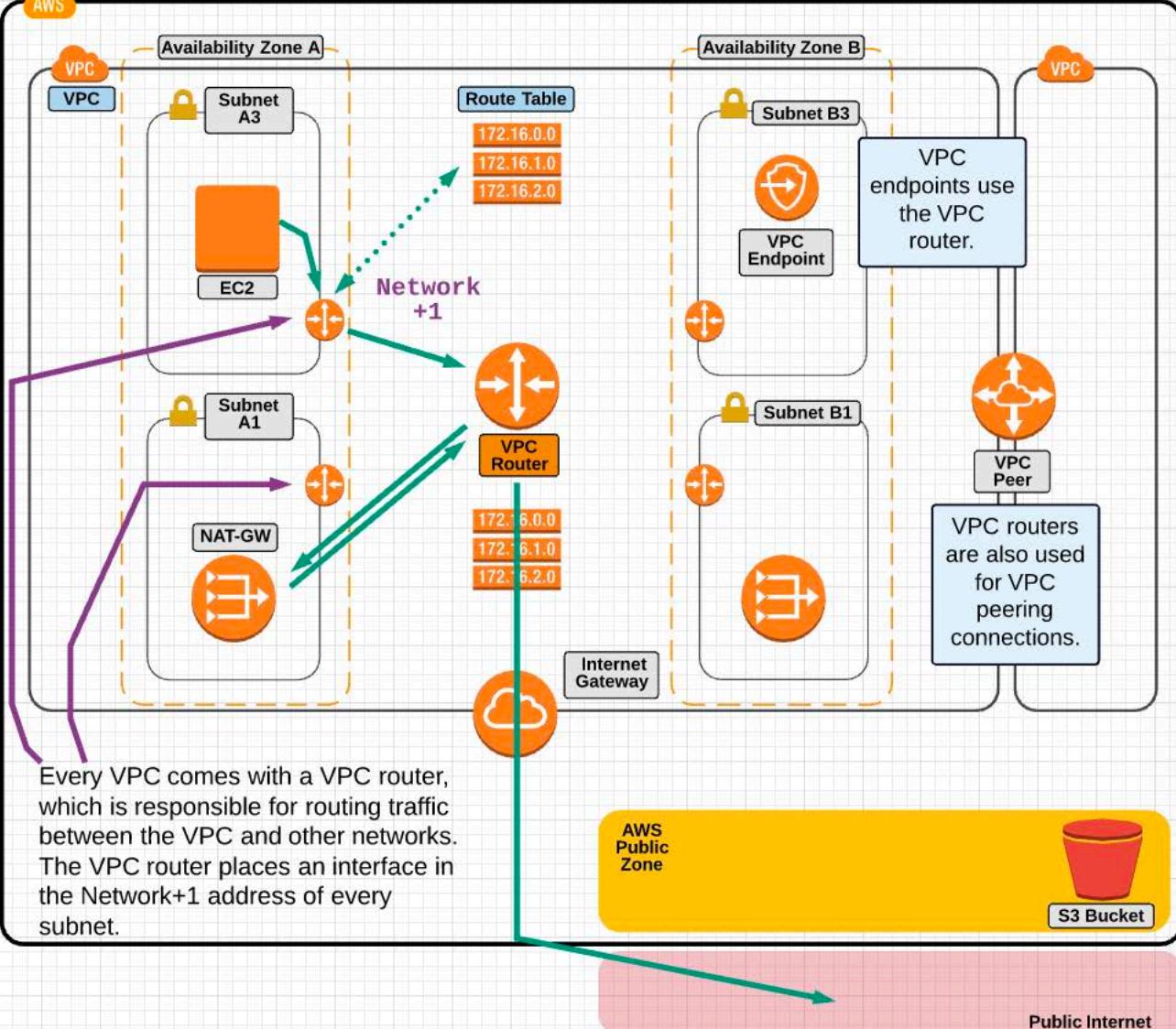
Destination	Target
10.10.0.0/16	local
0.0.0.0/0	igw-997a6be1

Internet

Public Internet

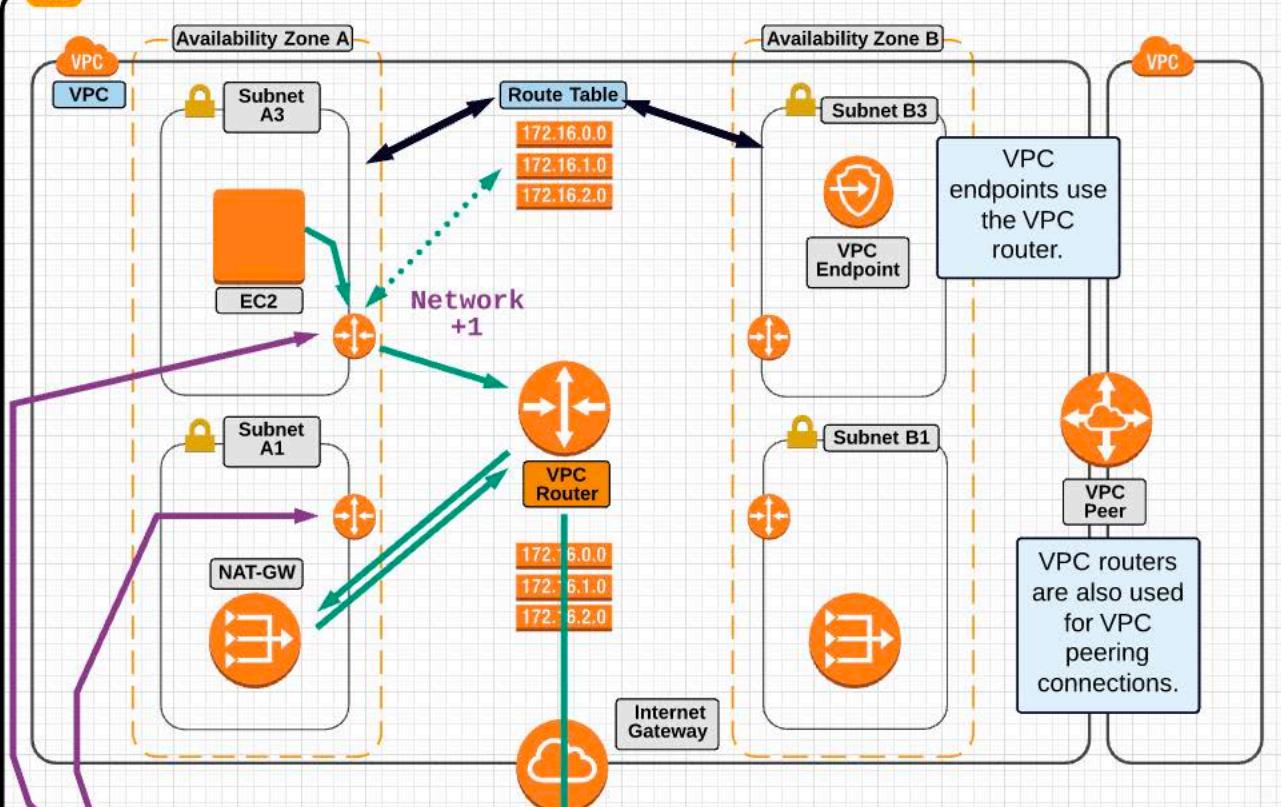


Admins





AWS



Every VPC comes with a VPC router, which is responsible for routing traffic between the VPC and other networks. The VPC router places an interface in the Network+1 address of every subnet.

The VPC router is controlled by route tables (RTs).

VPCs come with a default "main" route table.

Custom RTs can be defined and attached to subnets.



S3 Bucket

Public Internet

Route tables contain a **default "local"** route for the VPC.

Routes map to a **destination** and tell the VPC router where to direct IP traffic (at the **target**). More specific routes take priority (e.g., /32 > /16).

See VPN Lesson X

Destination	Target	Status	Propagated
10.10.0.0/16	local	active	No
0.0.0.0/0	igw-997a6be1	active	No

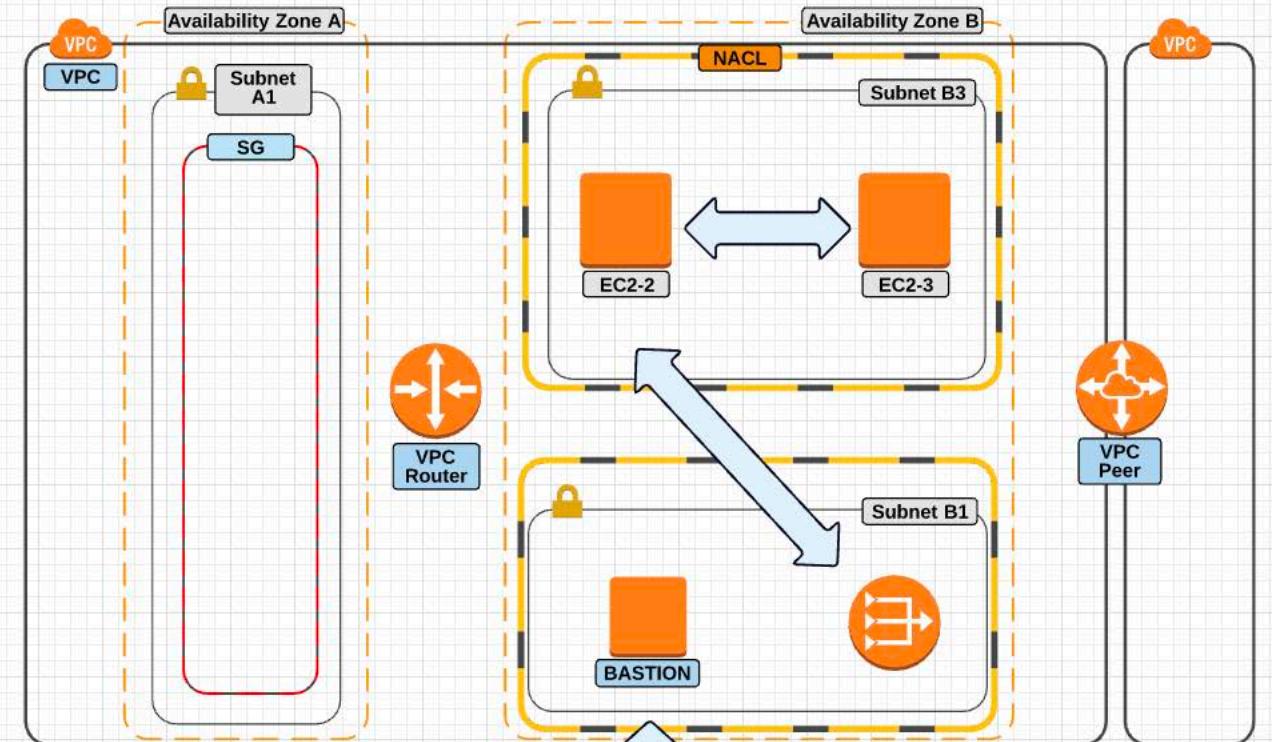


us-east-1



ap-southeast-2

AWS

AWS
Public
Zone**A subnet can have one associated NACL.****VPCs have a default NACL associated with any default subnets in the VPC.**

S3 Bucket

Public Internet

A NACL placed around the bastion and NATGW subnet could DENY inbound traffic from the public internet, control outbound traffic from the NATGW, and filter which EC2 instances are able to reach the NATGW.

NACLs are often used to provide explicit DENY protection to internet subnets.

Network Access Control Lists (NACLs) are firewalls attached to VPC subnets. They filter IP traffic entering or leaving the subnet. NACLs are *stateless*, which means ephemeral port rules are needed in order to permit response traffic.

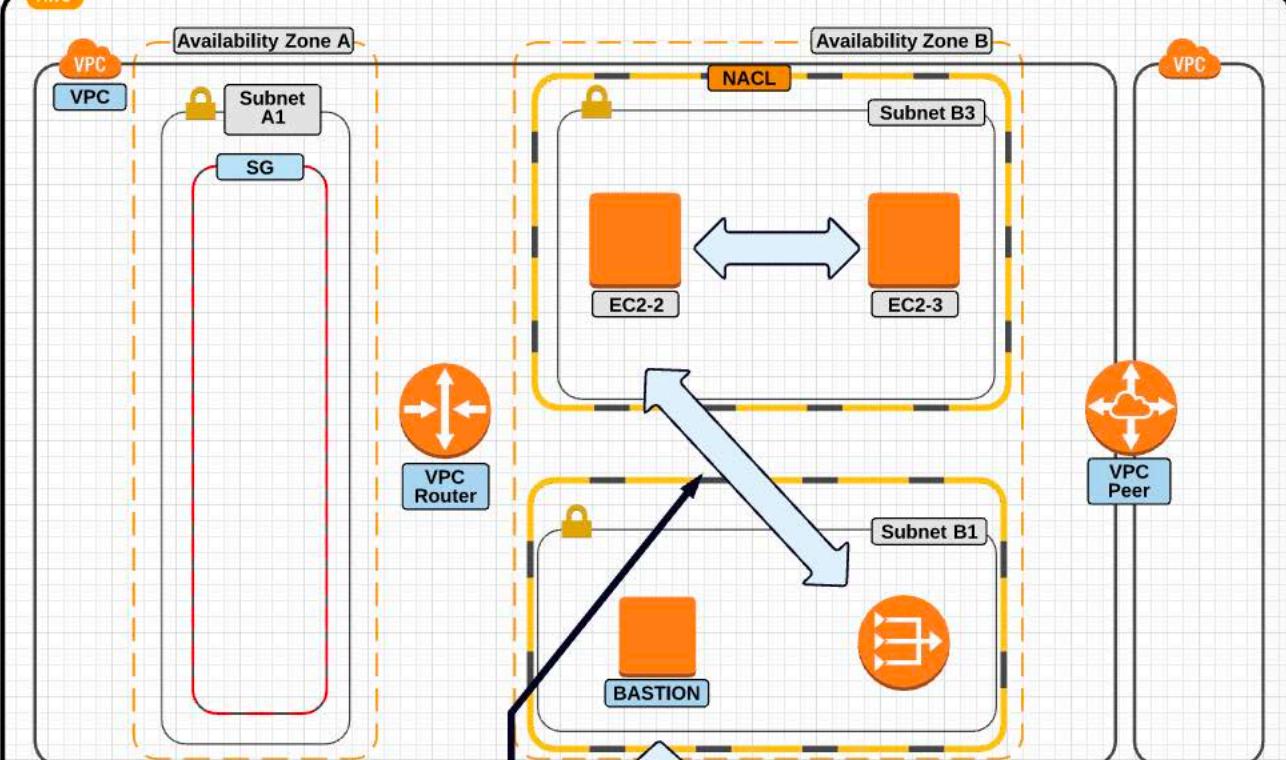
NACLs can explicitly DENY and their rules are processed in order.



us-east-1

ap-southeast-2

AWS

AWS
Public
ZoneVPN
Gateway

A subnet can have one associated NACL.

VPCs have a default NACL associated with any default subnets in the VPC.



S3 Bucket

Public Internet

Any IP traffic between the top and bottom subnets would be subject to processing out of the top subnet and in to the bottom subnet. The same applies for return traffic.

Network Access Control Lists (NACLs) are firewalls attached to VPC subnets. They filter IP traffic entering or leaving the subnet. NACLs are stateless, which means ephemeral port rules are needed in order to permit response traffic.

NACLs can explicitly DENY and their rules are processed in order.

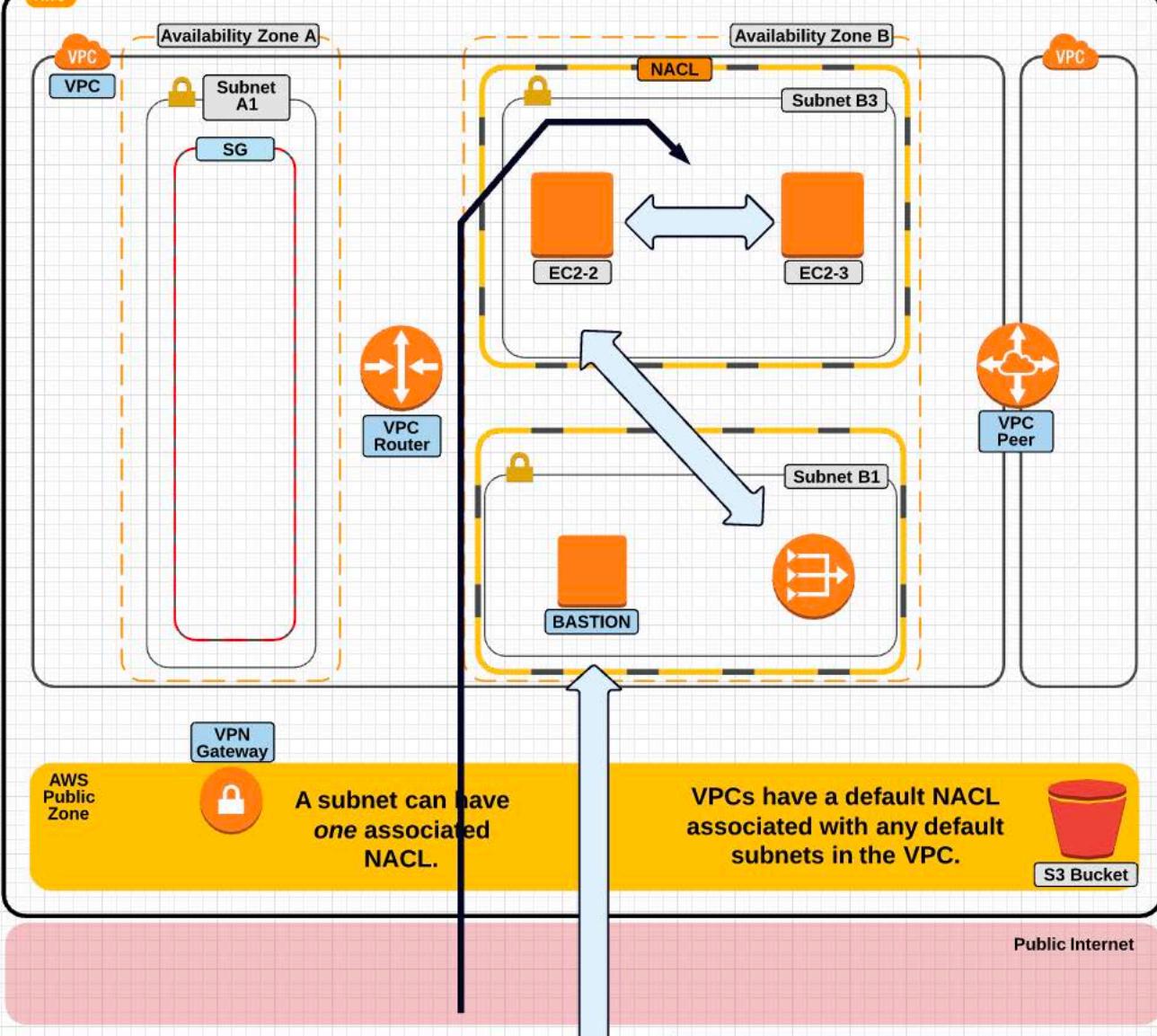


us-east-1



ap-southeast-2

AWS



Communications between the EC2 instances would be unaffected by any NACLs — NACL processing only occurs when traffic crosses over a subnet boundary (in or out).

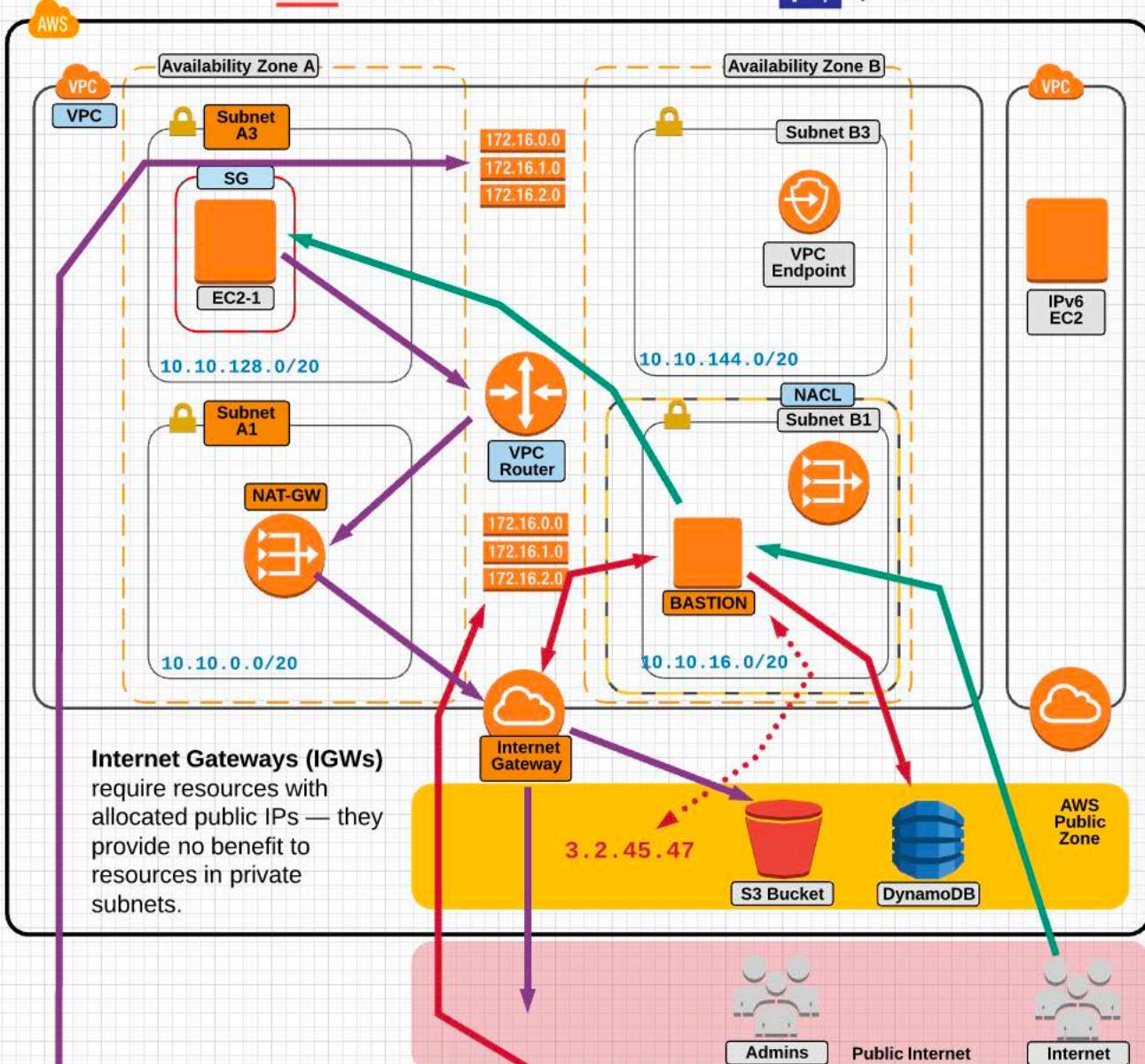
Network Access Control Lists (NACLs) are firewalls attached to VPC subnets. They filter IP traffic entering or leaving the subnet. NACLs are stateless, which means ephemeral port rules are needed in order to permit response traffic.

NACLs can explicitly DENY and their rules are processed in order.



us-east-1

ap-southeast-2



Destination	Target
10.10.0.0/16	local
0.0.0.0/0	nat-0d12780b15c2f8c86

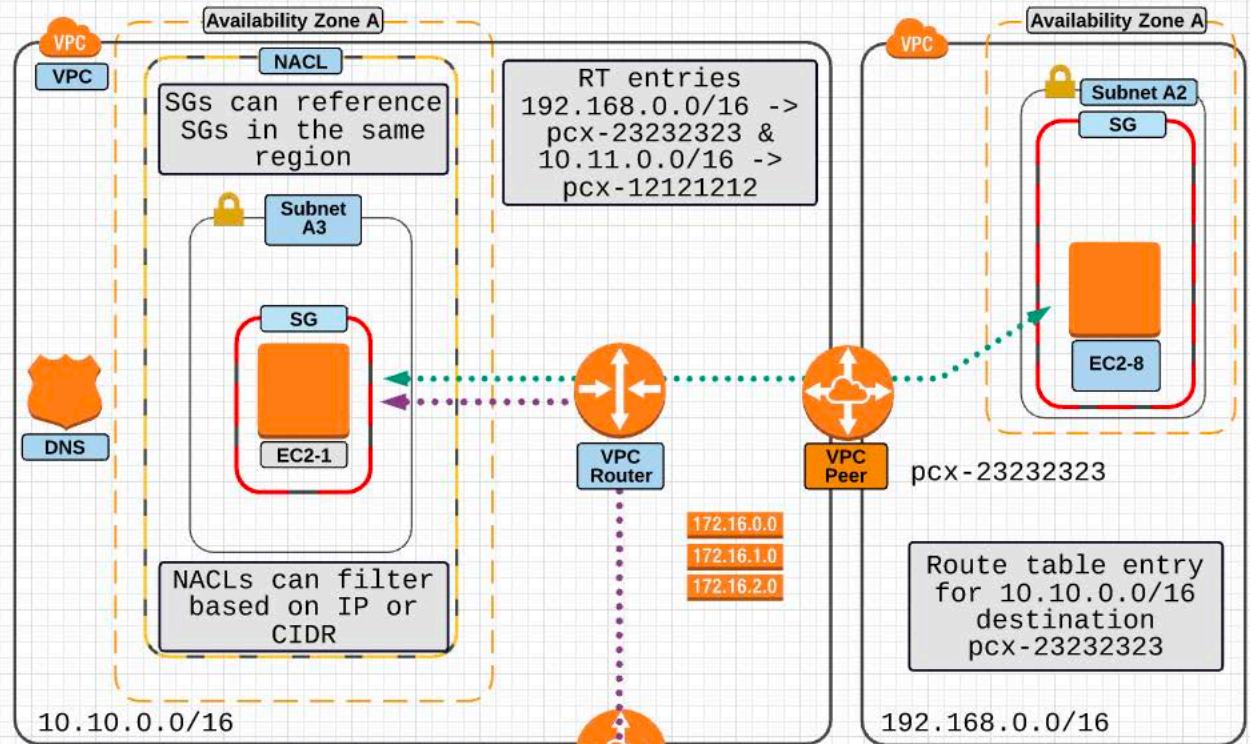
Destination	Target
10.10.0.0/16	local
0.0.0.0/0	igw-997a6be1



us-east-1

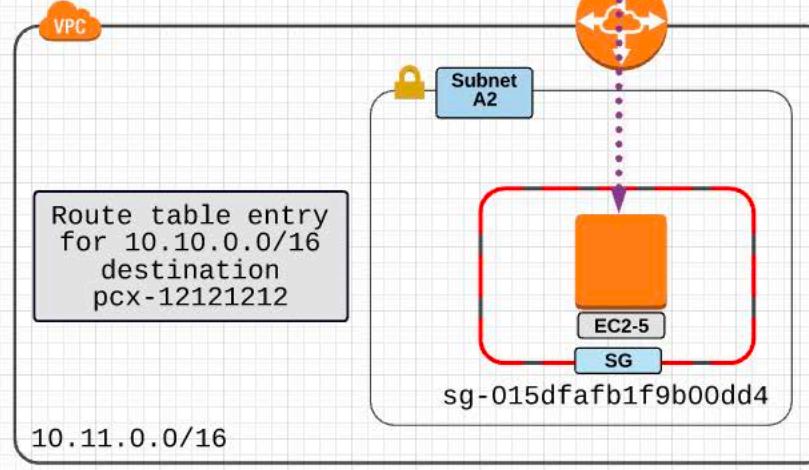
ap-southeast-2

AWS



Limitations and Considerations

Account Boundary



VPC peering connections allow communication between isolated VPCs. They work fully featured across accounts and in a feature-limited way between AWS regions.

VPC peering connections are created as logical objects and require route table entries at both sides.

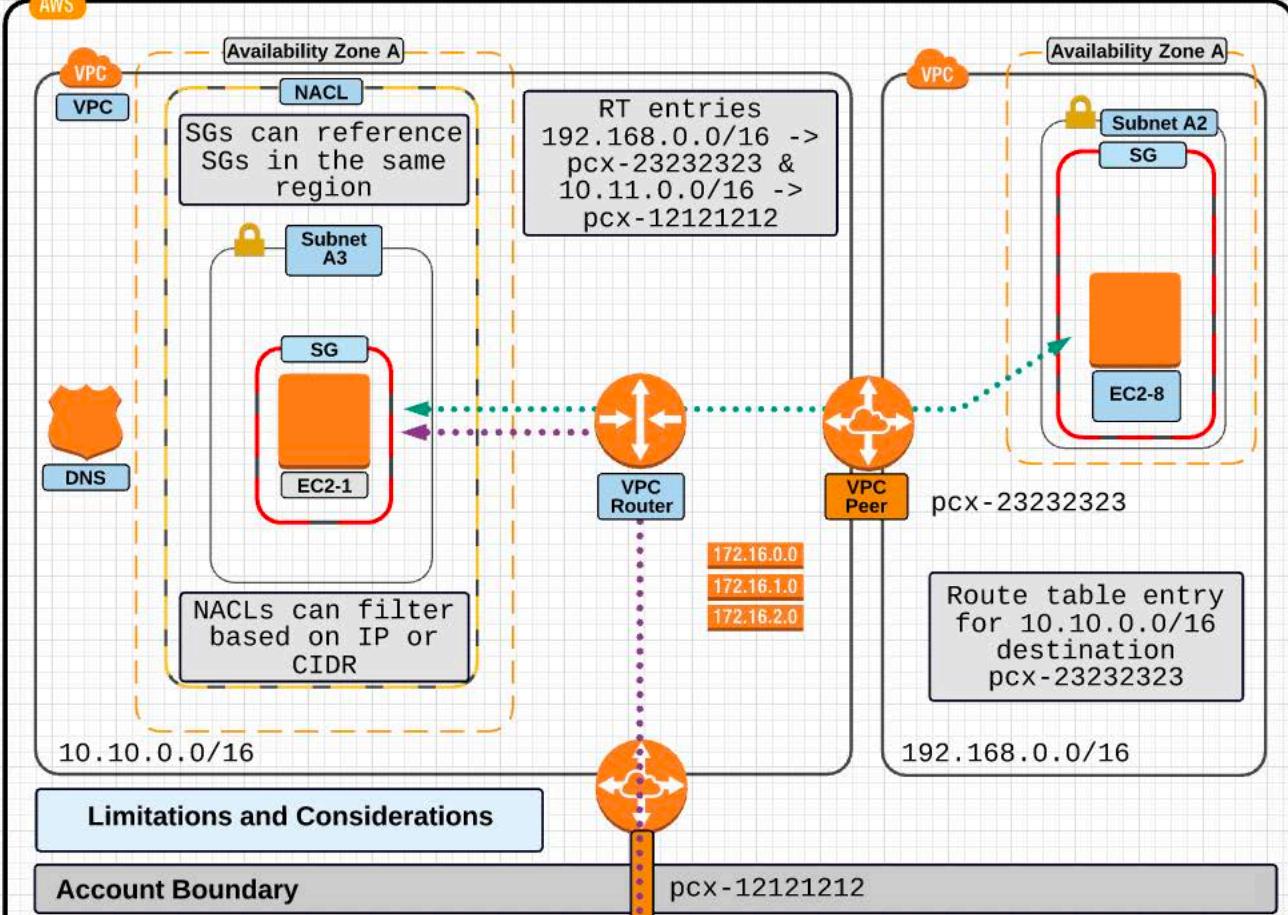
CIDRs cannot overlap!



us-east-1

ap-southeast-2

AWS



Intra-Region VPC Peering

Peering connections cannot peer VPCs with overlapping CIDRs.

Peering is *not* transitive. Peering A and B, then B and C, does not mean A and C are peered.

SGs can reference groups in remote VPCs. SGs and NACLs can also reference CIDR/IP.

IPv6 is supported, but not by default.

Private DNS hostnames can be resolved.

Inter-region VPC peering has the following additional limitations:

Inter (cross-) region peered VPCs do not support IPv6.

Security group logical referencing does *not* work cross-region.

Private DNS hostnames can be resolved but *must* be enabled.

X



us-east-1

ap-southeast-2

AWS

2600:1f18:23c5:d00::/56

Allocated VPC IPv6

2600:1f18:23c5:d00::/64

Allocated Subnet IPv6

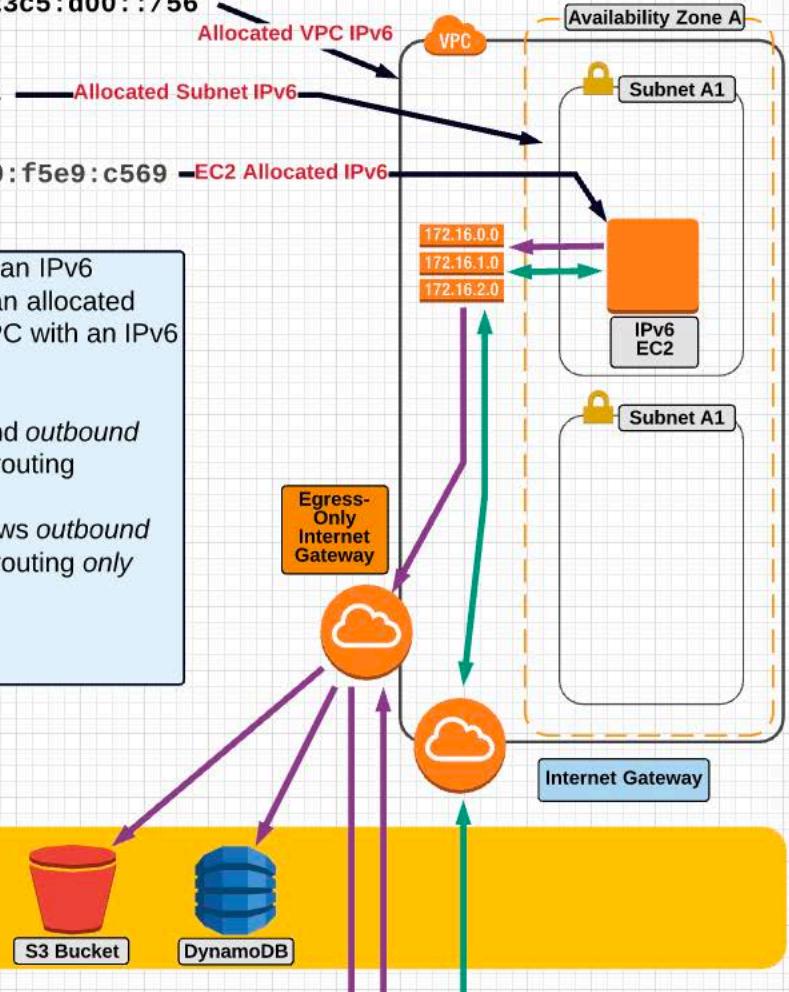
2600:1f18:23c5:d00:c488:42c9:f5e9:c569

EC2 Allocated IPv6

Scenario: A resource has been given an IPv6 address and occupies a subnet with an allocated IPv6 CIDR, which itself is part of a VPC with an IPv6 CIDR allocation.

Internet Gateway: Allows *inbound* and *outbound* public internet and AWS public zone routing

Egress-Only Internet Gateway: Allows *outbound* public internet and AWS public zone routing *only*

AWS
Public
Zone

S3 Bucket

DynamoDB

Resources inside a public subnet with IPv6 allocations are provided with publicly routable addresses because *all* IPv6 addresses are publicly routable.

As a result, **NAT gateways** aren't a valid option to provide outgoing-only access to the AWS public zone or public internet. NAT gateways translate between private-only IP addresses to a public address.



Admins



Public Internet



Internet

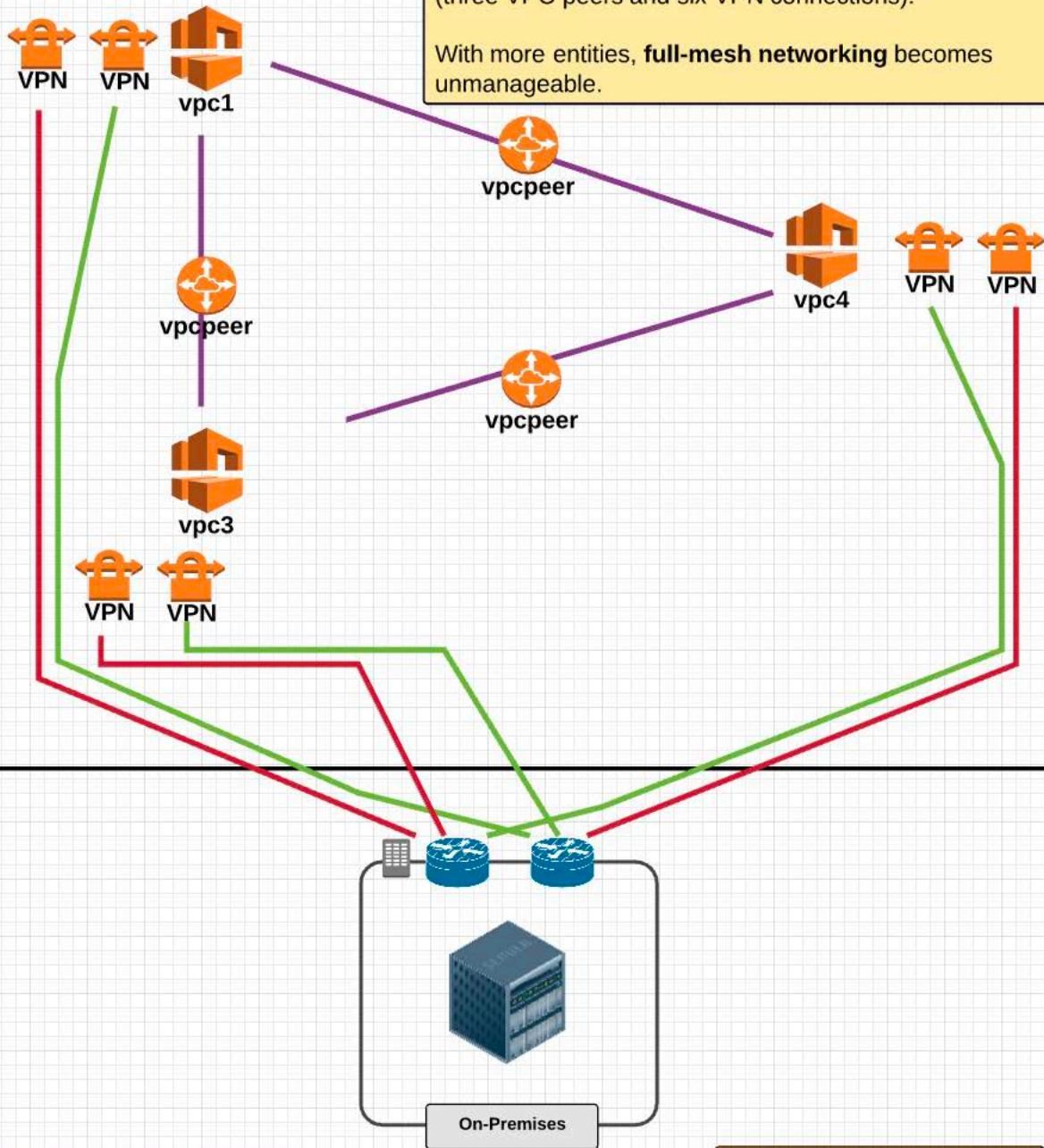


us-east-1

AWS

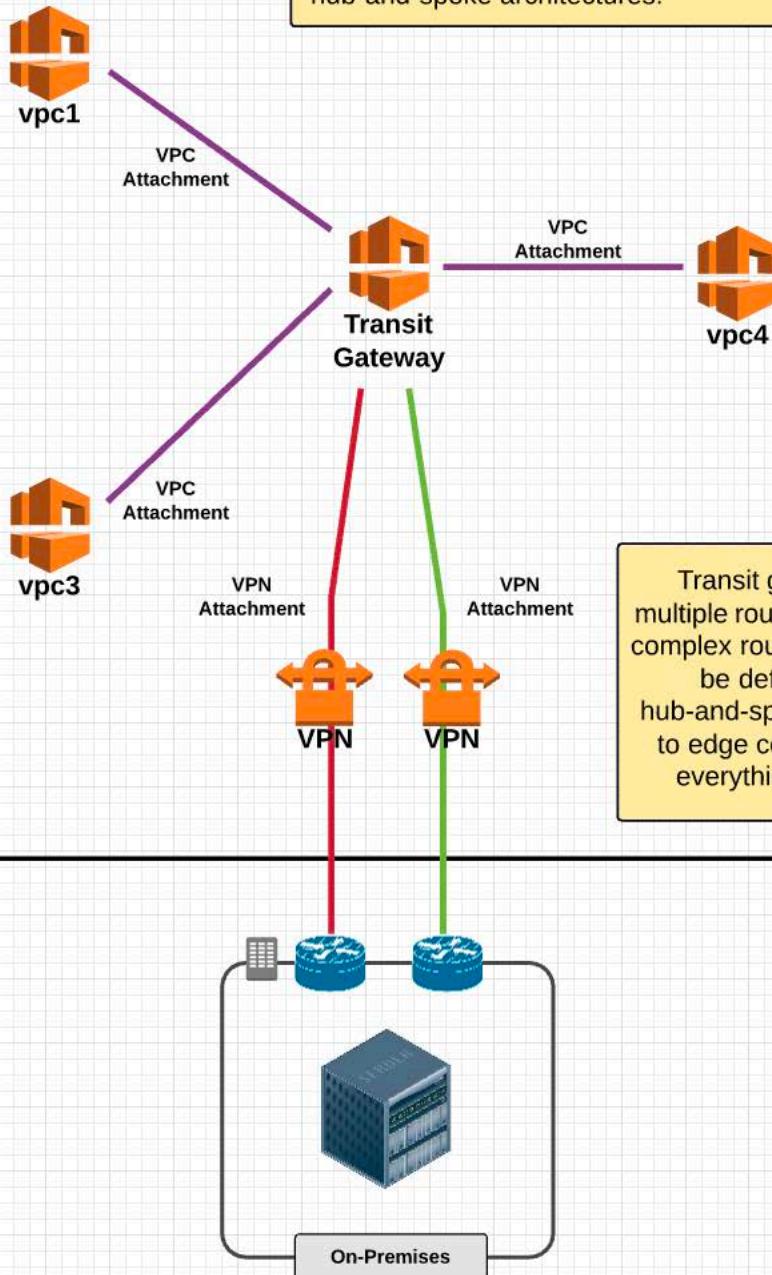
Non-transitive routing requires that all VPCs and on-premises locations have unique connections to all other entities. With a single on-premises network and three VPCs, this would result in a total of nine connections (three VPC peers and six VPN connections).

With more entities, **full-mesh networking** becomes unmanageable.





Transit gateways replace virtual private gateways for VPC and on-premises connections. Transit gateways can be attached to VPCs and VPNs (along with DX) and can perform transitive routing, allowing for hub-and-spoke architectures.





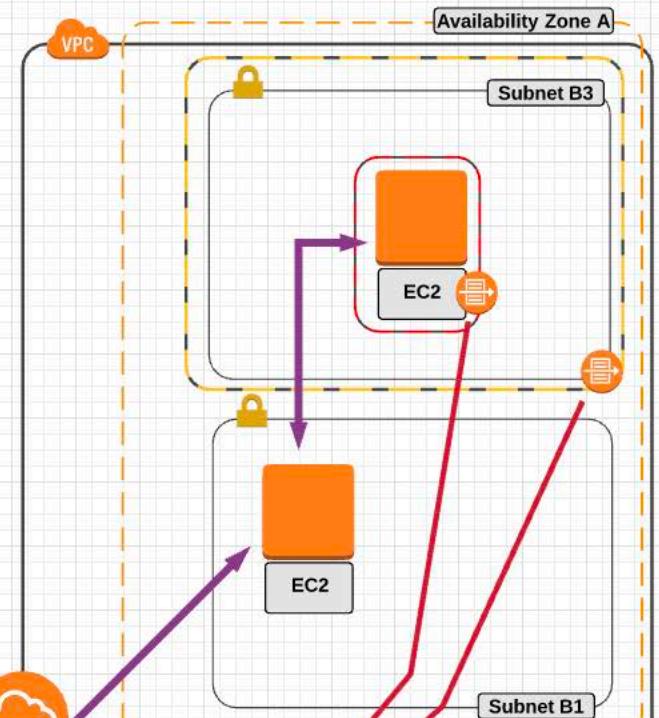
us-east-1

ap-southeast-2

AWS

```
version
account-id
interface-id
srcaddr
dstaddr
srcport
dstport
protocol
packets
bytes
start
end
action
log-status
```

LOG

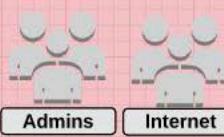


What Isn't Logged?

DHCP
AWS DNS
Metadata
License Activation Requests

AWS
Public
Zone

Public Internet



IAM roles provide permissions to the VPC Flow Logs service, either for CloudWatch log injection or for sending log files to an S3 bucket for storage.



VPC Flow Logs gather *metadata* for traffic flowing across a VPC. They don't log actual traffic, nor are they real-time. They collect data at one of three levels:

- VPC: All traffic to/from/inside the VPC
- Subnet: To/from/inside the subnet
- ENI: To/from the ENI



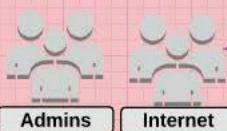
```
version
account-id
interface-id
srcaddr
dstaddr
srcport
dstport
protocol
packets
bytes
start
end
action
log-status
```

What Isn't Logged?

DHCP
 AWS DNS
 Metadata
 License Activation Requests

AWS
 Public
 Zone

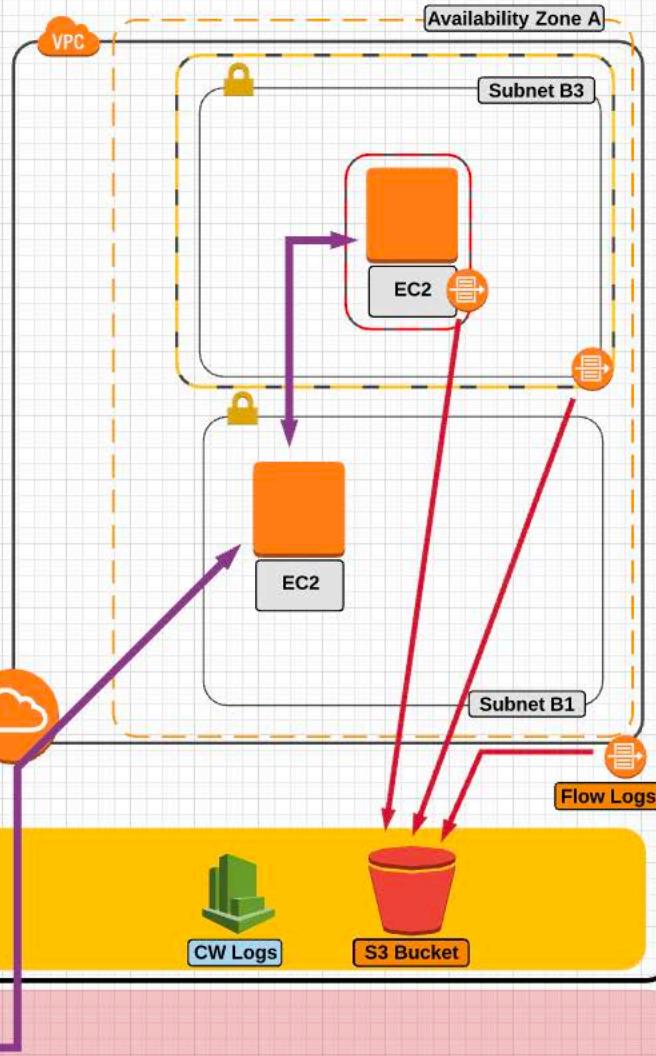
Public Internet



IAM roles provide permissions to the VPC Flow Logs service, either for CloudWatch log injection or for sending log files to an S3 bucket for storage.



IAM
 Role



VPC Flow Logs gather *metadata* for traffic flowing across a VPC. They don't log actual traffic, nor are they real-time. They collect data at one of three levels:

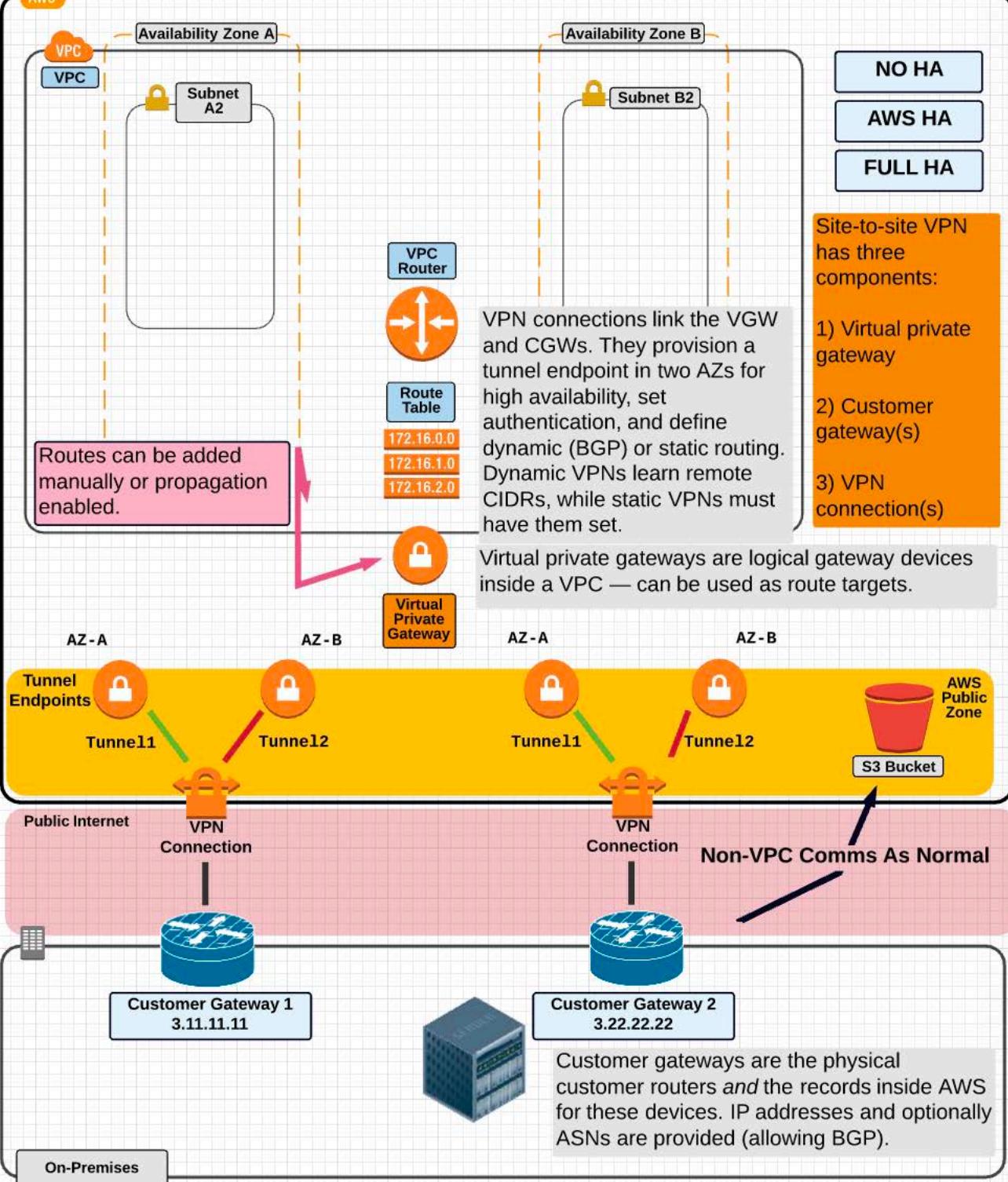
- VPC: All traffic to/from/inside the VPC
- Subnet: To/from/inside the subnet
- ENI: To/from the ENI



us-east-1

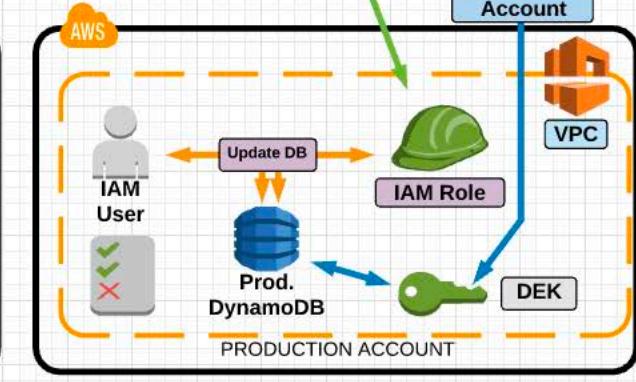
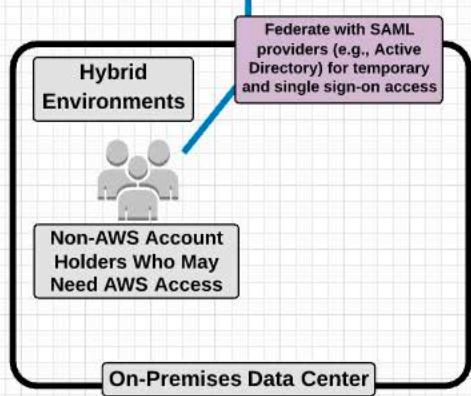
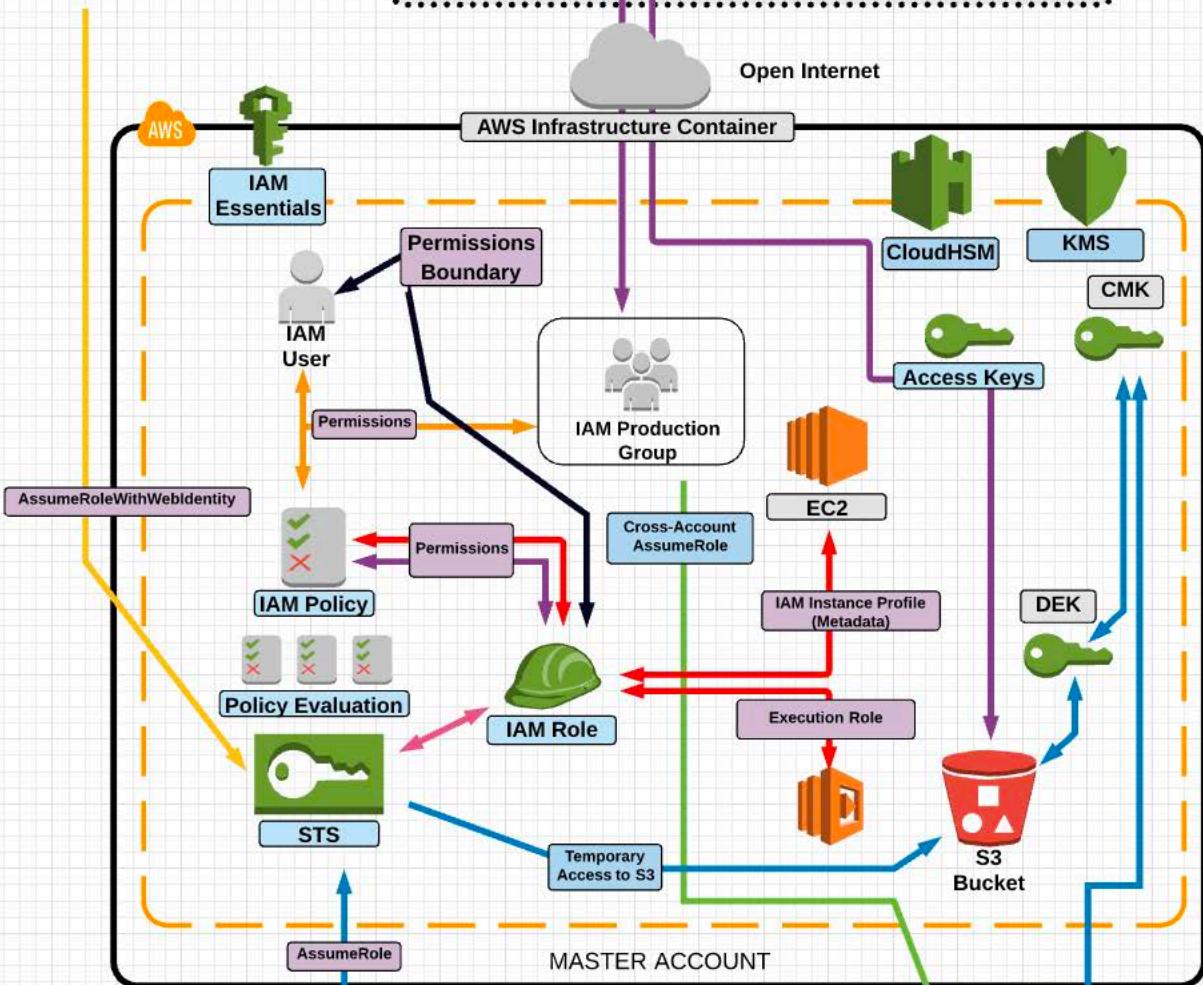
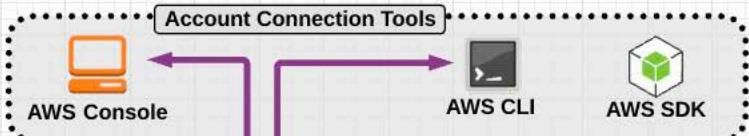
ap-southeast-2

AWS





Web Application



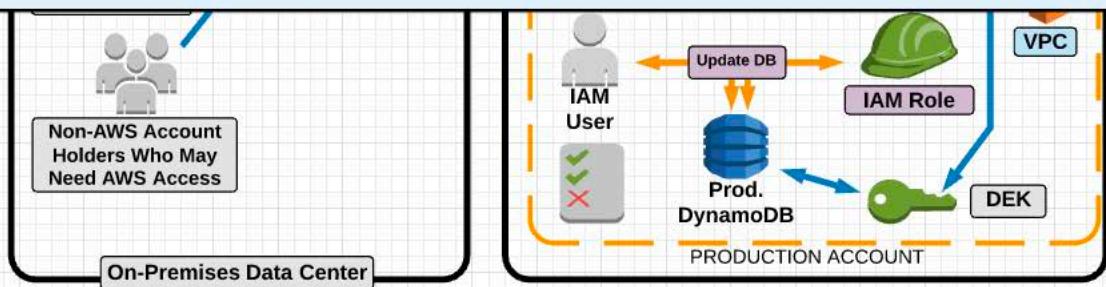


Identity & Access Management (IAM)

X

IAM Essentials

- **IAM (Identity & Access Management)** allows you to control access to AWS services and resources. It lets you manage identities and allocate permissions for different roles, users, or organizations. As a global service, the pool of identities is shared across all regions.
- IAM is used to manage:
 - Users
 - Groups
 - Roles
 - IAM Policies
 - Authentication Attributes: Usernames, Passwords, Access Keys, MFA, and Password Policies
- By default, any new IAM user you create in an AWS account has no permissions policies attached.
- All permissions policies have an implicit deny. If there is no **allow** in the policy, it is equivalent to an implicit **deny**.
- A **deny** always overrides any **allow**.
- For all users (except the root user), permissions must be given that grant access to AWS services. This is done through IAM policies.
- IAM provides identity services but also coordinates with STS to allow identity federation (the use of external identities) to access AWS resources.
- IAM Best Practices:
 - Delete your root access keys.
 - Activate MFA on your root account.
 - Create and use an IAM user with admin privileges instead of the root account.
 - Create individual IAM users.
 - Use groups to assign permissions.
 - Follow the principle of least privilege.
 - Apply an IAM password policy.





Identity & Access Management (IAM)

IAM Permissions and Policies

- A **policy** is a document that formally states one or more permissions.
- By default, all permissions are *implicitly denied*.
- An **explicit deny** always overrides an **explicit allow**.
- IAM provides pre-built policy templates to assign to users and groups. Examples include:
 - **Administrator Access**: Full access to *all* AWS resources
 - **Power User Access**: Admin Access, except it does not allow user/group management
 - **Read-Only Access**: Only view AWS resources (e.g., user can only view what is in an S3 bucket)
- You can also create custom IAM permissions policies using the **visual editor** or from scratch.
- More than one policy can be attached to a user or group at the same time.
- Policies cannot be directly attached to AWS resources (such as an EC2 instance). You must use roles for this.

IAM Policy Example: EC2 Full Access

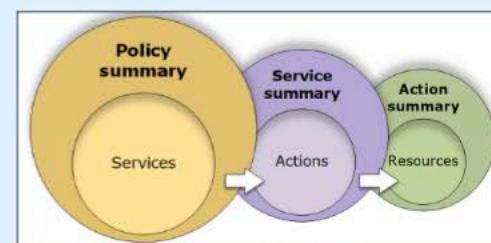
Service	Access level	Resource	Request condition
Allow (6 of 142 services) Show remaining 136			
Auto Scaling	Full access	All resources	None
CloudWatch	Full access	All resources	None
EC2	Full access	All resources	None
ELB	Full access	All resources	None
ELB v2	Full access	All resources	None
IAM	Limited: Write	All resources	Multiple

Boundaries

Boundaries that permissions can't exceed

Access Advisor

Shows what permissions exist and the last access time



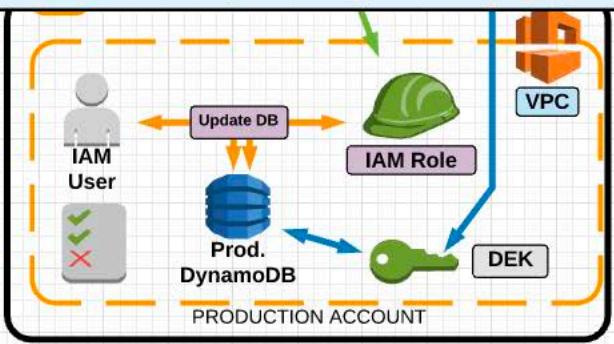
Hybrid Environments

Directory for temporary and single sign-on access



Non-AWS Account Holders Who May Need AWS Access

On-Premises Data Center

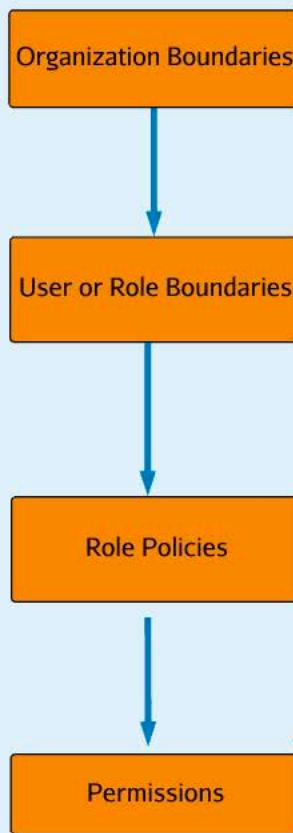




Identity & Access Management (IAM)



IAM Policy Evaluation

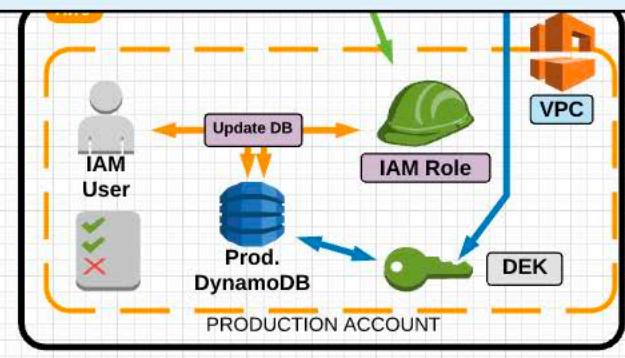
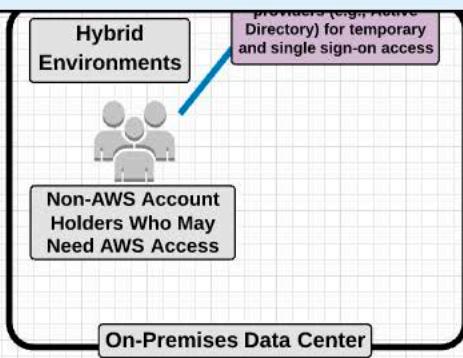


DENY → ALLOW → DENY

Boundaries are always processed first, starting with organization and then identity (user or role).

Then, AWS checks if you have chosen a subset of permissions for a `sts:AssumeRole`.

The final effective permissions are a merge of identity, resource, and ACL.





Identity & Access Management (IAM)

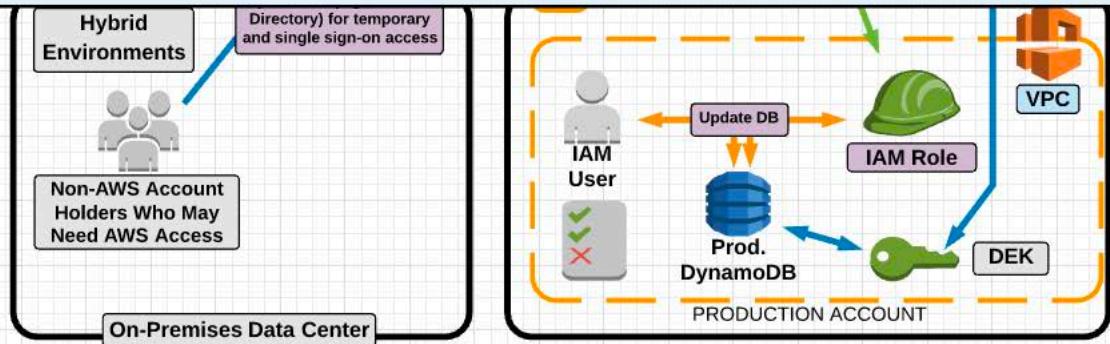
IAM Security Token Service (STS)

- STS allows you to create temporary security credentials that grant trusted users access to your AWS resources.
- These temporary credentials are for short-term use, with a configurable session duration that can be set anywhere between 15 minutes and 36 hours.
- Once expired, the credentials can no longer be used to access your AWS resources.
- An STS API call returns a credential object containing:
 - A session token
 - An access key ID
 - A secret access key
 - An expiration timestamp

When to use STS:

- Identity Federation
 - Enterprise identity federation (authenticate through your company's network).
 - STS supports SAML (Security Assertion Markup Language), which allows for the use of Microsoft Active Directory or your own solution.
 - Web identity federation (third-party identity providers like Facebook, Google, and Amazon).
- Roles for Cross-Account Access
 - Used for organizations that have more than one AWS account.
- Roles for Amazon EC2 (and Other AWS Services)
 - Allow an application running on an EC2 instance to access other AWS services without having to embed credentials.

For mobile applications, AWS recommends using Cognito rather than STS. Cognito provides additional mobile-specific functionality that makes the flow easier to manage.

[AssumeRole](#)[AssumeRoleWithWebIdentity](#)[AssumeRoleWithSAML](#)[Example](#)

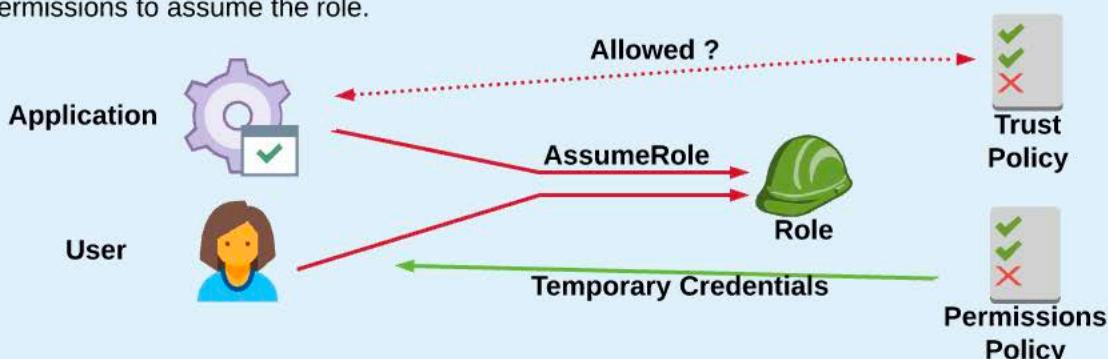


IAM Roles

Like an IAM user, an IAM role is an entity that can be granted permissions to interact with AWS resources. However, unlike IAM users, roles are designed to be assumed by anyone who needs to make use of their permissions.

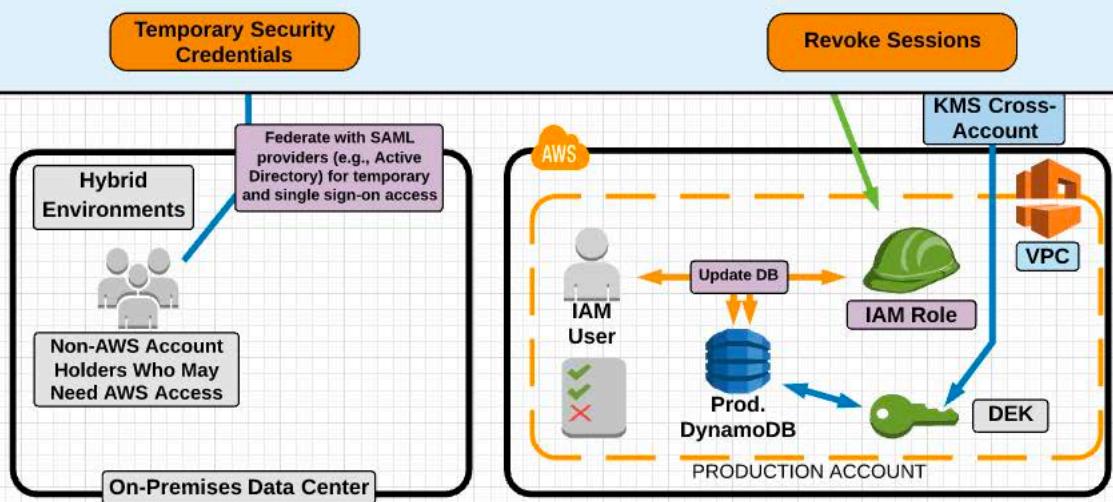
A role has two components: a **trust** policy, which defines the circumstances under which the role can be assumed, and a **permissions** policy, which defines the AWS access rights granted during that AssumeRole.

A role has no long-term access credentials (username, password, or key pairs) of its own. Instead, when the role is assumed, STS generates temporary credentials, which can last anywhere from a few minutes to several hours. Once the credentials expire, they are no longer valid. Temporary credentials can be renewed, even before expiration, and will be granted for as long as the requesting identity has permissions to assume the role.



Roles are used extensively in AWS. You grant permissions to services via roles. For example, Lambda gains permissions via its execution role, EC2 is given permissions via an instance role, and many services that use CloudWatch for logging are provided those services via roles.

When an entity assumes a role, it becomes that role. If you assume a role in an external AWS account and then interact with that account, that account owns any resources you create. You are an entity in that account for as long as you've assumed the role.





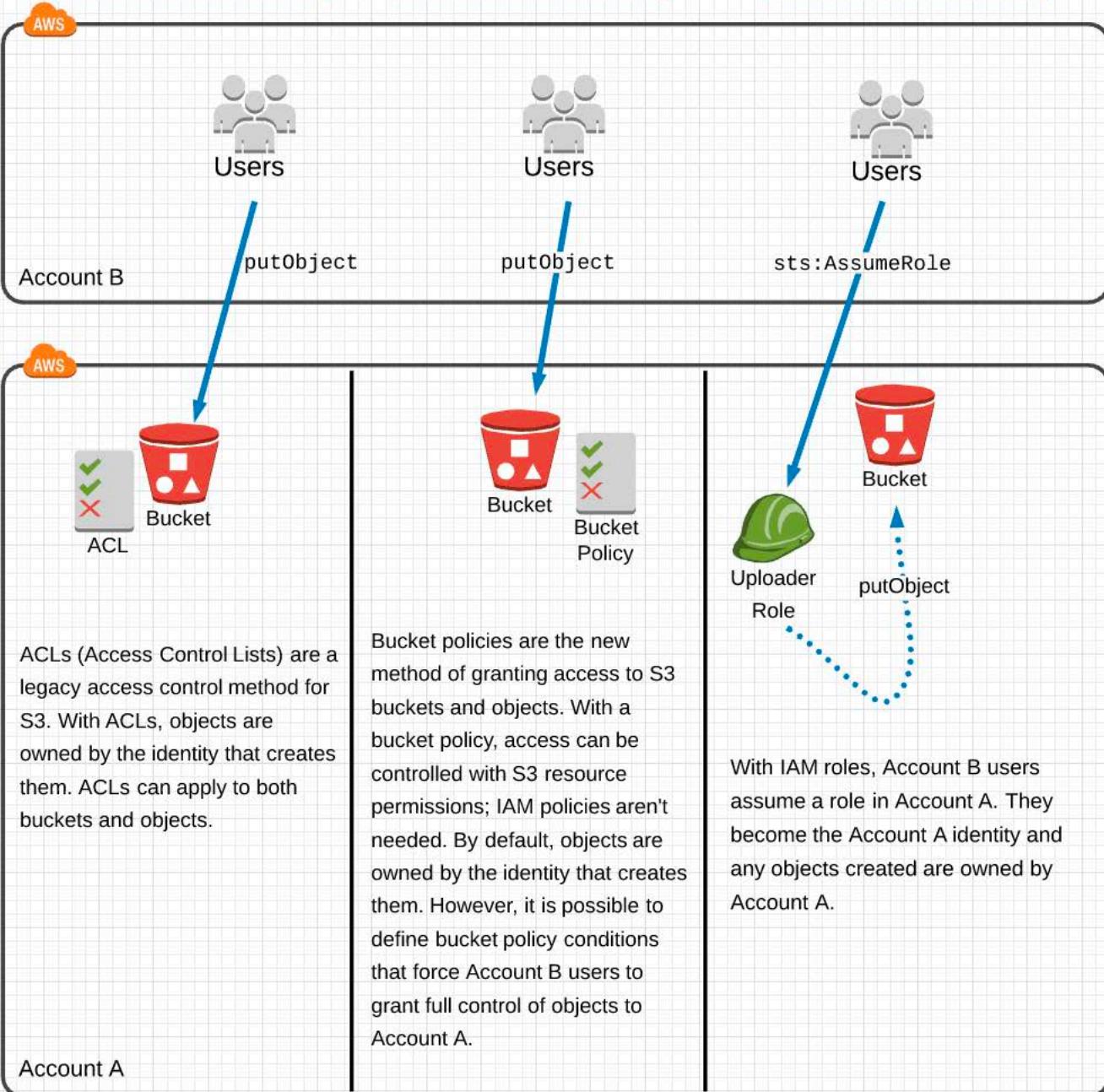
Cross-Account Access to S3 Buckets and Objects

There are three main ways to provide access to your S3 buckets from external AWS accounts: IAM Roles, Bucket Policies, and Bucket ACLs.

ACL

Bucket Policy

IAM Role





CloudHSM and On-Premises HSM

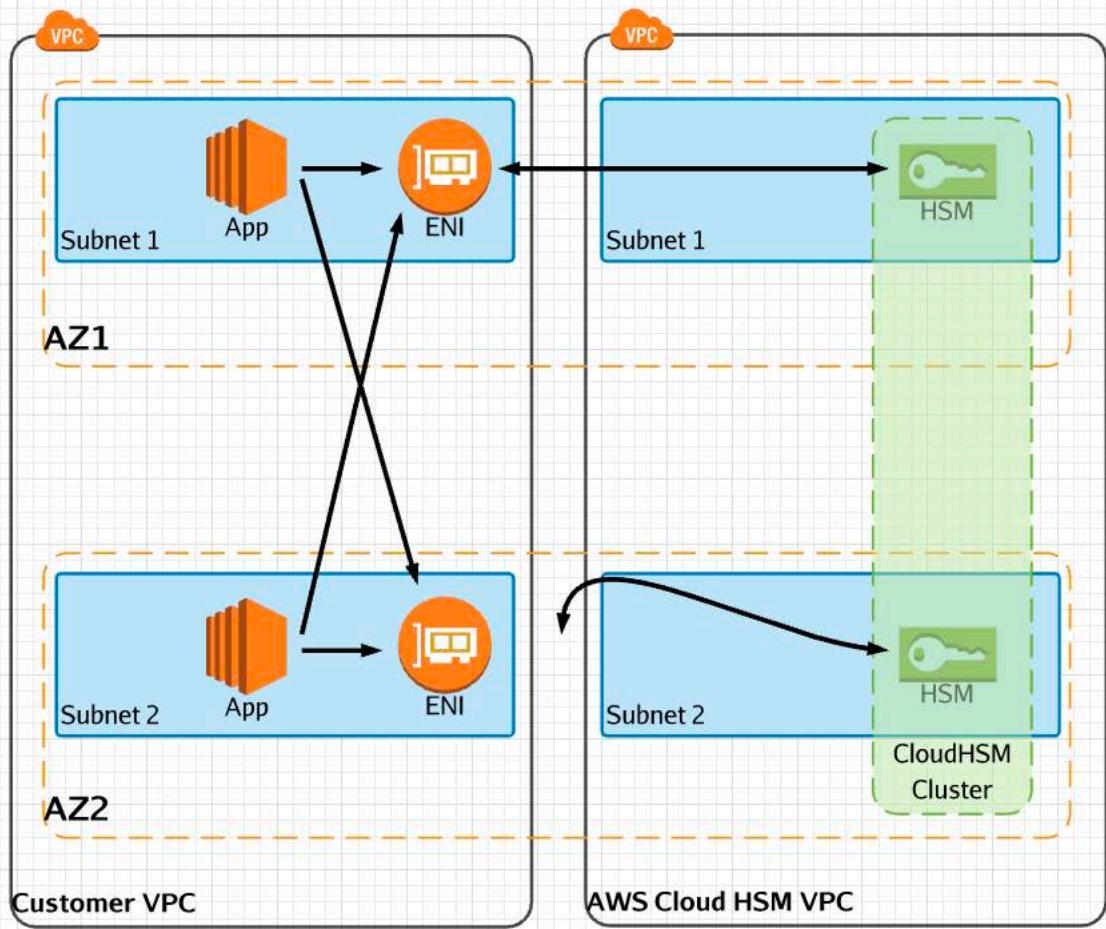
CloudHSM is a dedicated HSM that runs within your VPC, accessible only to you in a single-tenant architecture. AWS manages and maintains hardware but has **no** access to the cryptographic component. Interaction with CloudHSM is via industry-standard APIs, **not normal AWS APIs**.

- PKCS #11, Java Cryptography Extensions (JCE), Microsoft CryptoNG (CNG)

Keys can be transferred between CloudHSM and other hardware solutions (on-premises). Keys are shared between cluster members. No HA unless multiple HSMs are provisioned.

Applications can be **outside** the VPC — Direct Connect, peered, or VPN.

On-premises HSM is for if you **really** need to **control your own physical hardware**.





Identity & Access Management (IAM)

IAM Access Keys

Access keys are used to sign programmatic requests to AWS. They are required when making calls to AWS from:

- The AWS command line interface (CLI)
- Tools for Windows PowerShell
- AWS SDKs

Access Key Architecture

Access keys are made up of two parts: an **access key ID** and a **secret access key**.

Access Key ID: AKIAIOSFODNN7EXAMPLE

Secret Access Key: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY

The access key ID is public. The secret access key is sensitive and should be stored securely. You will need both parts to interact with AWS from the CLI or API.

Important Facts about Access Key Pairs

- Access keys are associated with IAM users, *not* groups or roles.
- They can be generated *only once*. The secret access key is *not* stored by AWS. If lost, the access key will need to be regenerated.
- AWS will *not* regenerate the same access key. Once an access key is lost, it cannot be recovered.
- Access keys are long-term credentials. They don't expire and can only be enabled, disabled, or deleted.
- IAM roles do not have associated access keys; they use short-term credentials.
- IAM users can only have two key pairs at a time. If you require new credentials and already have two key pairs, you must deactivate one of your current keys and generate a new one.
- Never create or store access keys on an EC2 instance or GitHub repository.
- Access keys grant access to whatever actions and resources the associated IAM user has access to.
- Access keys provide access to AWS from anywhere an API connection is possible.

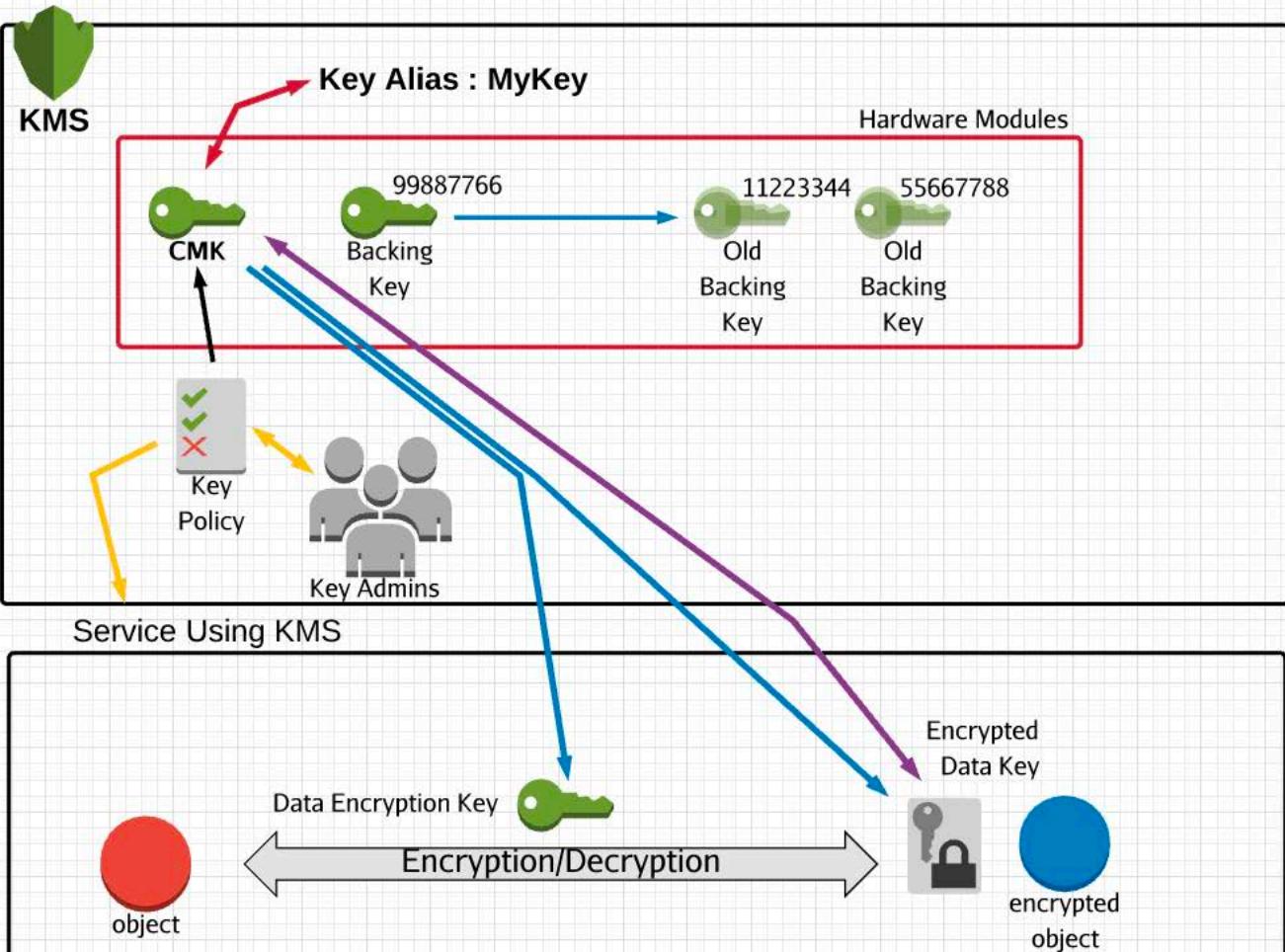


Key Management Service (KMS)

KMS is a key management service that uses FIPS 140-2 compliant hardware modules to manage access to key material. It integrates fully with IAM and CloudTrail for permissions management and auditing functions. KMS can be optionally used with most AWS services that support encryption.

KMS manages customer master keys (CMKs), with which it can encrypt and decrypt small amounts of data. It can also use CMKs to generate data encryption keys (DEKs), which AWS services can use to encrypt and decrypt data.

All interaction with KMS is via AWS services or AWS APIs. KMS does not support industry standard key management APIs but integrates well with AWS services.





Identity & Access Management (IAM)

IAM Access Keys

Access keys are used to sign programmatic requests to AWS. They are required when making calls to AWS from:

- The AWS command line interface (CLI)
- Tools for Windows PowerShell
- AWS SDKs

Access Key Architecture

Access keys are made up of two parts: an **access key ID** and a **secret access key**.

Access Key ID: **AKIAIOSFODNN7EXAMPLE**

Secret Access Key: **wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY**

The access key ID is public. The secret access key is sensitive and should be stored securely. You will need both parts to interact with AWS from the CLI or API.

Important Facts about Access Key Pairs

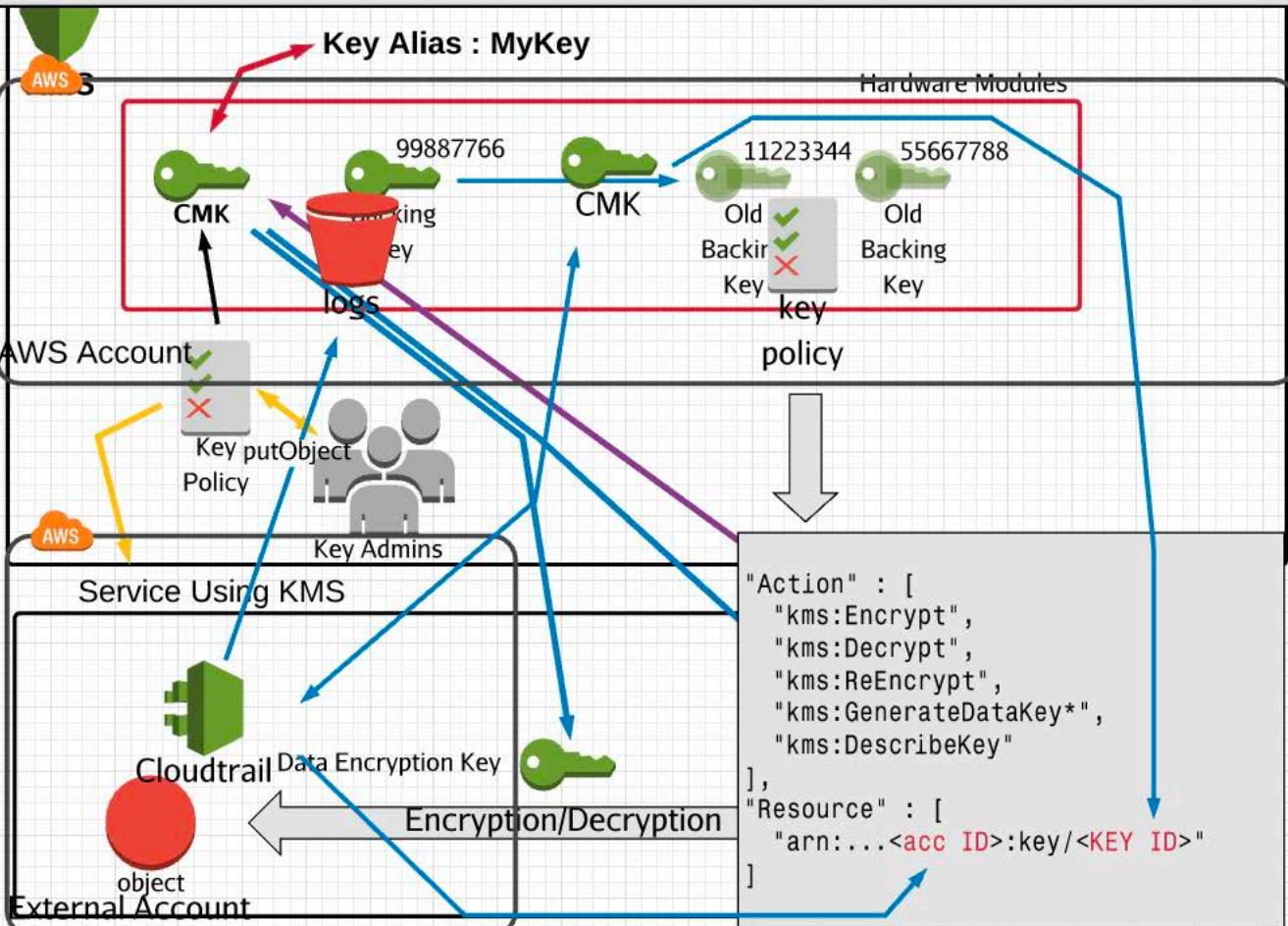
- Access keys are associated with IAM users, *not* groups or roles.
- They can be generated *only once*. The secret access key is *not* stored by AWS. If lost, the access key will need to be regenerated.
- AWS will *not* regenerate the same access key. Once an access key is lost, it cannot be recovered.
- Access keys are long-term credentials. They don't expire and can only be enabled, disabled, or deleted.
- IAM roles do not have associated access keys; they use short-term credentials.
- IAM users can only have two key pairs at a time. If you require new credentials and already have two key pairs, you must deactivate one of your current keys and generate a new one.
- Never create or store access keys on an EC2 instance or GitHub repository.
- Access keys grant access to whatever actions and resources the associated IAM user has access to.
- Access keys provide access to AWS from anywhere an API connection is possible.



KMS in a Multi-Account Configuration

CMKs can be configured to allow other accounts to use them. The key won't appear in the external account, but if it is configured using a key policy, that account can interact with the key for cryptographic functions.

Remember: Key *usage* and key *admin* are not the same thing!





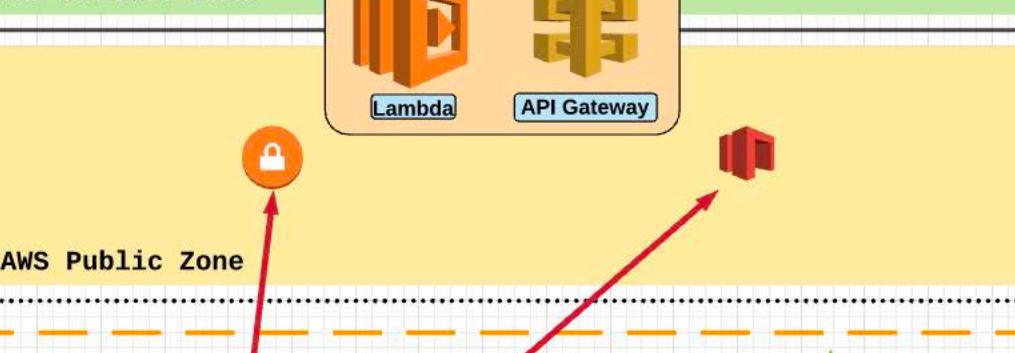
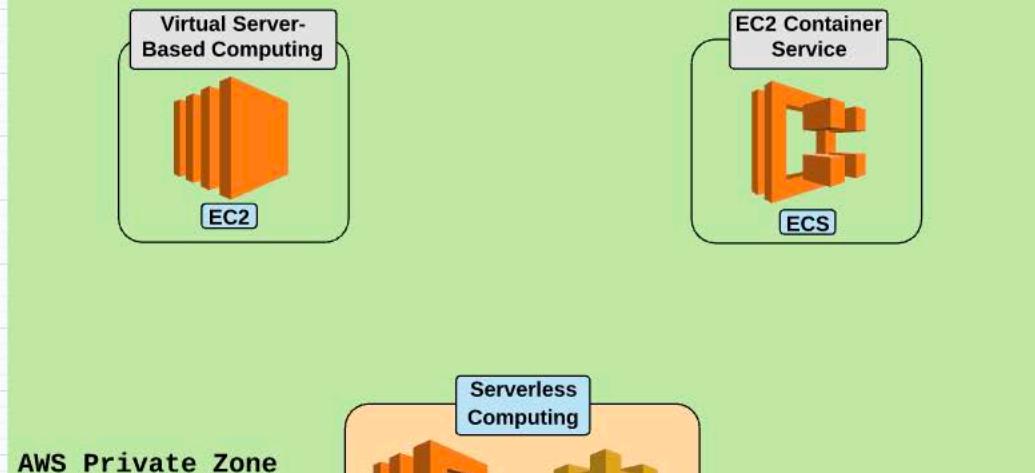
AWS

AWS Infrastructure "Container"

AWS Account
(i.e., Production Account)

AWS Compute Services

VPC



Public Internet

On-Premises Data Center



Corp Users



On-Premises Servers



Customer Router



Virtual Private Network



AWS Storage Gateway



AWS Direct Connect

Direct Connect Location



Partner Router



DX Endpoint



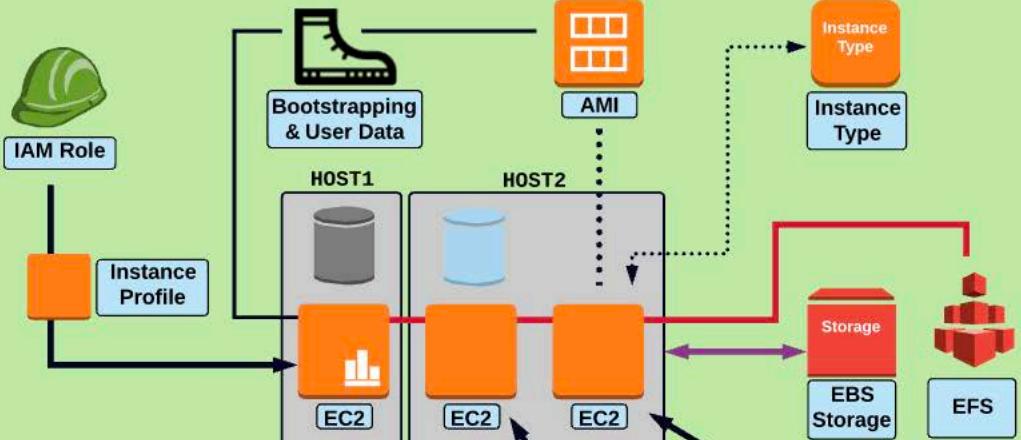
AWS

AWS Infrastructure "Container"

AWS Account
(i.e., Production Account)

VPC

AWS Private Zone



Placement Groups



SSM



Key Pair (.pem)



Network Interfaces



Elastic IP



CW Logs CloudWatch

AWS Public Zone



AZ-A

AWS Private Zone

AZ-B



Cluster Placement Group



Cluster placement groups exist in a single AZ and are created to allow high speed and low latency connectivity between instances in the group.

Cluster placement groups reserve physical capacity, keeping instances in physical vicinity. The chances of capacity-related failure increases if different types of instances are used or instances are added at separate times.

Partition Placement Group



Designed to spread large infrastructure sets across infrastructure in isolated fault domains. Used when you need to ensure HA but need physical location control.

Spread Placement Group



Designed for a small number of critical components where you *need* to ensure separation. A spread placement group can be single-AZ or multi-AZ.

Availability Zone

Availability Zone



EC2 Instance Profiles and Instance Roles (IAM Role)

IAM roles provide short-term or temporary access credentials within AWS when used with the sts:AssumeRole API call. Identities within AWS can be granted access to assume roles using that roles TRUST policy.

Instance profiles allow a role to be assigned to a single EC2 instance and for applications running on that instance to assume a role while being abstracted away from any AWS identity. Applications running on an EC2 instance are not "AWS identities," and so it's the instance profile which bridges that gap.

IAM role credentials can be obtained from the EC2 instance metadata if a role is attached via an instance profile:

<http://169.254.169.254/latest/meta-data/iam/security-credentials/ROLENAMES>

The credentials returned take the form of a JSON structure:

```
{  
  "Code" : "Success",  
  "LastUpdated" : "2012-04-26T16:39:16Z",  
  "Type" : "AWS-HMAC",  
  "AccessKeyId" : "ASIAIOSFODNN7EXAMPLE",  
  "SecretAccessKey" : "wJalrXUtnFEMI/K7MDENG/bPxRficyEXAMPLEKEY",  
  "Token" : "token",  
  "Expiration" : "2017-05-17T15:09:54Z"  
}
```

Credentials are automatically renewed when expired using calls to the instance metadata. Explicitly defined credentials take priority for the CLI tools if present on EC2 instances.

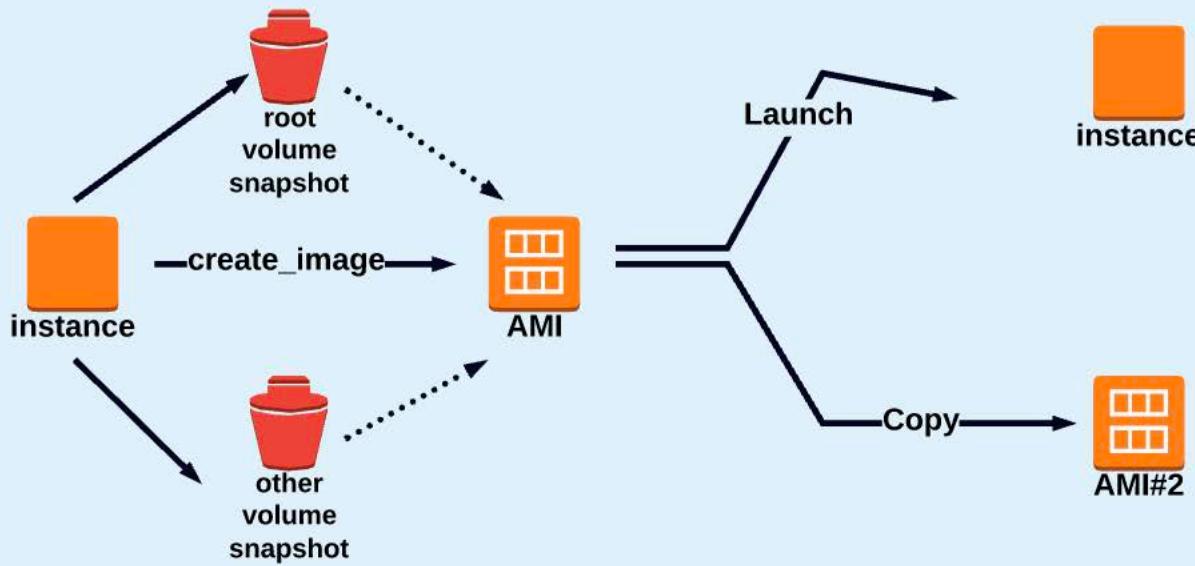
Only a single instance profile can be associated with an EC2 instance, containing a single EC2 role. If associating a role via the console, the instance profile is automatically created and associated. If using APIs, the CLI ,or CloudFormation, the two steps are distinct and must be done explicitly.

Roles (via profiles) can be assigned at the time of creating the instance or afterwards via the console, CLI, or APIs. All applications running on the instance share the role credentials — it's not possible to be more granular.



Amazon Machine Images (AMIs)

AMIs are objects containing all the information required to launch an instance, the owner of the AMI, launch permissions (public, explicit, and/or implicit), the architecture and operating system, and a block device mapping of all volumes required.

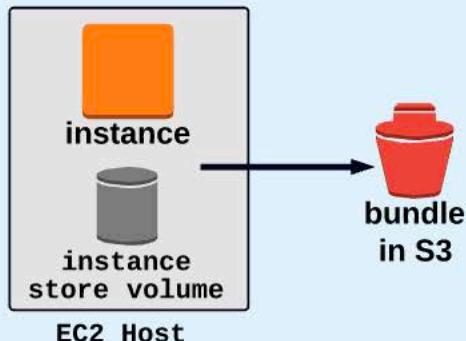


AMIs contain mappings to any volumes, and these mappings reference EBS volume snapshots in the same region.

AMIs are regional based but can be copied between regions, which also copies any volume snapshots.

By default, an AMI has an implicit permission allowing use by the creator. An AMI can be set to public, or accounts can be granted explicit rights to use the AMI and any associated volume snapshots.

AMIs can be sold. Users of those AMIs pay the base EC2 charges for the instance they create from the AMI, in addition to charges set by the creator.



When creating AMIs from instances with instance store root volumes, the process is slightly different — a bundle is created and stored on S3 containing ALL the root vol data.



	EBS only	NVMe EBS	Instance store	Placement group	Enhanced networking
A1	Yes	Yes	No	Yes	ENa
C4	Yes	No	No	Yes	Intel 82599 VF
C5	Yes	Yes	No	Yes	ENa
C5d	No	Yes	NVMe *	Yes	ENa
C5n	Yes	Yes	No	Yes	ENa
D2	No	No	HDD	Yes	Intel 82599 VF
F1	No	No	NVMe *	Yes	ENa
G3	Yes	No	No	Yes	ENa
H1	No	No	HDD	Yes	ENa
I3	No	No	NVMe *	Yes	ENa
M4	Yes	No	No	Yes	m4.16xlarge: ENA All other sizes: Intel 82599 VF
M5	Yes	Yes	No	Yes	ENa
M5a	Yes	Yes	No	Yes	ENa
M5d	No	Yes	NVMe *	Yes	ENa
P2	Yes	No	No	Yes	ENa
P3	p3dn.24xlarge: No	p3dn.24xlarge: Yes	p3dn.24xlarge: NVMe *	Yes	ENa
	All other sizes: Yes	All other sizes: No			
R4	Yes	No	No	Yes	ENa
R5	Yes	Yes	No	Yes	ENa
R5a	Yes	Yes	No	Yes	ENa
R5d	No	Yes	NVMe *	Yes	ENa
T2	Yes	No	No	No	No
T3	Yes	Yes	No	No	ENa
u-xtb1.metal	Yes	Yes	No	No	ENa
X1	No	No	SSD	Yes	ENa
X1e	No	No	SSD	Yes	ENa
z1d	No	Yes	NVMe *	Yes	ENa

Instance Types: General Purpose, Compute Optimized, Memory Optimized, Storage Optimized, Accelerated Computing, Burstable

Accelerated Computing: GPU and FPGA

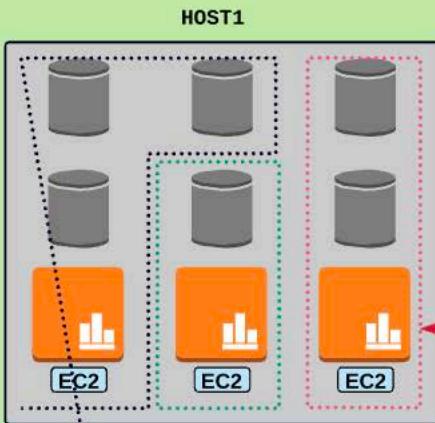
NITRO, NITRO, NITRO — if in doubt, the preference is Nitro.



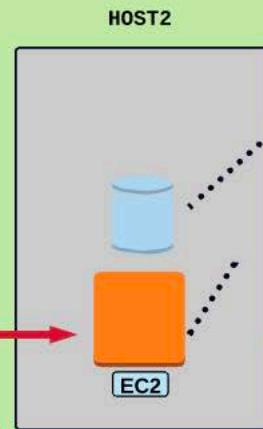
AWS AWS Infrastructure "Container"

VPC

AWS Private Zone



EC2 hosts have physical storage attached that can be made available to instances ON that host. The volumes are known as ephemeral23.



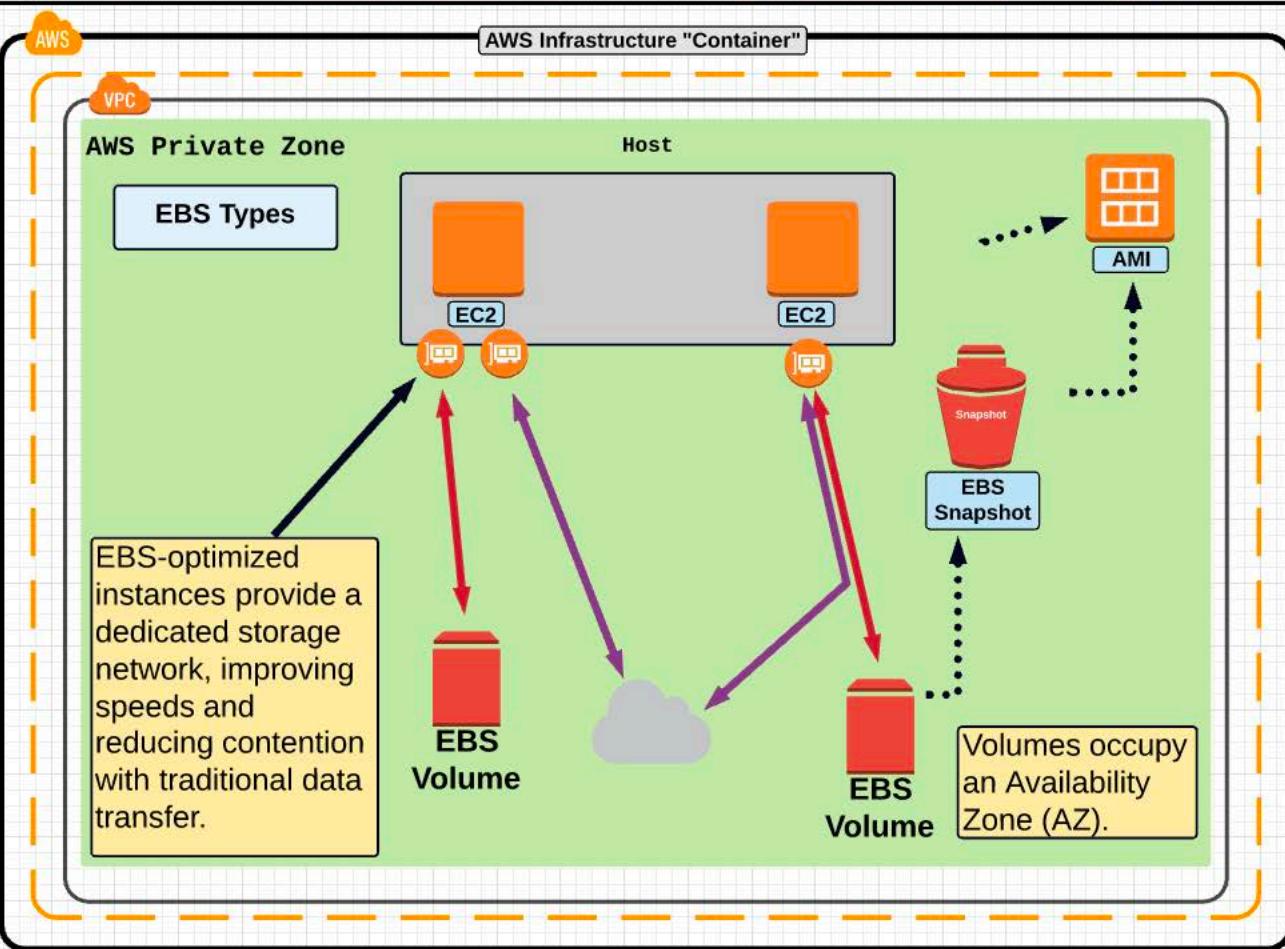
If instances stop and start (switching hosts), terminate, or if hardware fails, the data on instance store volumes is lost — it's ephemeral.

Ideal Patterns

- Temporary Storage:** For any non-persistent storage needs, instance store volumes are included in the instance cost.
- High I/O & Throughput:** Instance store offers direct attached storage. Depending on instance type, this offers the highest level of performance available for AWS storage.

Anti-Patterns

- Shared Storage:** Volumes are attached to one instance only and can't be shared across instances.
- Persistence:** Volumes will be removed from instances if that instance moves between hosts (stop/start).
- Durability/Resilience:** No support for snapshots and volumes are single physical disks, so they can and do fail.
- Elasticity:** Storage is directly tied to physical devices and can't easily be scaled.

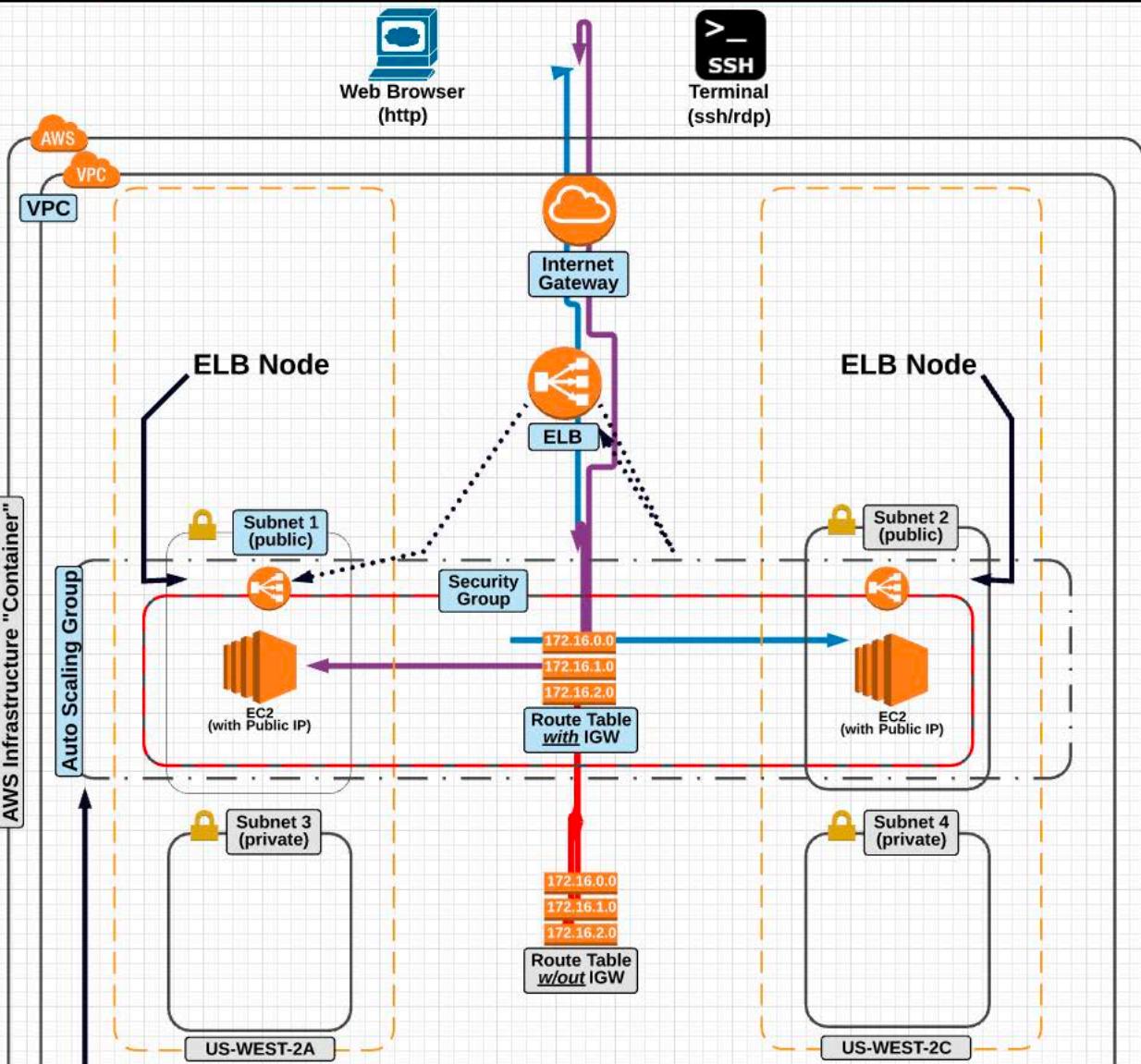


Ideal Patterns

- Persistence:** A requirement for data to exist beyond the lifecycle of a given instance and able to tolerate reboots/instance failure.
- Durability:** Data that requires snapshots or resilience. Note: EBS is not the most durable storage service.
- Elasticity:** Requirements for variable or changing performance demands over time.
- Provisioned Performance:** Certain EBS volumes can provide guaranteed performance levels.

Anti-Patterns

- Temporary Storage:** Instance store would be more suited to this requirement. EBS provides persistence and has extra cost.
- Static Content Distribution:** S3 or S3 + CloudFront is more ideal for object-based distribution.
- Shared Access:** EBS volumes are attached to a single instance as block storage. They cannot be shared between instances.
- High Durability:** EBS cannot provide super high 99.5-99.9%+ durability.



Launch Configuration



IAM Role



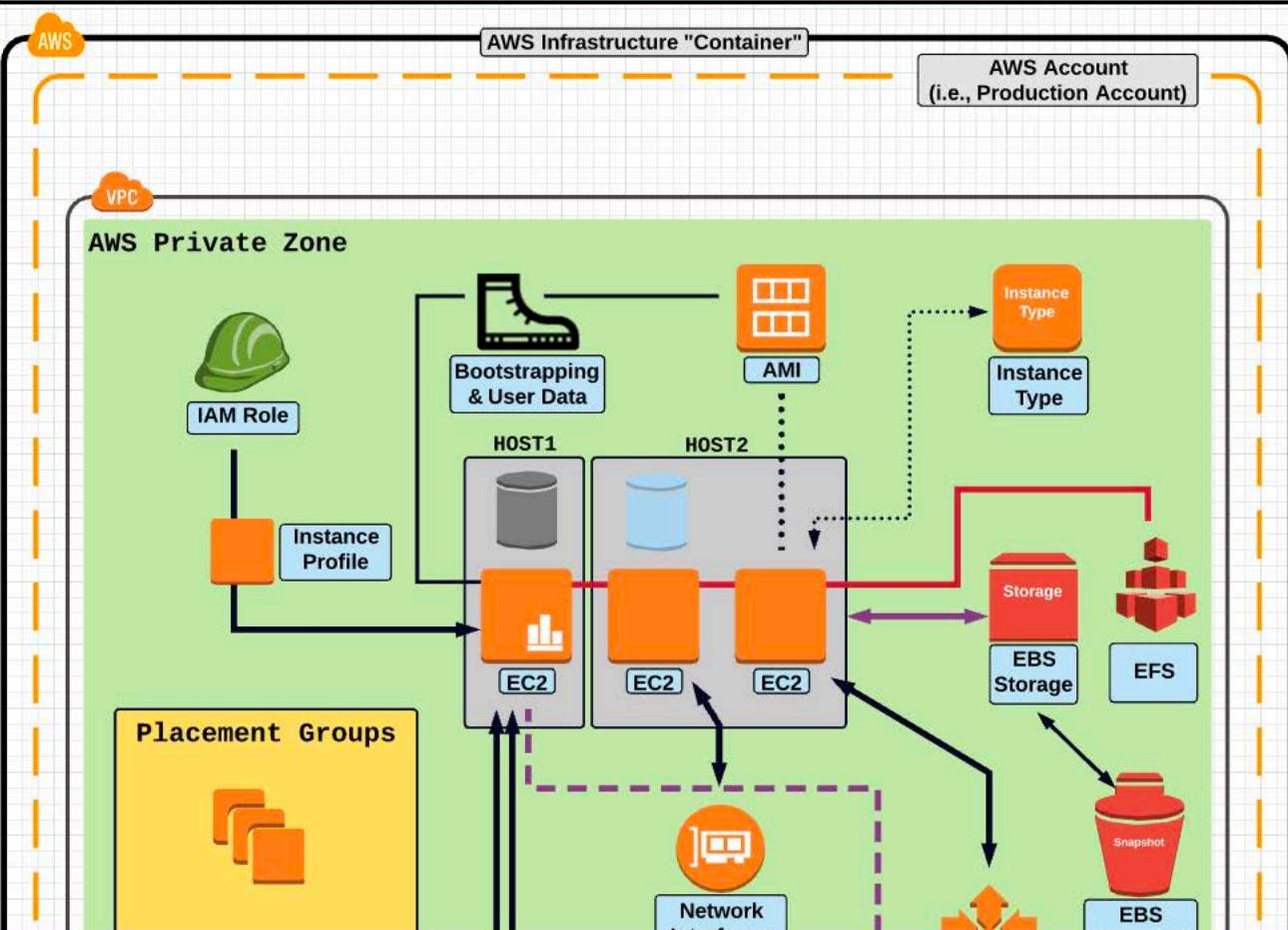
Bootstrapping & User Data



Instance Type



AMI



EBS Snapshots

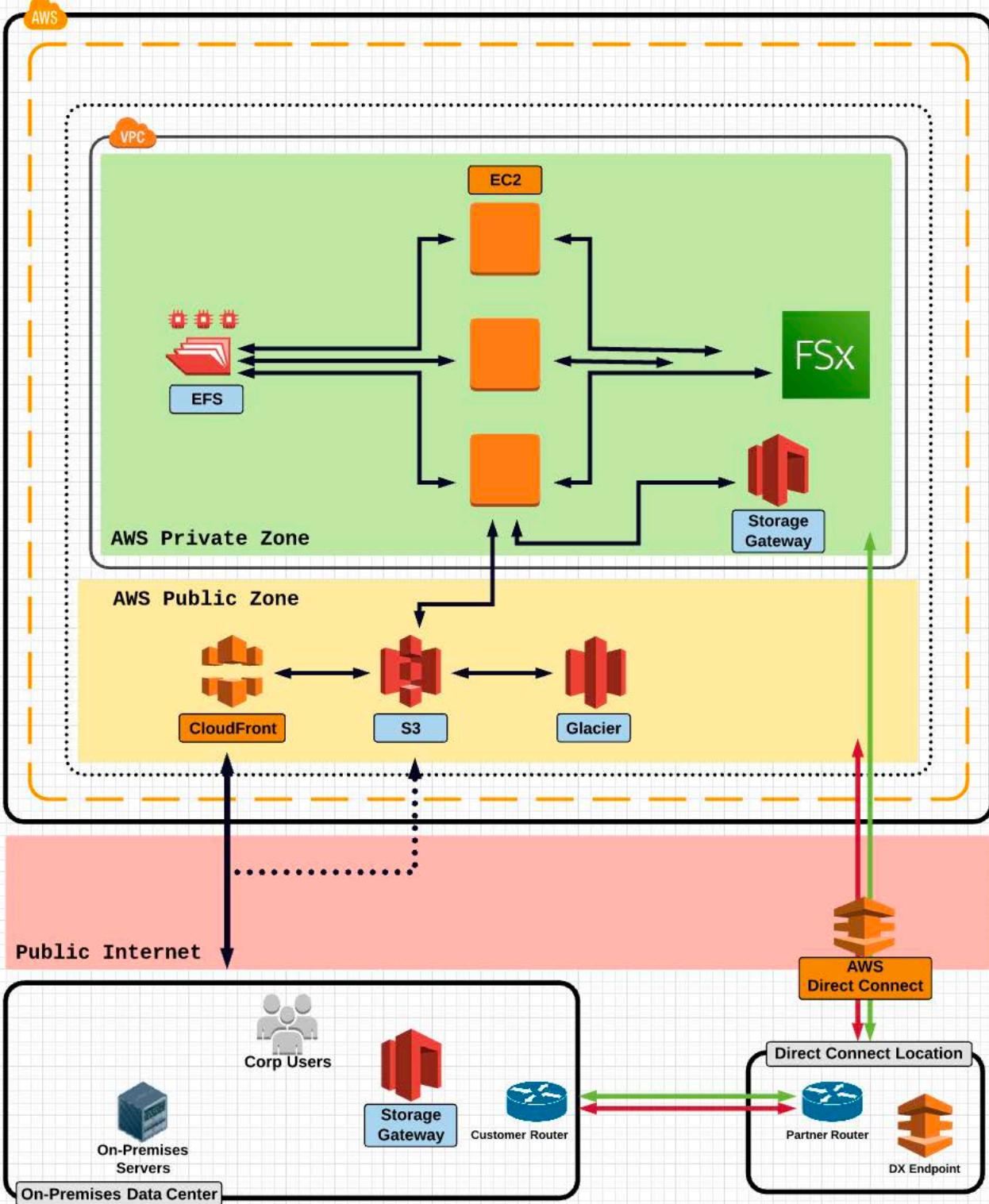


Snapshots are point-in-time backups of an EBS volume that are stored in S3. An initial snapshot contains an exact copy of the data on the EBS volume and can take some time to complete.

- Snapshots are incremental in nature.
- A snapshot only stores the changes since the most recent snapshot, thereby reducing costs (by only having to pay for storage for the “incremental changes” between snapshots).
- When deleted, blocks required to restore the other snapshots are retained.
- Snapshots can be used to create EBS volumes or used when creating AMIs.
- Snapshots are point-in-time and are known as crash-consistent. Activity should be frozen on highly transactional volumes where consistency is required.
- Ideally, for root volumes the instance should be in a **stopped** state.
- Where EBS is an AZ-scoped service, snapshots are in S3 and are region resilient.



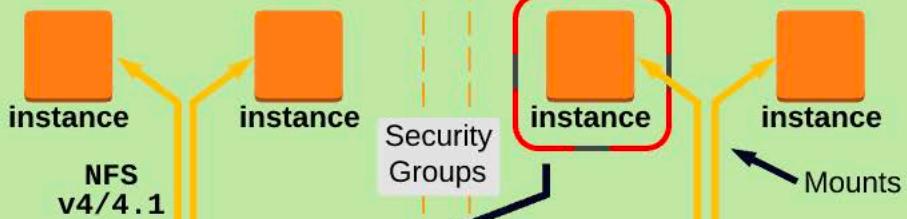
AWS





AWS

VPC



10.0.0.20
Mount Target

Availability Zone

10.0.1.27
Mount Target

Availability Zone

POSIX Permissions

EFS - FileSystem

Public Internet

On-Premises Data Center

On-Premises Servers

AWS Direct Connect

Direct Connect Location

Partner Router

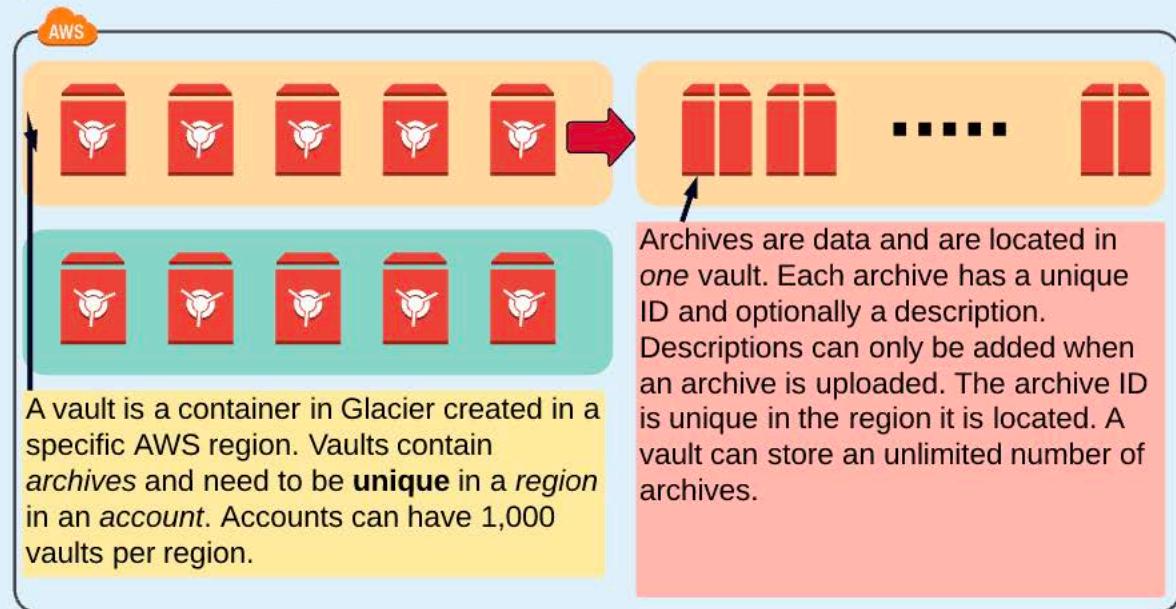
DX Endpoint



Glacier

X

The name *Glacier* is often used to refer to both the product S3 Glacier as well as the storage classes based on Glacier available within S3. S3 Glacier is an isolated product used for long-term archival storage. It's not a product you use directly. Beyond simple operations, it can only be interacted with from the command line or APIs.



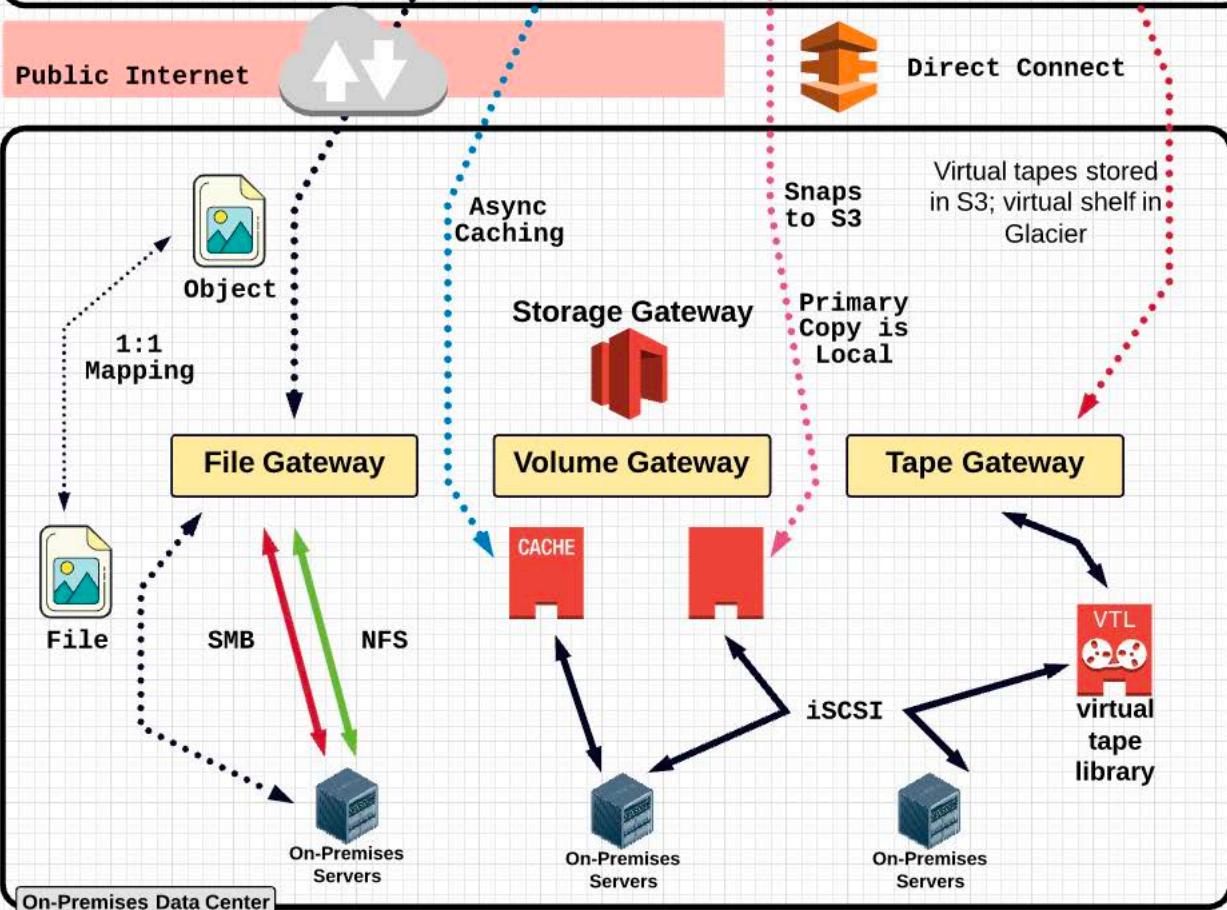
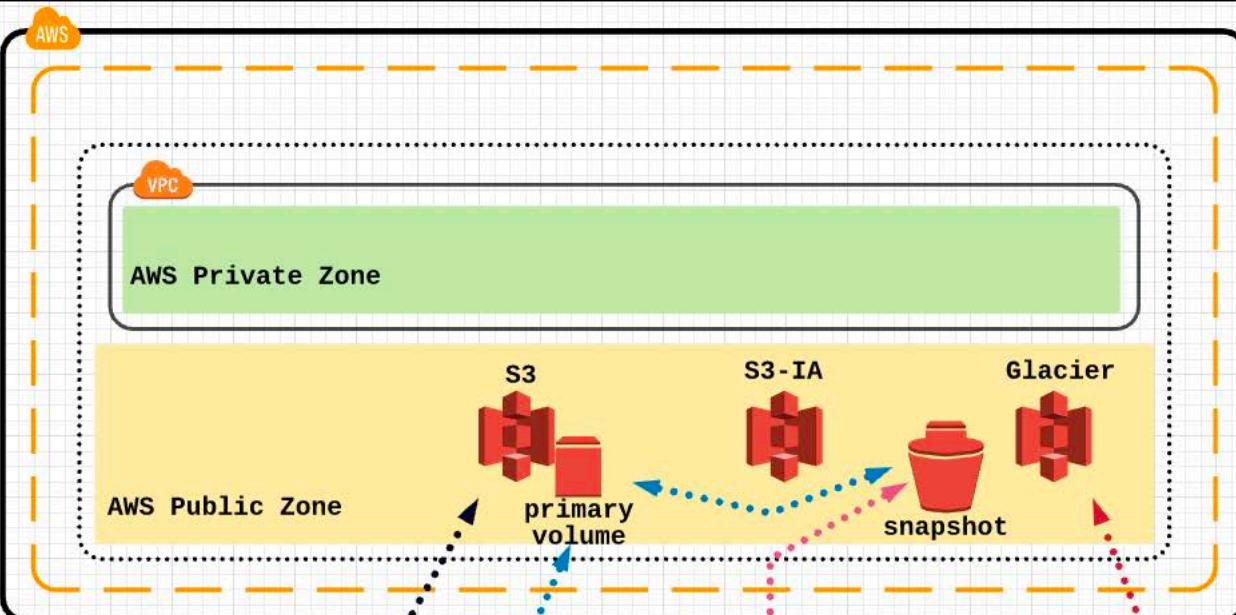
All vault operations are region specific — vaults are created in a region, and listing vaults returns vaults in that region.

Inventories of vault contents is an asynchronous operation with notifications available from SNS. Glacier performs an automatic inventory of every vault every 24 hours. A vault inventory lists archive IDs, creation date, and the size.

No user-definable metadata is stored with archives in Glacier. Archives don't have a name — only a unique ID and optional description set when initially uploaded. Archives cannot be edited — they can only be deleted and new ones uploaded (given a new archive ID).

Any jobs to download archives are asynchronous, and there are three speeds:

- **Expedited:** Jobs are typically completed in 1-5 minutes for anything below 250 MB archives.
- **Standard:** Jobs are completed generally within 3-5 hours.
- **Bulk:** Economic option for large amounts — completed within 5-12 hours.





AWS Service Resilience



ELBs

Scaling Architectures



Route 53

Stateless Architectures

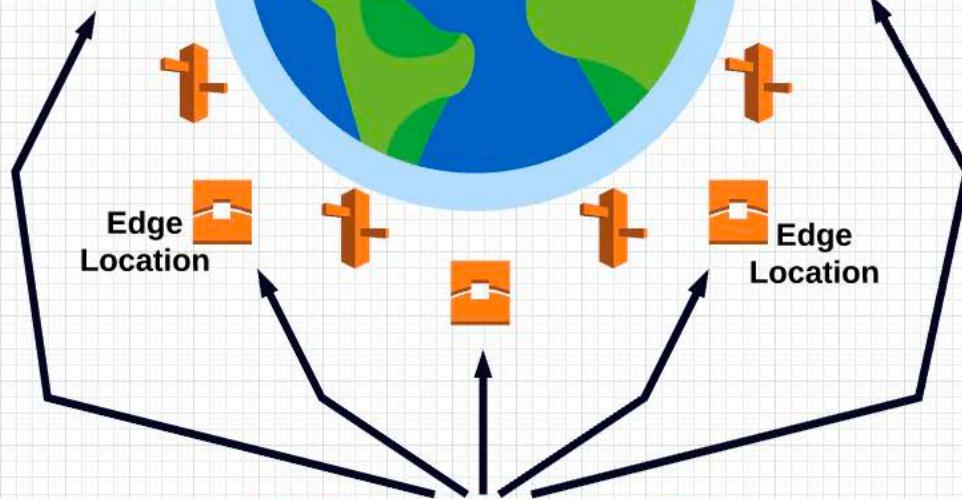


R53 Name Servers

Edge Location



Edge Location



Edge Location



Origins

CloudFront



[Back](#)[Global Infrastructure](#)[Account & Services](#)[Networking](#)[Well-Architected Appendix](#)[IAM](#)[EC2/EBS](#)[S3](#)[R53](#)[ELB/ALB](#)[VPC](#)[NAT GW](#)[ASG](#)[VPN](#)

us-east-1

IAM



us-east-1a

us-east-1b

us-east-1c

us-west-1



us-west-1a

us-west-1b

us-west-1c

IAM is global — same data globally, accessible from all regions.



IAM

EC2/EBS

S3

R53

ELB/ALB

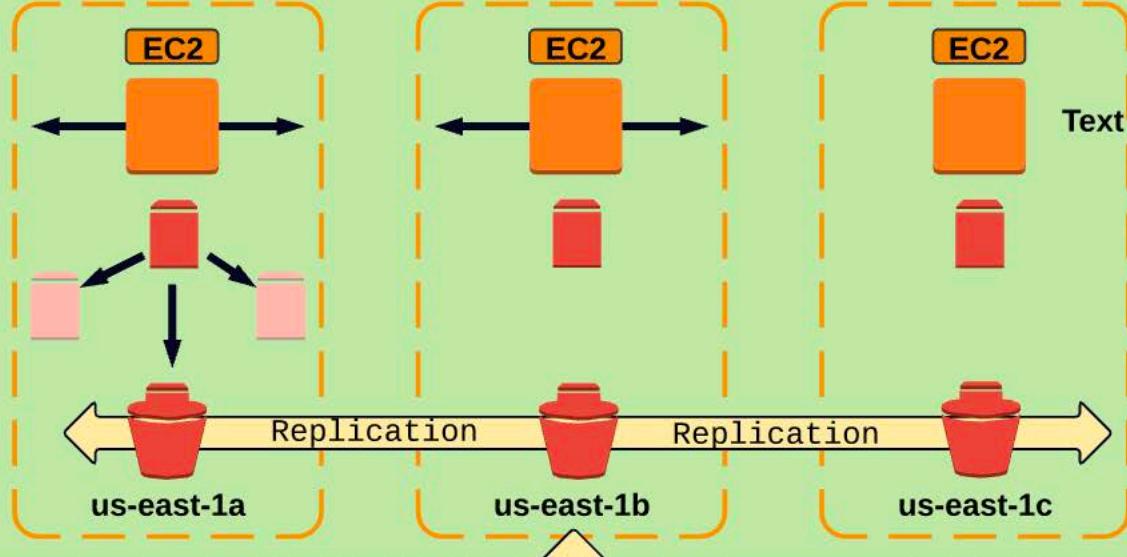
VPC

NAT GW

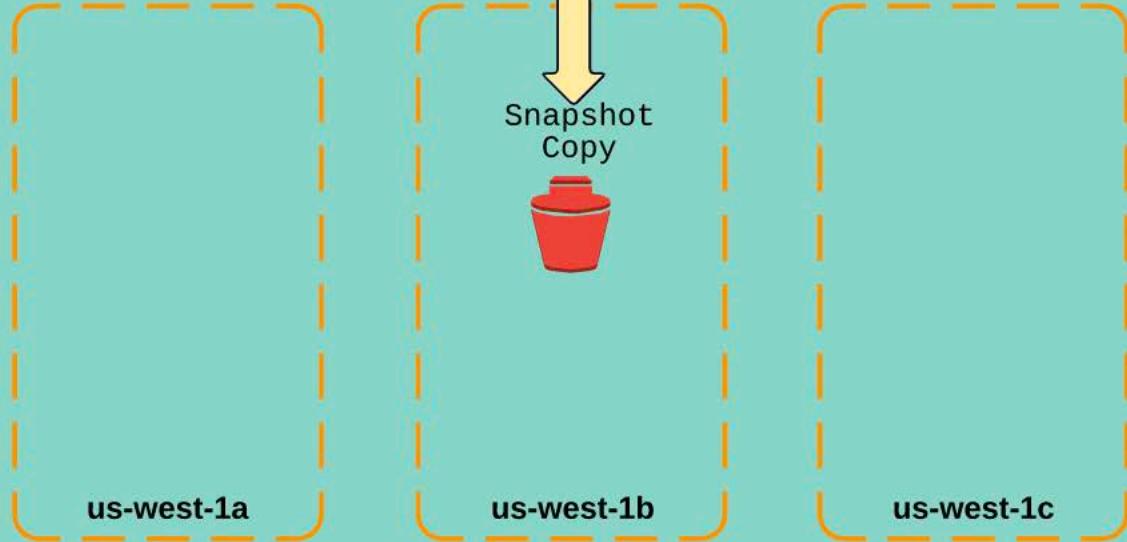
ASG

VPN

us-east-1



us-west-1



EC2 instances run on EC2 hosts located in specific AZs. Failure of an AZ will cause failure of that host and failure of all instances on the host. EBS volumes are stored in a specific AZ. Replication happens in that AZ, but AZ failure can result in data loss. Snapshots are replicated across AZs and optionally copied cross-region.

[Back](#)[Global Infrastructure](#)[Account & Services](#)[Networking](#)[Well-Architected Appendix](#)[IAM](#)[EC2/EBS](#)[S3](#)[R53](#)[ELB/ALB](#)[VPC](#)[NAT GW](#)[ASG](#)[VPN](#)

us-east-1



S3

Replication to ≥ 3 Availability Zones

us-east-1a

us-east-1b

us-east-1c



One Zone

Replication to 1 Availability Zone

us-west-1a

us-west-1b

us-west-1c

S3 is a regional service. For most storage classes, data is replicated across three or more AZs in the region a bucket is located in.



Back

Global Infrastructure

Account & Services

Networking

Well-Architected
Appendix

IAM

EC2/EBS

S3

R53

ELB/ALB

VPC

NAT GW

ASG

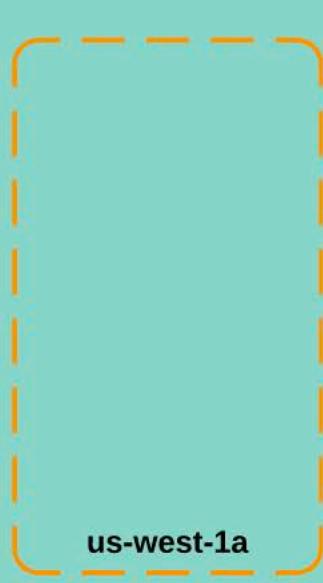
VPN

us-east-1



us-east-1b

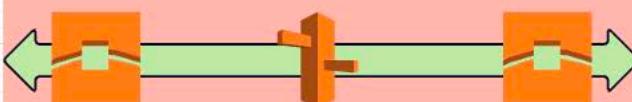
us-east-1c



us-west-1b

us-west-1c

us-west-1



Route 53 operates its name servers from edge locations positioned globally. As such, it's resistant even to issues in AWS regions.

[Back](#)[Global Infrastructure](#)[Account & Services](#)[Networking](#)[Well-Architected Appendix](#)[IAM](#)[EC2/EBS](#)[S3](#)[R53](#)[ELB/ALB](#)[VPC](#)[NAT GW](#)[ASG](#)[VPN](#)

us-east-1

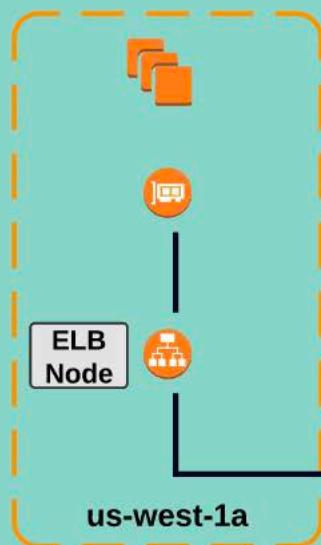


us-east-1a

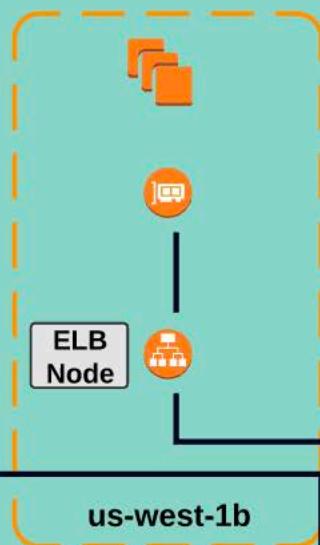
us-east-1b

us-east-1c

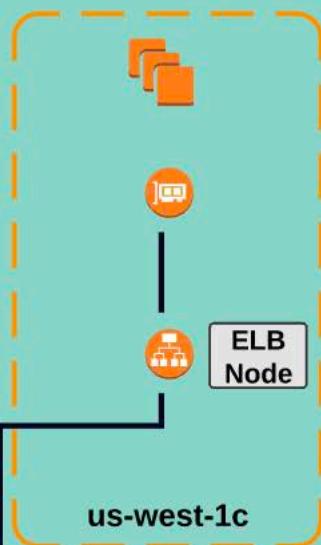
us-west-1

**ELB Node**

us-west-1a

**ELB Node**

us-west-1b

**ELB Node**

us-west-1c

ELBs are regional services and are enabled for specific Availability Zones. When enabled, an ELB node is provisioned for an AZ and network interfaces are placed in a specific subnet. The suggested architecture is to enable the ELB in all AZs where instances are placed.

**ELB**



IAM

EC2/EBS

S3

R53

ELB/ALB

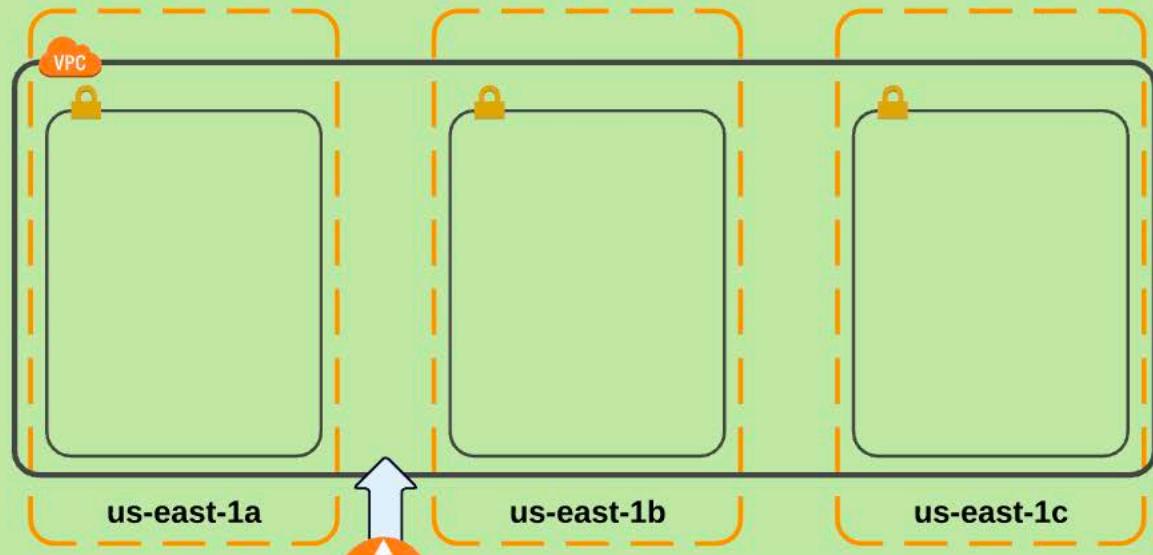
VPC

NAT GW

ASG

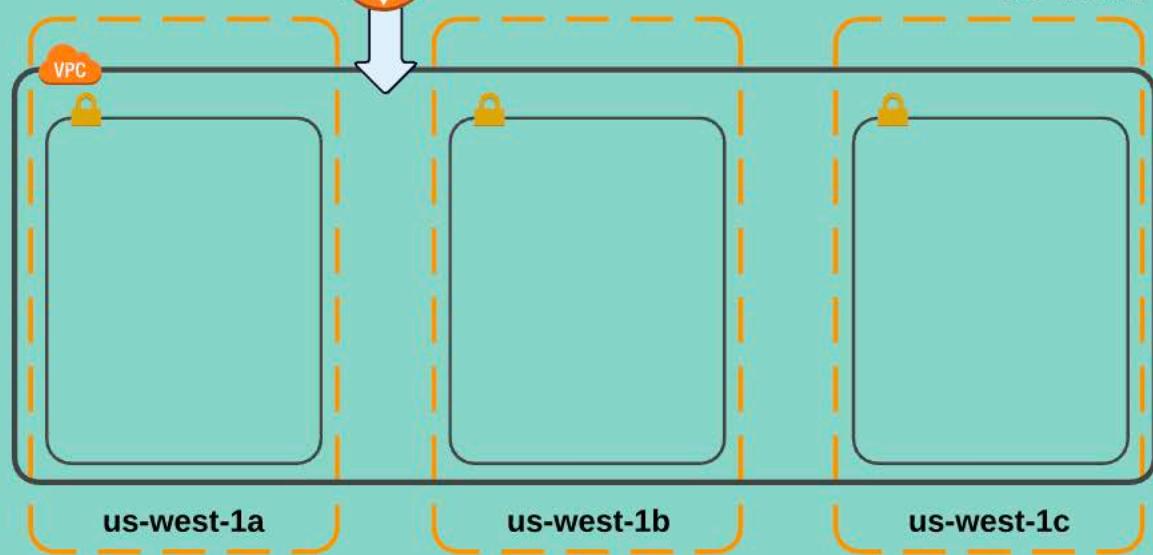
VPN

us-east-1



VPC Peer

us-west-1



VPCs are regional services — they can operate in all Availability Zones in a region. They are tolerant to AZ failure in a region, but certain components such as subnets are linked to a particular AZ. Other components, such as internet gateways and VPC peering, is HA by design.



Back

Global Infrastructure

Account & Services

Networking

Well-Architected
Appendix

IAM

EC2/EBS

S3

R53

ELB/ALB

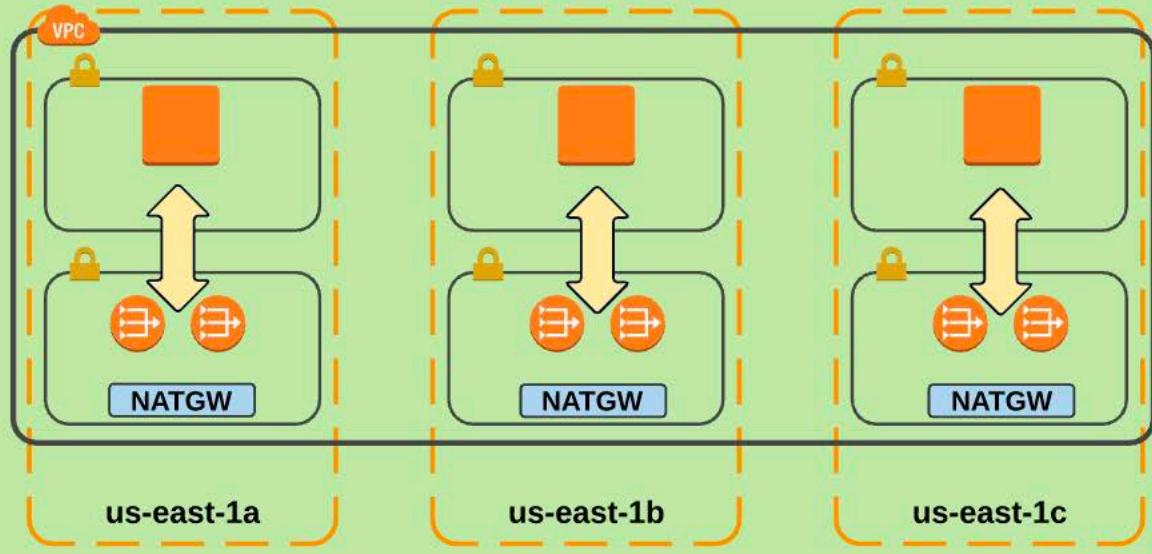
VPC

NAT GW

ASG

VPN

us-east-1



us-east-1a

us-east-1b

us-east-1c

us-west-1

us-west-1a

us-west-1b

us-west-1c

NAT Gateways operate in a specific subnet that is linked to an Availability Zone. NAT Gateways are implemented with redundancy inside each subnet — but to implement zone-independent redundancy, create a NAT Gateway in each Availability Zone.



IAM

EC2/EBS

S3

R53

ELB/ALB

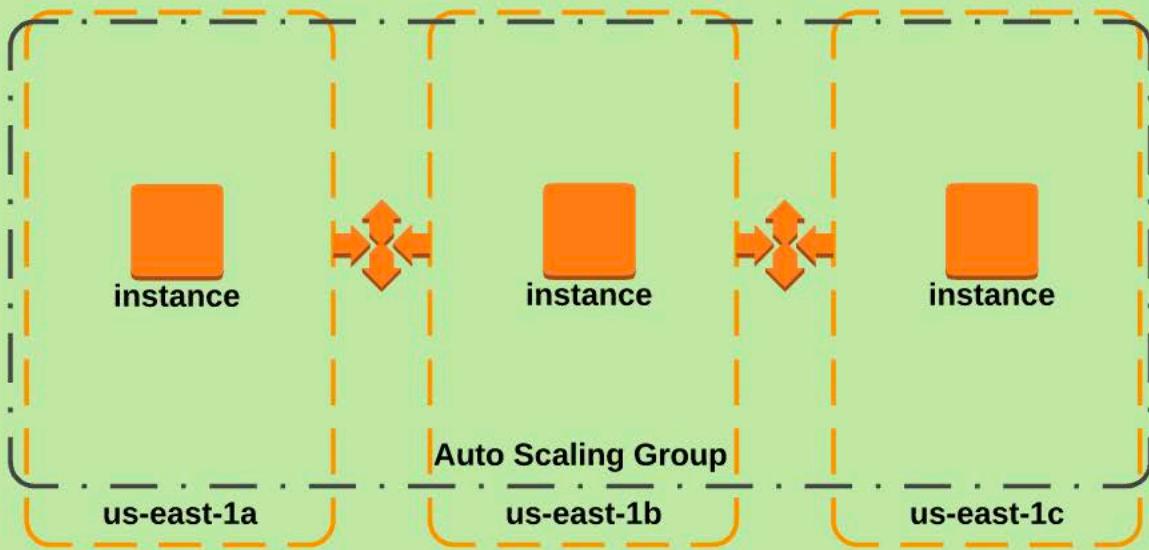
VPC

NAT GW

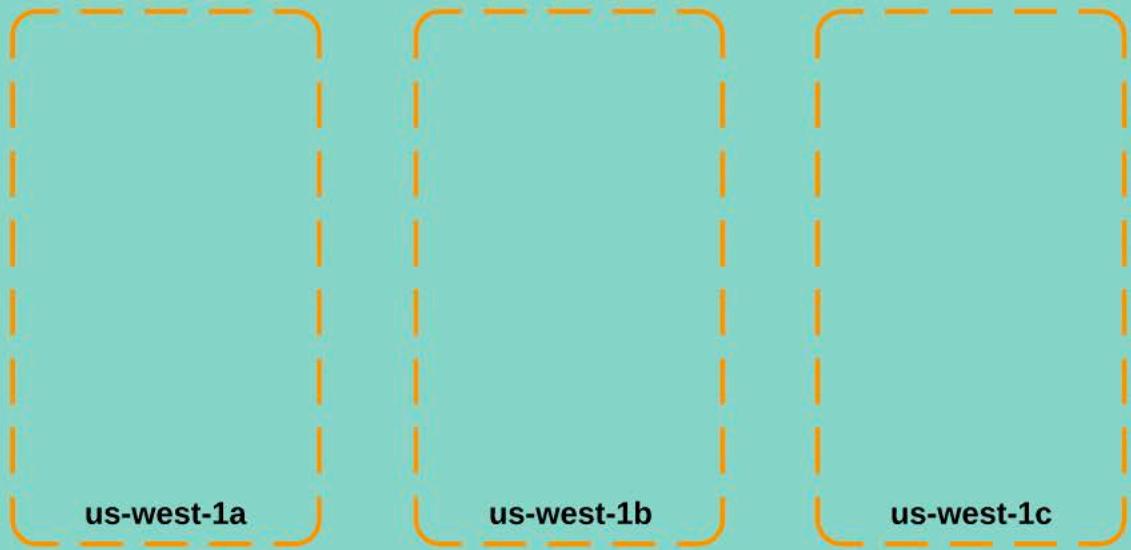
ASG

VPN

us-east-1



us-west-1



EC2 Auto Scaling groups allow EC2 instances to be provisioned and terminated based on load demands. Because min, max, and desired values are specified, ASGs can manage a unit of compute capable of reacting to AZ failure. ASGs operate in a VPC across subnets in that VPC.



Back

Global Infrastructure

Account & Services

Networking

Well-Architected
Appendix

IAM

EC2/EBS

S3

R53

ELB/ALB

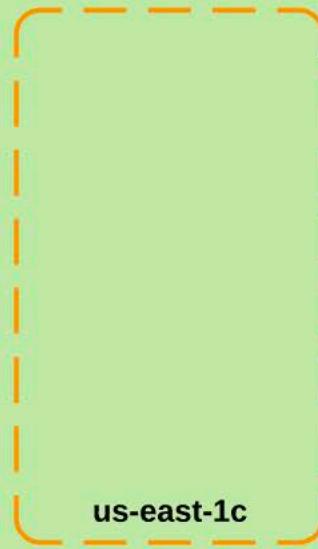
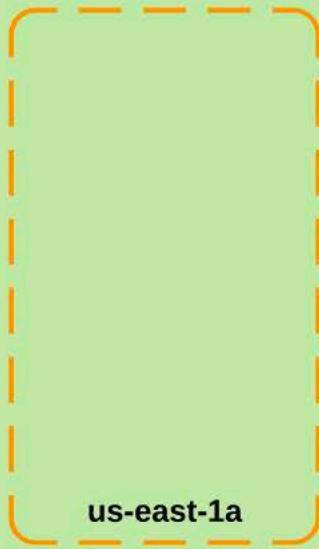
VPC

NAT GW

ASG

VPN

us-east-1

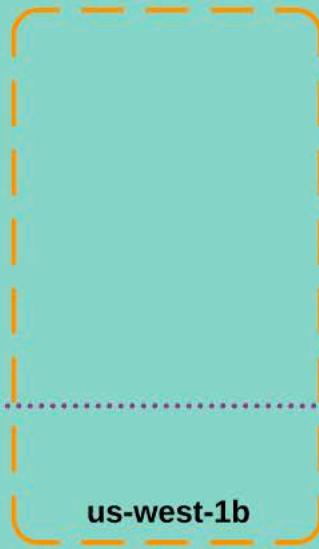
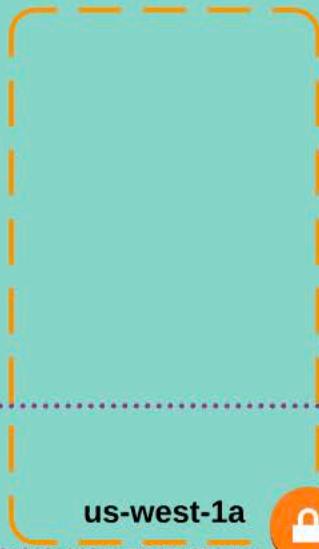


us-east-1a

us-east-1b

us-east-1c

us-west-1



VPGW



us-west-1a



us-west-1b



us-west-1c

Virtual private gateways provision two IPsec endpoints. These endpoints operate in different Availability Zones and so can tolerate failures of individual Availability Zones.



Elastic Load Balancing (ELB)

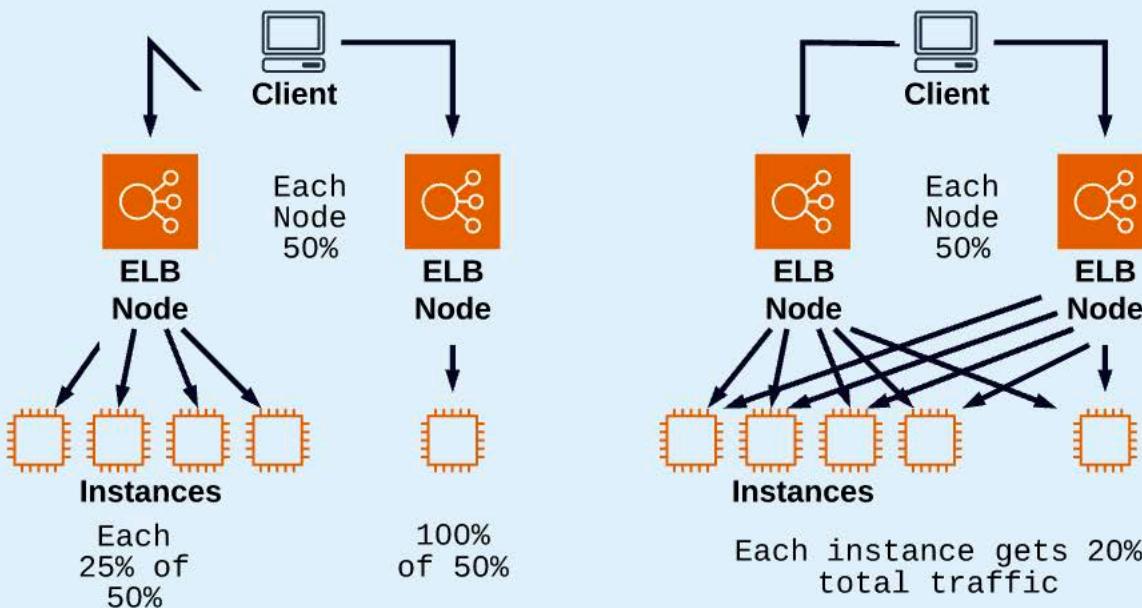
ELB Essentials

Classic LB

Application LB

Network LB

- **Load balancing** is a common method used for distributing incoming traffic among servers.
- Incoming connections are made to the load balancer, which distributes them in some way to associated resources.
- **Elastic Load Balancing (ELB)** is a service that provides a highly available and scalable load balancer as a service. It's available in three forms: Classic (CLB), Application (ALB), and Network (NLB).
- Elastic Load Balancing can be paired with Auto Scaling to enhance high availability and fault tolerance, as well as allow for automated scalability and elasticity.
- An ELB has its own DNS record set (A record) that allows for direct access from its external side.
- Cross-zone load balancing: An ELB can load balance traffic to instances located across multiple Availability Zones. This allows for a highly available and fault-tolerant architecture.



- An ELB can be **public facing**, meaning it accepts traffic from the public internet, or an **internal** load balancer, which is only accessible from within a VPC and is often used to decouple tiers of an application.
- ELBs will automatically stop serving traffic to an instance that becomes unhealthy (via **health checks**).
- A CLB or ALB can help reduce compute power on an EC2 instance by allowing for an SSL certificate to be applied directly to the ELB. SSL encryption and decryption is "**offloaded**" by the ELB in a process called **SSL offload**.



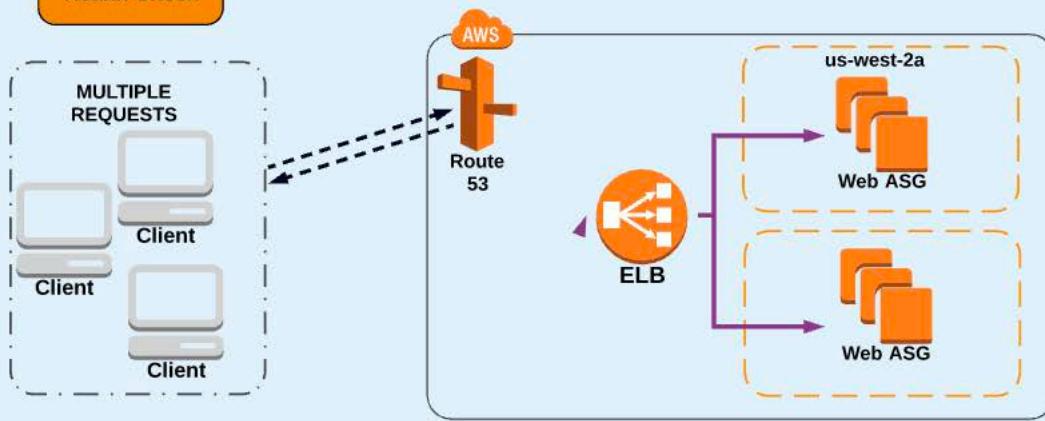
Elastic Load Balancing (ELB)

[ELB Essentials](#)[Classic LB](#)[Application LB](#)[Network LB](#)

X

Classic Elastic Load Balancer

- A "classic" elastic load balancer is designed for balancing traffic (load) to multiple EC2 instances.
- It was originally introduced with ec2-classic and is *not* recommended for use within a VPC.
- It isn't a Layer 7 device, so there are no granular routing "rules" — all instances get routed to evenly, and no special routing requests can be made based on specific paths in the request (e.g., /images/).
- A listener is created that listens to client requests and routes them to the correct destination on the backend.
- There are several protocols supported, including:
 - TCP
 - SSL
 - HTTP
 - HTTPS
- Utilizing SSL/TLS at the load balancer level is considered SSL offloading; all of the cipher suite work is done at the load balancer, which frees up resources on the servers.
- Classic Load Balancers are usually associated with Auto Scaling groups, allowing for a stateless architecture.
- One SSL certificate and cipher configuration *per* listener (effectively per CLB).
- You do *not* assign public IPs to load balancers! Communication is based on an A record (DNS).

[Health Check](#)

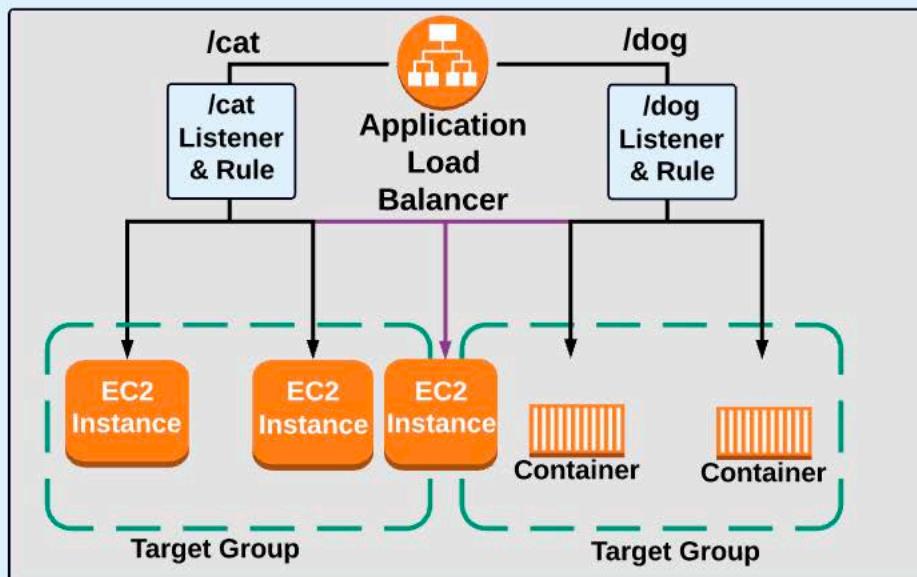


Elastic Load Balancing (ELB)

[ELB Essentials](#)[Classic LB](#)[Application LB](#)[Network LB](#)

Application Load Balancer

- Application Load Balancers operate and understand up to and including Layer 7 of the OSI 7-Layer networking model.
- ALBs are the recommended load balancer type to use within VCPs rather than Classic Load Balancers.
- They perform better than Classic Load Balancers, are almost always cheaper, and can take routing decisions based on Application level (Layer 7) specifics.
- Content rules are used to direct traffic toward target groups.
- Content-based rules (set up on the listener) can be configured using:
 - **Host-based rules:** Route traffic based on the *host field* of the HTTP header.
 - **Path-based rules:** Route traffic based on the *URL path* of the HTTP header.
 - This allows you to structure your application as smaller services and even monitor/Auto Scale based on traffic to specific "**target groups**."
 - You can balance traffic to multiple ports.
- An ALB also supports ECS and EKS, HTTPS, HTTP/2, WebSockets, access logs, sticky sessions, and AWS WAF (web application firewall).
- ALBs allow routing requests to multiple applications on one EC2 such as containers.
- ALBs can be used with other AWS services such as ECS for microservices.
- Targets can belong to multiple target groups.
- ALBs can send traffic to Lambda functions.
- Multiple SSL certificates can be managed by a single ALB using SNI.





Elastic Load Balancing (ELB)

[ELB Essentials](#)[Classic LB](#)[Application LB](#)[Network LB](#)

Network Load Balancer

Network Load Balancers operate at Layer 4 of the OSI 7-Layer model. The Network Load Balancer is designed for *extreme performance*. It's capable of handling millions of requests per second and can scale to handle volatile and unpredictable traffic levels.

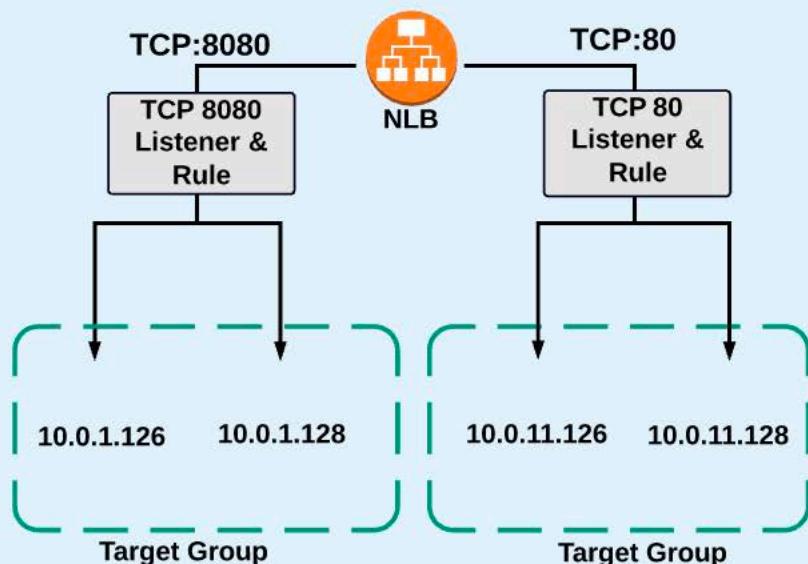
Network Load Balancers support static IPs and ultra low latency.

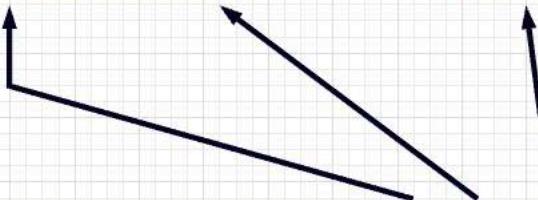
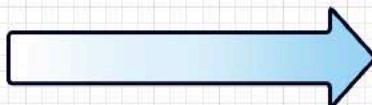
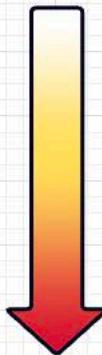
Network Load Balancers share the same core configuration entities with ALBs:

- Listeners
- Target groups
- Targets

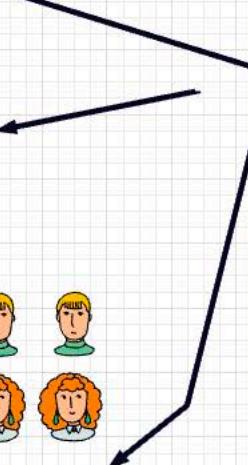
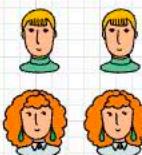
NLBs are assigned a single IP address per Availability Zone and can be assigned an EIP, including ones you own for the lifetime of the NLB. This allows for *much easier* firewall management because this static IP can be whitelisted.

Network Load Balancers **preserve the client IP**. It's a passthrough device — they don't alter the source IP packets.



EC2
Instance

Horizontal scaling means increasing or decreasing system load by increasing or decreasing the number of instances serving the demand. Elasticity is an evolution of this process where this scaling is fully automatic based on system load.



Vertical scaling means increasing load demands by increasing the size of compute resources. Extra CPU, memory, and disk capacity allow systems to handle additional load. Vertical scaling is generally used when applications are monolithic and the state is stored with the application on the server.

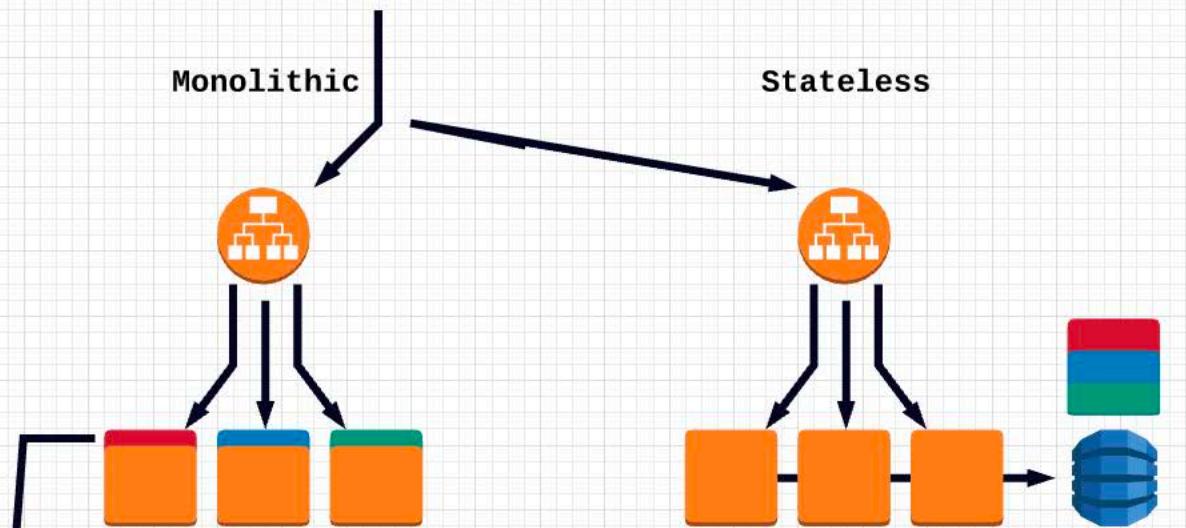


The screenshot shows the Linux Academy website interface. At the top, there's a navigation bar with links for Home, Training, Playground, Quick Training, Hands-On Labs, Learning Paths, and Community. Below the navigation is a search bar. The main content area displays a list of courses under the heading "Learn the AWS Cloud (80+ AWS services, 400+ training modules and 100+ labs)". Some visible course titles include "Amazon Lightsail Deep Dive", "Creating and Interacting with a Lightsail Instance", "Installing and Configuring MongoDB in Lightsail", "Implementing a MEAN App using Lightsail", "Scaling a MEAN App in Lightsail Using Auto Scaling", and "Implementing and Scaling a LAMP Application". On the right side of the page, there's a sidebar with sections for "Welcome, Adrian", "Bookmarks", and "Community Posts".

With a monolithic and unscalable architecture, the "state" is stored on the application server. Examples of the state include a shopping cart or the login state of a web application.

Using a stateless architecture, the state is held separately from the application servers.

Monolithic **Stateless**



Stateless architectures allow load to be evenly distributed across any available instances and allow traffic to be redistributed when failure occurs without disrupting the user experience.

Session data is stored on the application servers. Each session store is unique to the server. If the server fails, the load balancer will redirect clients to alternative servers. Because the session data will be lost, the user experience will be disrupted.

With stateless servers and applications, the session state is moved to a separate store such as DynamoDB. With this style of architecture, when an individual instance fails and traffic is migrated to another instance, session data is still available.



Performance



Geo Restriction

Viewer Protocol

Signed URL

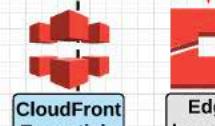
Deploy configuration to edge locations



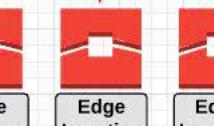
Web Distribution



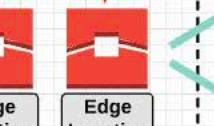
RTMP Distribution

CloudFront
Essentials

Edge Location



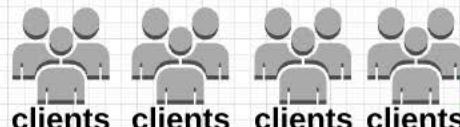
Edge Location



Edge Location

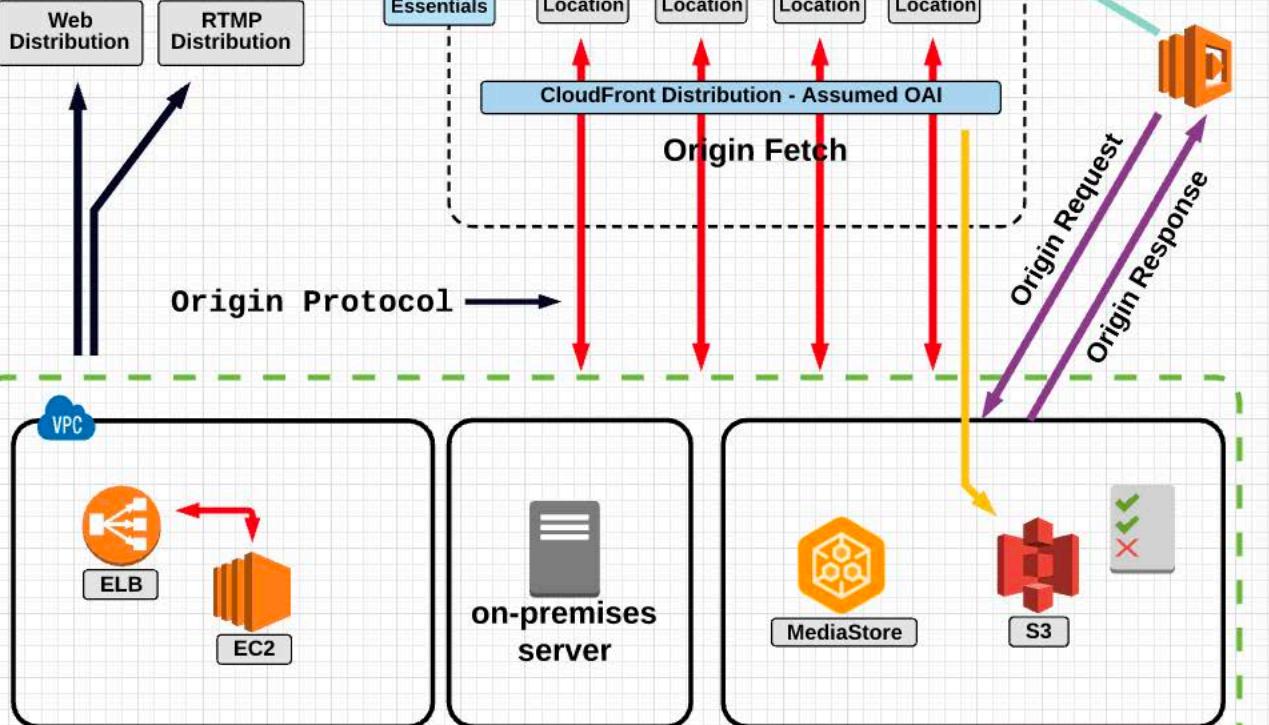
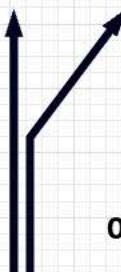


Edge Location

Route 53
(DNS)Viewer Request
Viewer Response

Origin Protocol

Origin Fetch

Origin Request
Origin Response

CloudFront "Origin"



CloudFront Performance Considerations

- Regional caches
 - Save content ejected from cache close to edge location
- Point entire domain to CloudFront to speed up dynamic content
 - Connection optimizations
 - Dynamic objects are still cached
- CloudFront performance can be affected by:
 - File size and type of file
 - Having to remake the request from the edge location to the origin
 - Downloading the object from the origin takes time.
 - As well as writing it to cache and responding to the end-user request.
 - The more requests that have to go to the origin, the higher the load is on your source — which can also cause latency and load performance issues.
- Query strings (request to the origin to serve a specific object) reduce cache “hits”:
 - <http://bestcatpicsintheworldever/?query=redcats>
 - It reduces performance because query strings are often unique, so it reduces the cache hits and also requires extra “work” in order to forward to the origin location.
- CloudFront performance can be increased by:
 - Longer cache periods increase performance (less frequent requests to the source).



CloudFront Essentials

- CloudFront is a global CDN that delivers content to users from the nearest edge location.
- CloudFront retrieves content from origins.
- CloudFront distribution (web or RTMP)
 - Define origins for content (static and dynamic)
 - CloudFront can integrate with Route 53 for “alternate” CNAMEs.
 - This allows you to create a URL such as <http://bestcatpicsintheworldever.com> that works with your distribution.
 - Cache behaviors
 - PATH pattern (images/*.jpg)
 - Min, Max, Default TTL
 - Query string forwarding and caching
 - HTTP methods (GET, POST, PUT, HEAD, etc.)

Updating Cached Files:

- Caching is done based on the cache key.
- In order to serve a new version of an object, either create a new object with a new name or create an “invalidation” on the CloudFront distribution based on the object name.
- “Invalidations” can be costly.
- Set TTL = 0



Restricting S3 to CloudFront

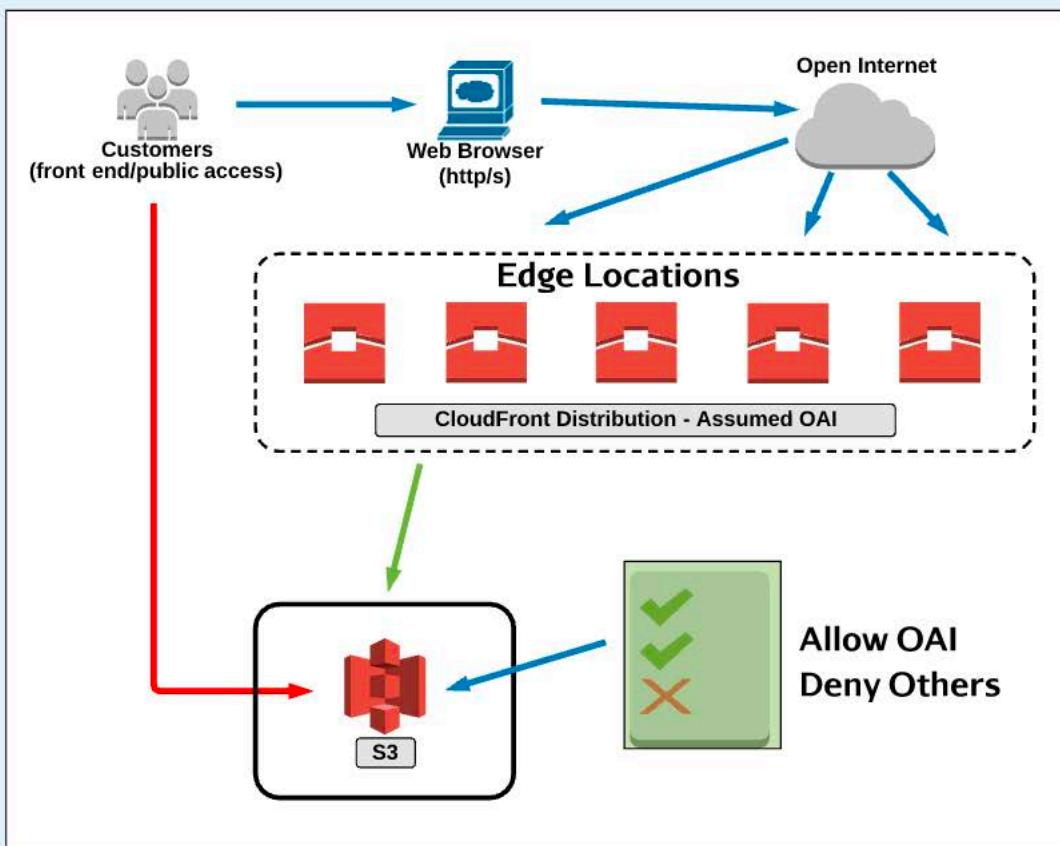
By default, when using CloudFront with S3, CloudFront is optional, and S3 can be accessed directly. This can be changed by creating an origin access identity (**OAII**).

What's an OAI?

An OAI is a "virtual" identity. A distribution can be configured to use it so that, when accessing S3, CloudFront assumes this identity.

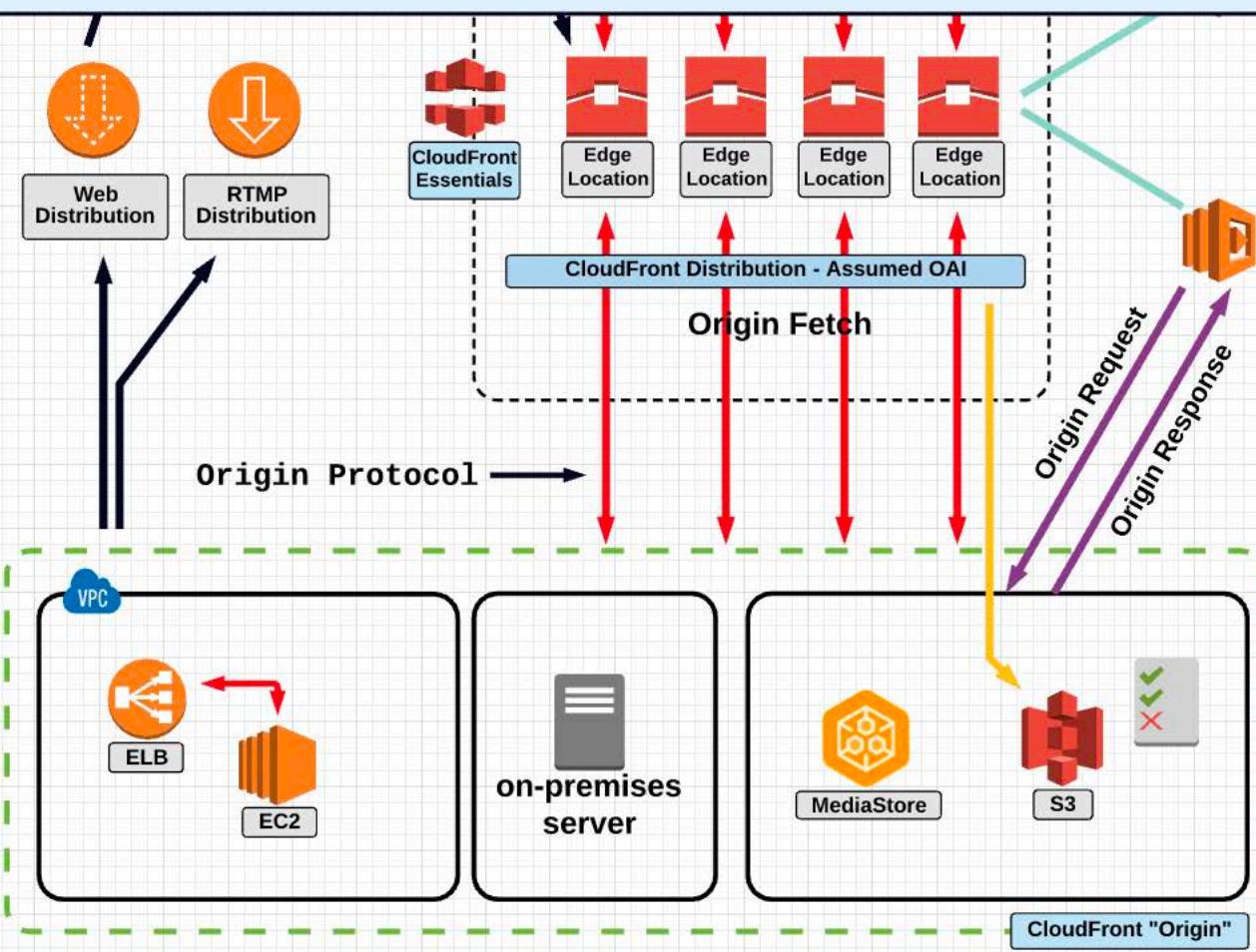
How Is an OIA Used and Why?

To use an OAI, public permissions are removed from your S3 bucket policy and permissions for the OAI are added. Only the CloudFront using that OAI can access your S3 bucket.





```
{  
    "Version": "2008-10-17",  
    "Id": "PolicyForCloudFrontPrivateContent",  
    "Statement": [  
        {  
            "Sid": "1",  
            "Effect": "Allow",  
            "Principal": {  
                "AWS": "arn:aws:iam::cloudfront:user/CloudFront Origin  
Access Identity XXXXXXXXXXXX"  
            },  
            "Action": "s3:GetObject",  
            "Resource": "arn:aws:s3:::mybucket/*"  
        }  
    ]  
}
```





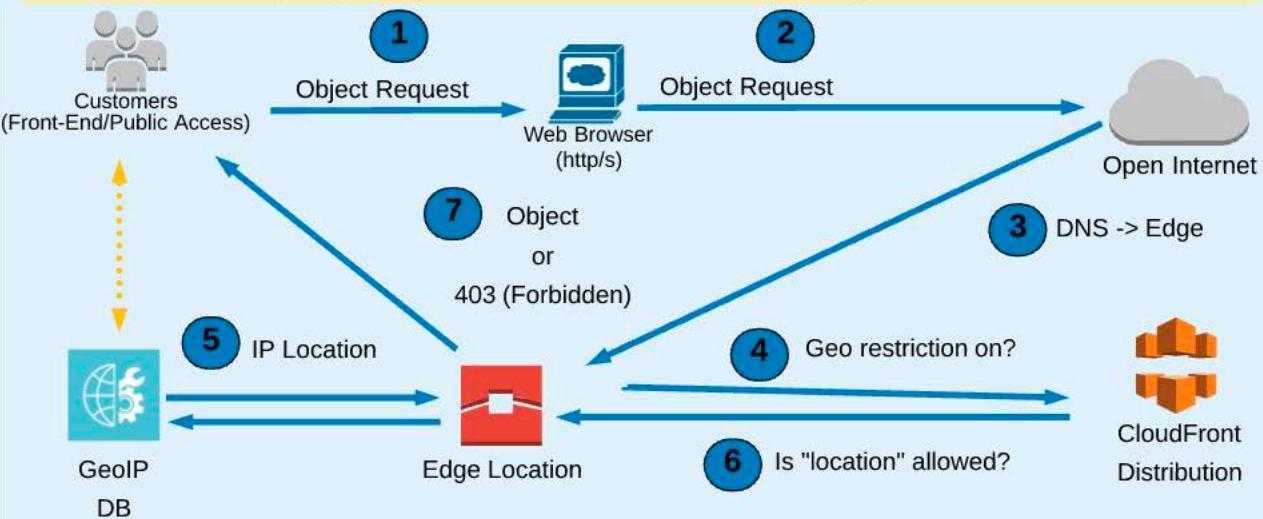
Geo Restriction — CloudFront can restrict content in one of two ways:

CloudFront geo restriction is a simple implementation.

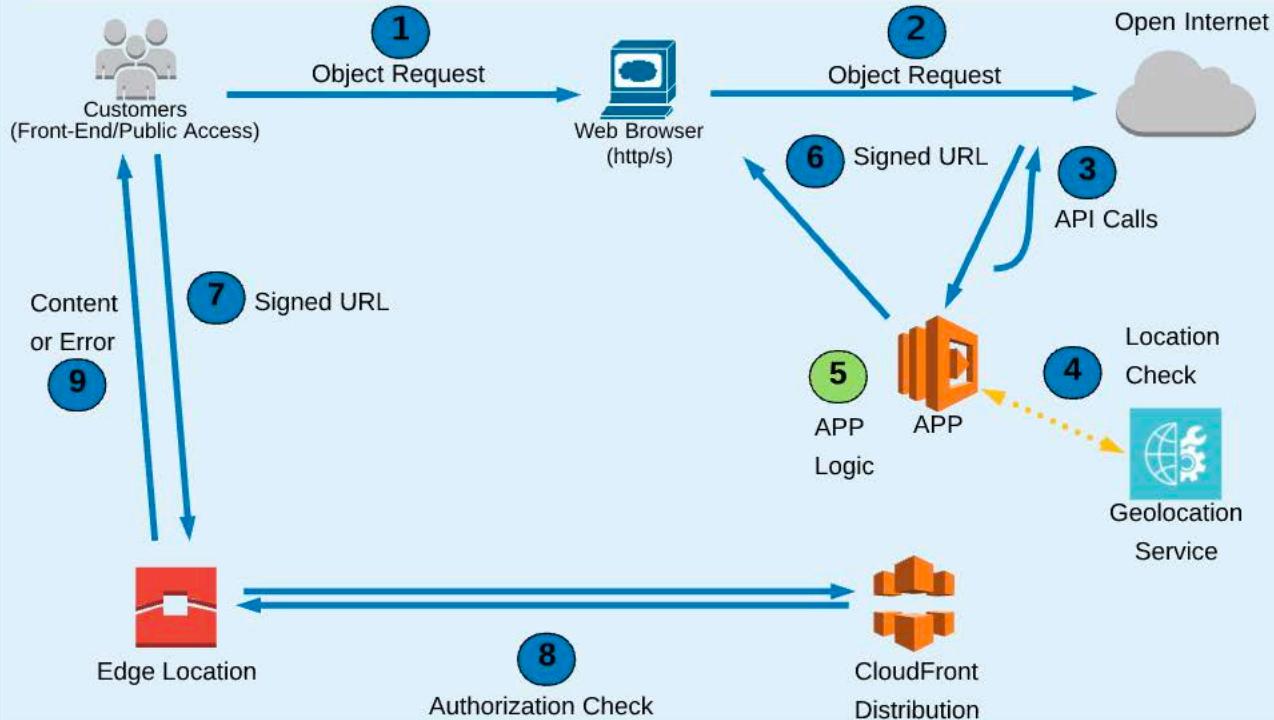
Whitelist or blacklist, and it works on country restrictions *only*.

Location is based on IP country location — backed by a GeoIP database (~99.8% accuracy)

No restrictions on anything else (session/cookie/content/browser, etc.).



Third-party geo restriction needs a server/serverless application — signed URLs are used. A third-party geolocation service is used for extra accuracy. Your application can apply additional restrictions such as session, browser, or account. Location can be *much* more accurate: city, locale, latitude/longitude in some cases.





?

Back

Global Infrastructure

Account & Services

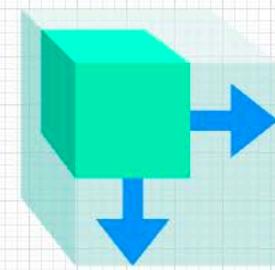
Networking

Well-Architected

Appendix



EC2 Self-Managed Databases



Database Models

SQL Databases



Aurora



RDS



Athena

NoSQL Databases



DynamoDB



Neptune



QLDB



Timestream



DocumentDB



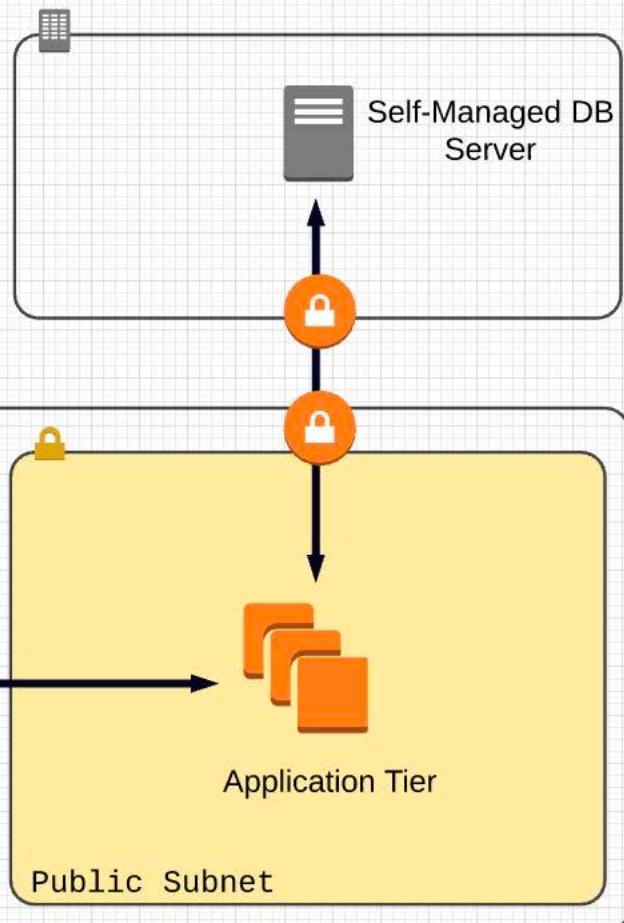
Elasticache



AWS

Self-managed databases can be installed and operated either from business premises or from an EC2 instance running inside an AWS VPC.

This architecture provides OS-level root user access and full control over all OS and DB vendors and components.



Risks and Limitations

Running a database (DB) on EC2 adds admin overhead. The system administrator is responsible for installation, configuration, performance management, and ongoing backup and DR issues.

Additionally, the DB runs on EC2, bringing additional risk of data or service loss if an AZ fails. The EC2 instance and backing EBS volume are both "AZ" services.

Backups can be performed via EBS snapshots to S3. These backups are replicated across multiple AZs in the region but are self-managed and the responsibility of the system's administrator.

If HA is required, the system administrator is responsible for any sync or async replication in addition to management of additional DB nodes.



Relational Database

Relational database management systems (RDBMS) are used when the data to be managed has formal and fixed relationships. Data is stored on disk as rows, and entire rows must be parsed even if individual attributes are all that's needed. Reading 1 attribute from 10,000 records requires 10,000 rows to be read from disk.

Every table has a schema that defines a fixed layout for each row, defined when the table is created. Every row in the table **needs** to have all the attributes and the correct data types.

RDBMS conforms to the ACID system: **A**tomicity, **C**onsistency, **I**solation, and **D**urability. This does impact the ability to achieve high performance levels and limits scalability, but for most applications, the tradeoff is worth it.

Data stored in RDBMS is stored in a normalized form, which means removing repeating data and storing only single values in attributes. To query data from related tables (see below), table joins are used, where attributes are matched based on primary and foreign keys.

SQL (or Structured Query Language) is used to interact with most SQL (relational) database products.

ID	Name	Color
0001	Roffle	B/W
0002	Penny	All the colors
0003	Winkie	White
0004	Truffles	Mixed

ID	Human Slave
0001	Adrian
0001	Nat
0002	Adrian
0002	Nat
0003	Adrian
0003	Nat
0004	Adrian
0004	Nat

Fixed relationships exist between tables based on table keys. Queries join tables to use information from both.



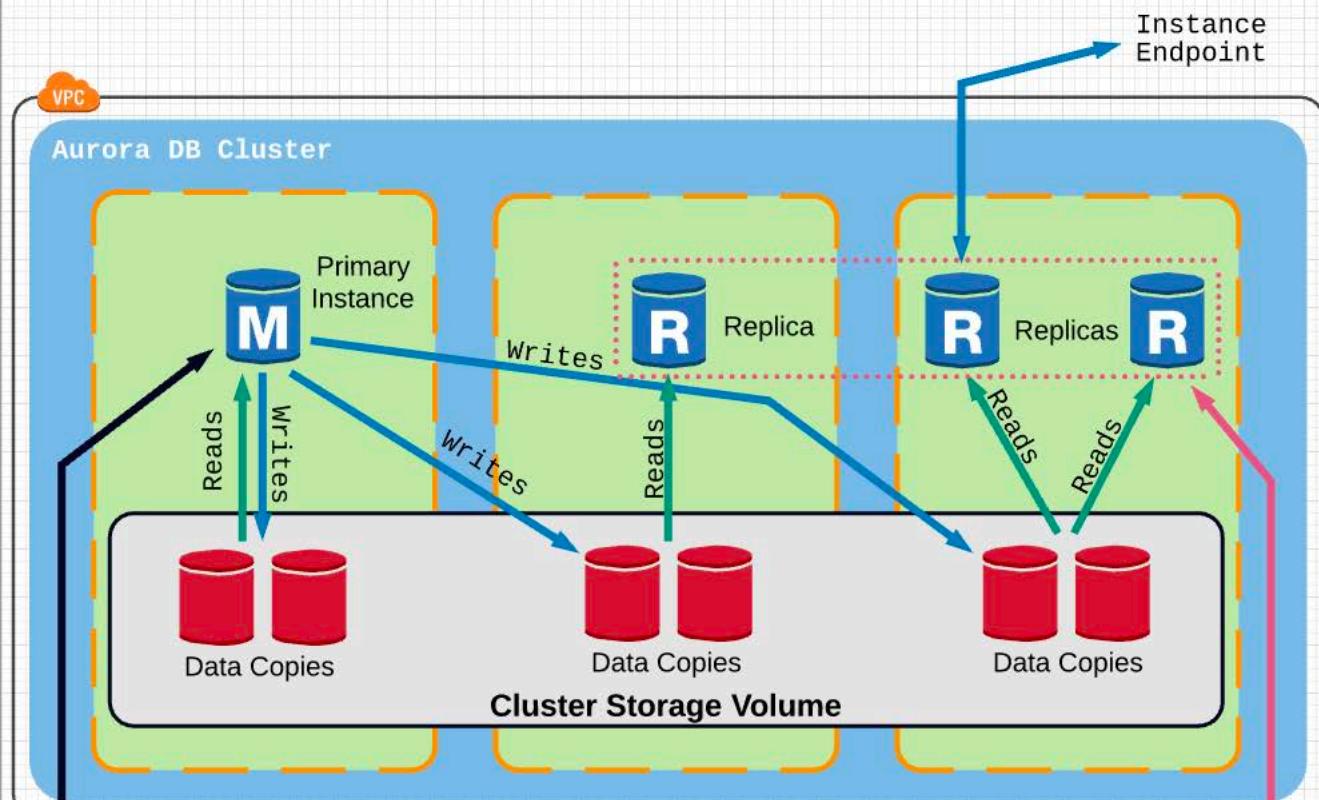
Aurora

Aurora Serverless

Aurora Global

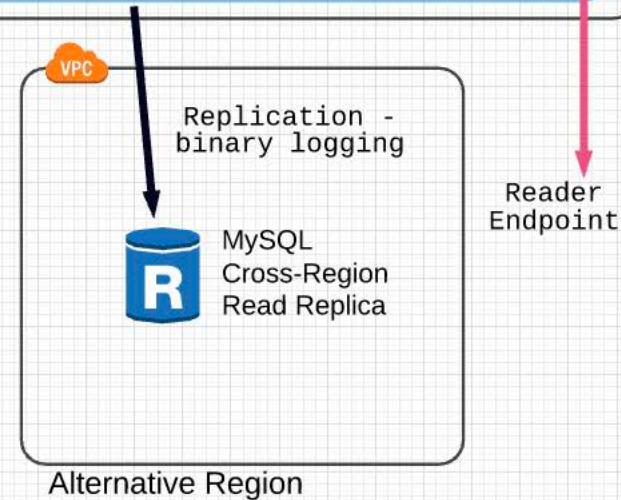
Amazon Aurora

A fully managed relational database product that provides **MySQL** and **PostgreSQL** compatibility. Aurora has been designed to deliver up to 5x the performance of MySQL and 3x the performance of PostgreSQL while maintaining compatibility.



The maximum size for Aurora storage is 64 TiB, but billing is based on storage amount used, not provisioned. Storage grows automatically as data is added.

Cluster Endpoint





Aurora

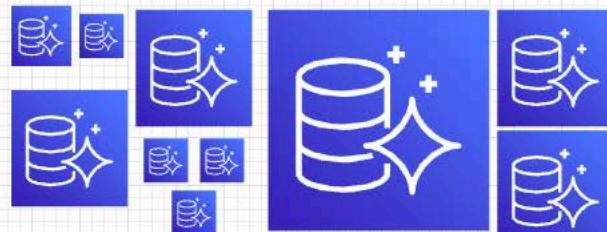
Aurora Serverless

Aurora Global

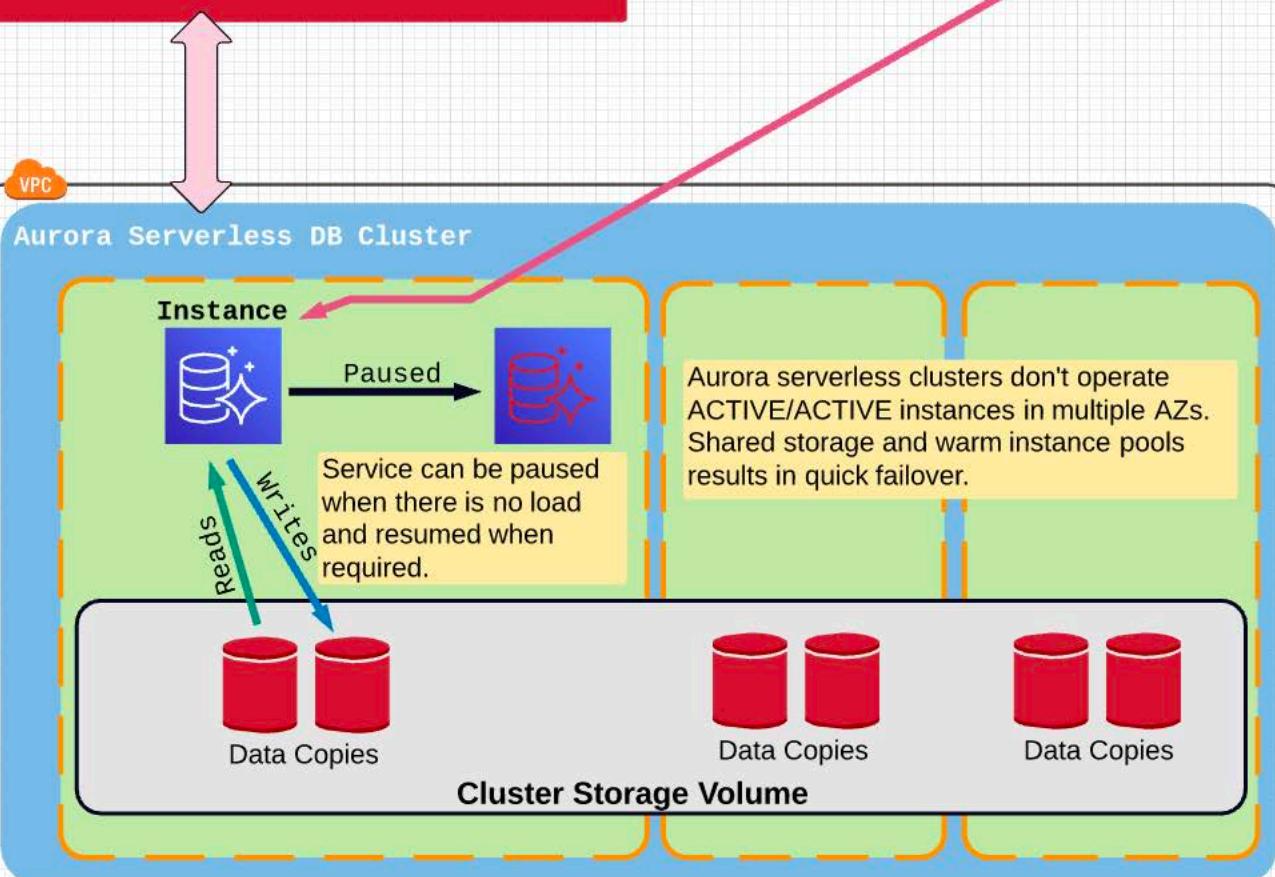
Aurora maintains a warm pool of instances (without any data) ready to swap into an existing serverless cluster as capacity is needed.

Connections are managed by a shared proxy layer that brokers connections and stops interruptions due to scaling events or failures.

Connections to the databases can be achieved either by using traditional MySQL SDKs/CLIs or via the new AWS **DATA API**.



Warm Instance Pool



Aurora serverless clusters don't operate ACTIVE/ACTIVE instances in multiple AZs. Shared storage and warm instance pools results in quick failover.

An Aurora Capacity Unit (ACU) is 1vCPU and 2 GB RAM. For a given cluster, a min/max size is defined. Aurora scales **up** and **down** based on CPU utilization and connections.

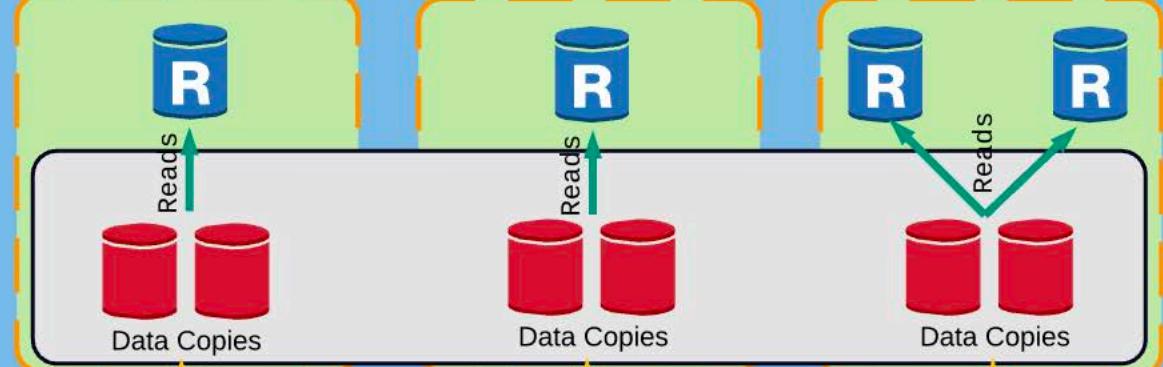




Aurora

Aurora Serverless

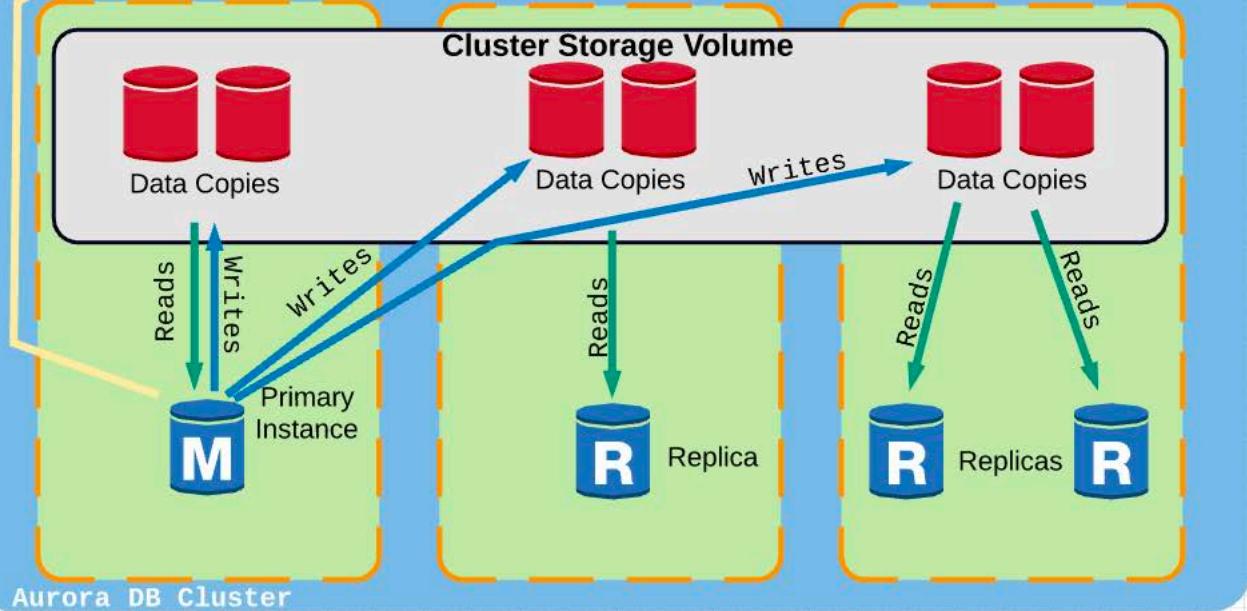
Aurora Global

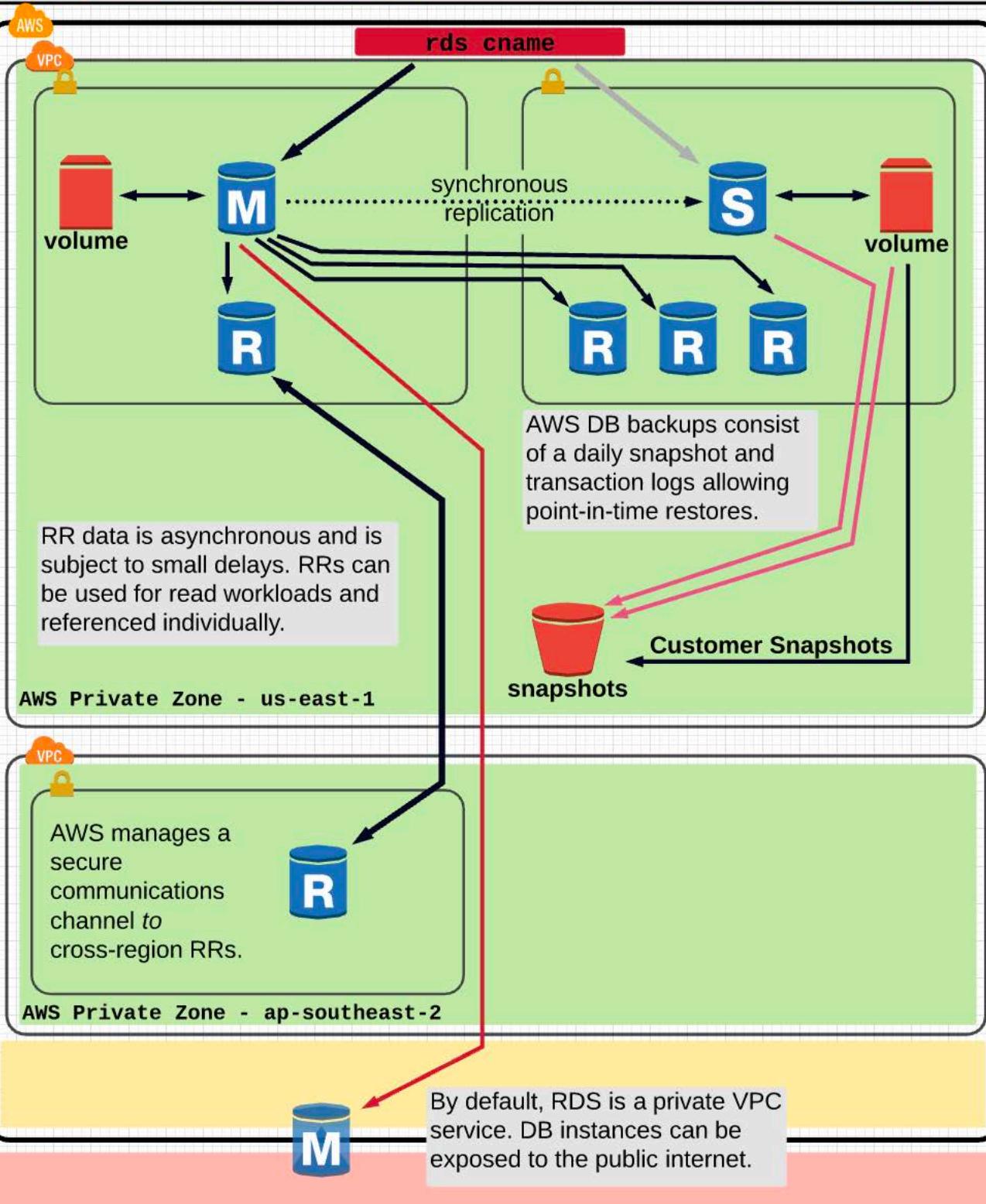


Replication Agent(s) - Remote Region

Logs are streamed between the replication server and agent.

Replication Server(s) - Local Region







XML

JSON

CSV

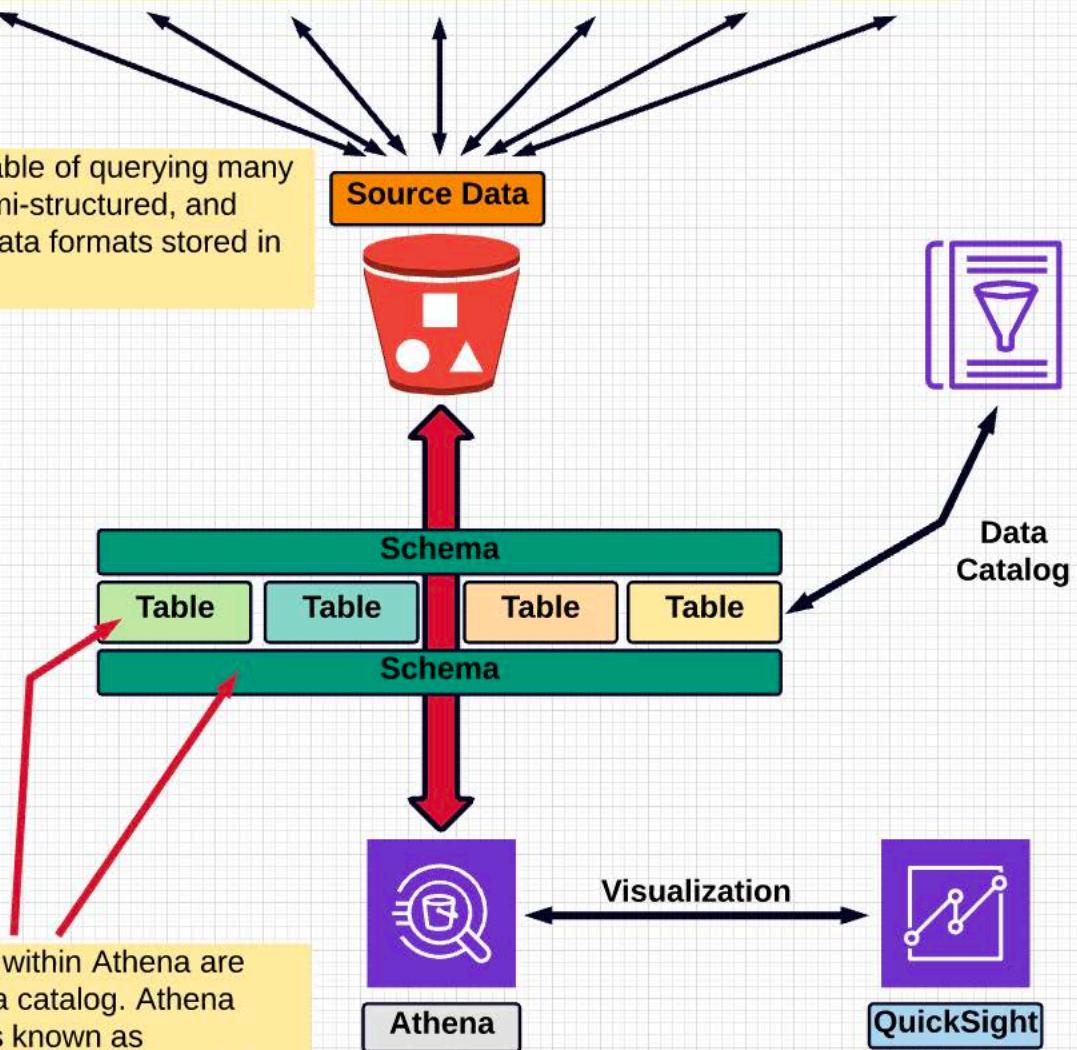
TSV

AVRO

ORC

PARQUET

Athena is capable of querying many structured, semi-structured, and unstructured data formats stored in S3.



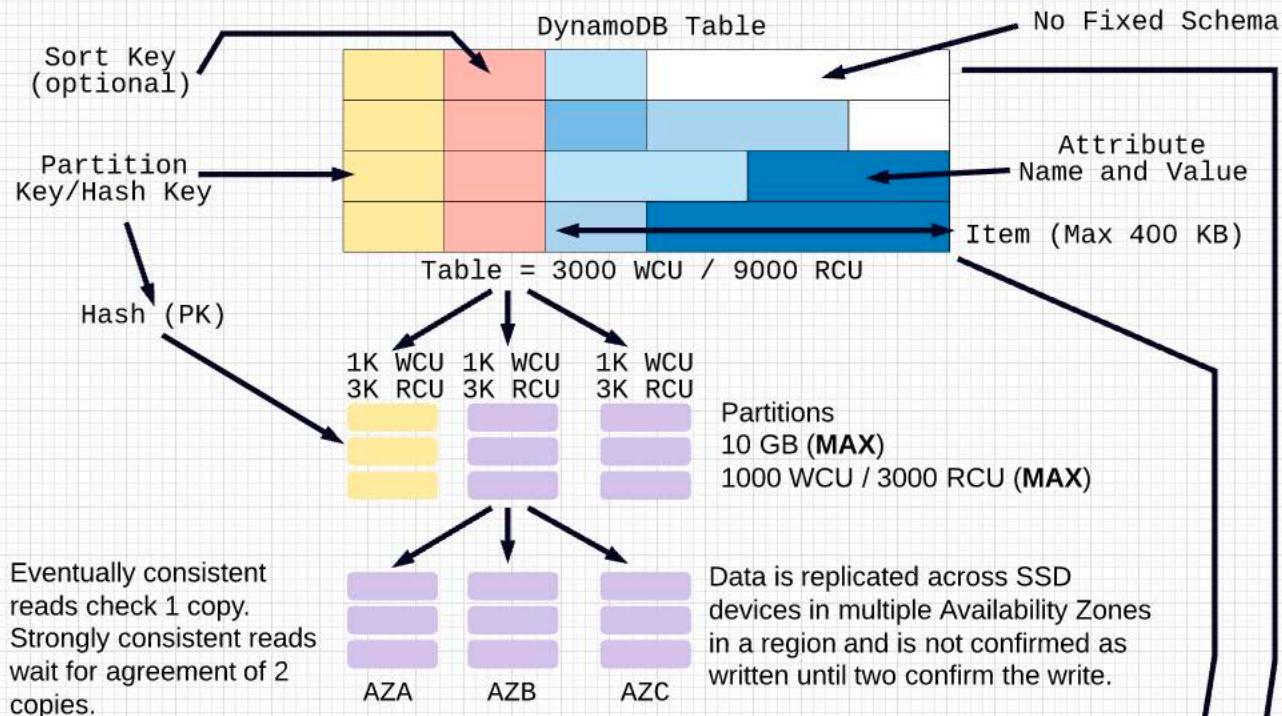
Tables defined within Athena are stored in a data catalog. Athena uses a process known as *schema-on-read*, which means the schema is overlaid on the data and used every time objects in S3 are queried.

Because of this architecture, there is no need for data loading or transformation either in advance or at query time.

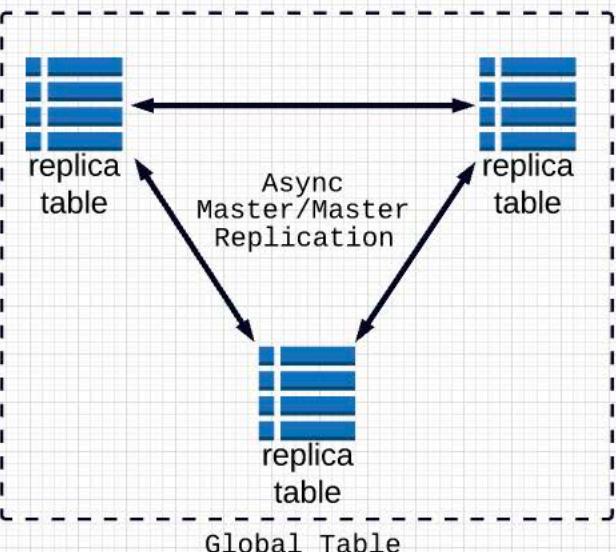
No data is modified within S3 during the query process.

Athena is a serverless query service allowing data stored in S3 to be queried using standard SQL. There is no base cost for using Athena — costs are based on the queries executed.

Athena can be used to query most AWS logging and data output types in addition to most common standard data formats.



Tables can be queried or scanned. Scans are inefficient. Querying can only be done based on partition keys (PK) and any optional sort keys (SK). Local secondary indexes (LSIs) can be added to allow queries against the same PK but alternative SKs. Global secondary indexes (GSIs) allow querying against alternative PKs and SKs.



Streams can be configured to store changes to the DynamoDB table. Streams can either list the **keys** being changed, new image, old image, or new and old images.

Lambda functions can be configured as triggers for when there is stream activity.

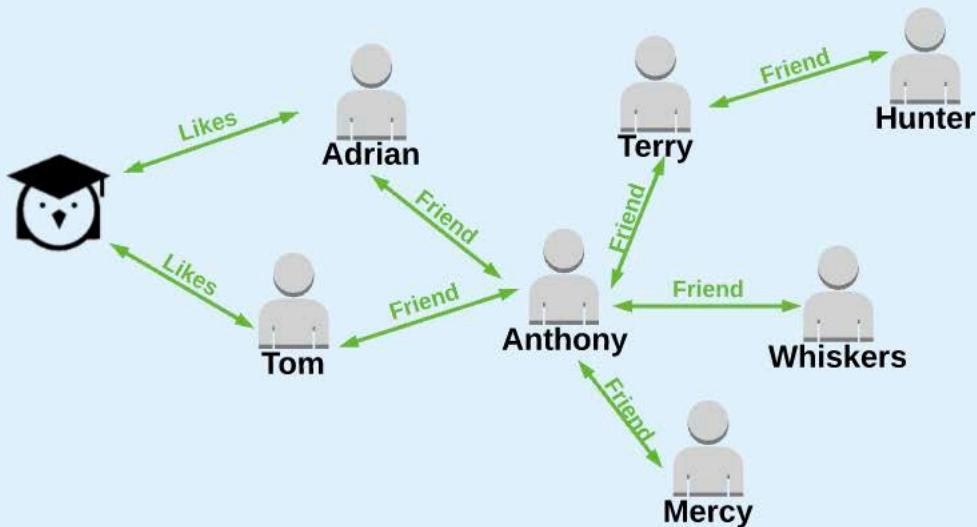




Neptune Essentials

X

- Neptune is a fully managed, NoSQL graph database service provided by AWS.
- Open-source APIs:
 - TinkerPop Gremlin — use Gremlin Console client
 - RDF SPARQL — use RDF4J Console
- Highly available
- Store up to 64 TB and up to 15 Read Replicas
- Server-side encryption
- Graph database:



- Popular use cases include:
 - Social network
 - Knowledge graphs
 - Fraud detection
 - Recommendation engine



Amazon Quantum Ledger Database is a fully immutable database using cryptographically verifiable ledger technology to track every change to data in a way that cannot be manipulated. QLDB uses a document storage model, allowing data with complex, nested structures to be stored and queried.

QLDB is an ACID-based database with strong transactional consistency. This means either an entire transaction is completed or it's fully rolled back. \$47 moved from Account A to Account B must complete with the full amount moved or not at all.

QLDB is fully serverless. It is a DBaaS product and requires no IO provisioning.

```
{  
  name : "truffles",  
  age : "6",  
  status : "cat",  
  slaves : ["Adrian, "Nat"]  
}
```

SQL is supported allowing familiar queries
SELECT *
FROM things
where things.status = 'cat'

Changes in QLDB are cryptographically verifiable using a "hash-chained" journal. When data is initially added, a "hash" is created of that data. When a new version is added, a new hash is created of current data plus the old hash — this forms a chain. It makes tampering or changes immediately detectable. And the more data added, the harder it would be to "fake" this chain.



Because QLDB's data history is verifiable, it makes it ideally suited to applications where the validity of data is a priority (e.g., manufacturing, finance, healthcare, payroll, supply chain, insurance, and voting).



Amazon Quantum Ledger Database is a fully immutable database using cryptographically verifiable ledger technology to track every change to data in a way that cannot be manipulated. QLDB uses a document storage model, allowing data with complex, nested structures to be stored and queried.

QLDB is an ACID-based database with strong transactional consistency. This means either an entire transaction is completed or it's fully rolled back. \$47 moved from Account A to Account B must complete with the full amount moved or not at all.

QLDB is fully serverless. It is a DBaaS product and requires no IO provisioning.

```
{  
  name : "truffles",  
  age : "6",  
  status : "cat",  
  slaves : ["Adrian, "Nat"]  
}
```

SQL is supported allowing familiar queries
SELECT *
FROM things
where things.status = 'cat'

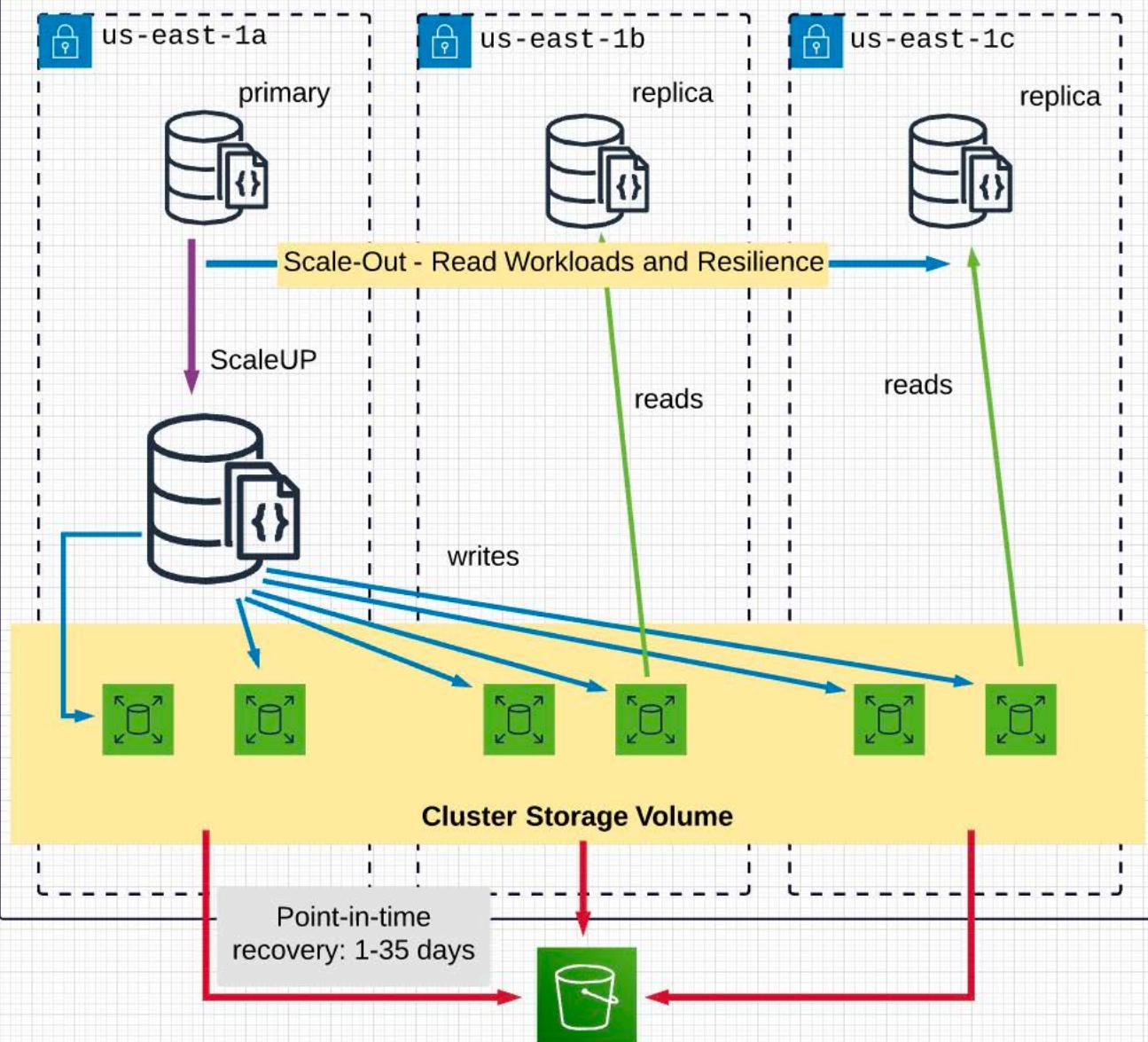
Changes in QLDB are cryptographically verifiable using a "hash-chained" journal. When data is initially added, a "hash" is created of that data. When a new version is added, a new hash is created of current data plus the old hash — this forms a chain. It makes tampering or changes immediately detectable. And the more data added, the harder it would be to "fake" this chain.



Because QLDB's data history is verifiable, it makes it ideally suited to applications where the validity of data is a priority (e.g., manufacturing, finance, healthcare, payroll, supply chain, insurance, and voting).



us-east-1



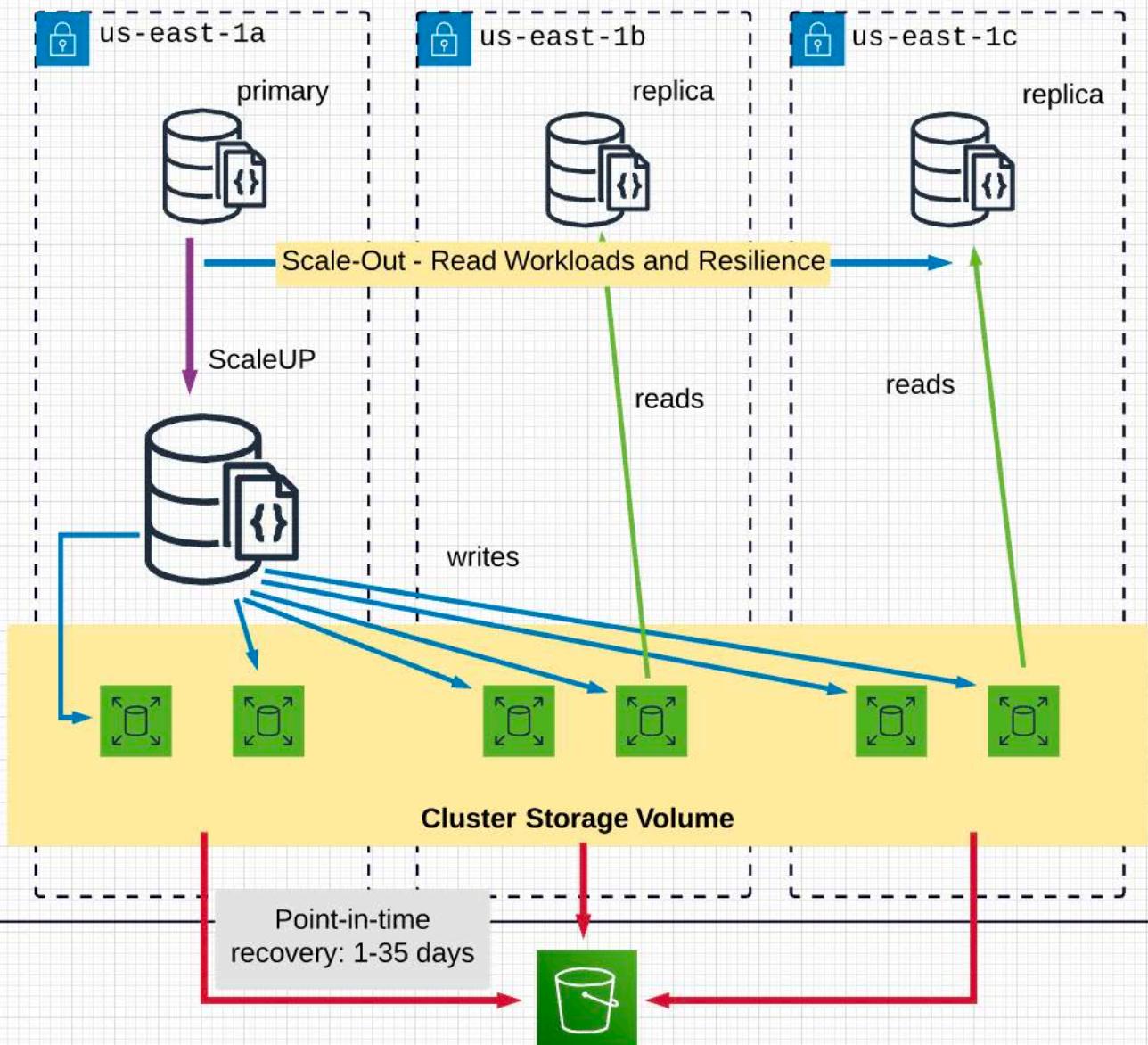
DocumentDB is a managed document storage database compatible with MongoDB. It supports continual backup and point-in-time recovery. It can scale horizontally for durability and read workloads, as well as vertically for write-heavy workloads.

SQL vs DocumentDB

Table <-> Collection
Row <-> Document
Column <-> Field
Primary Key <-> ObjectId



us-east-1



DocumentDB is a managed document storage database compatible with MongoDB. It supports continual backup and point-in-time recovery. It can scale horizontally for durability and read workloads, as well as vertically for write-heavy workloads.

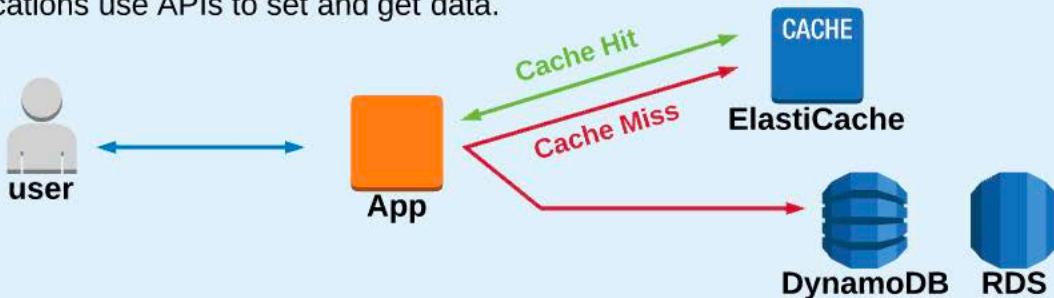
SQL vs DocumentDB

Table <-> Collection
Row <-> Document
Column <-> Field
Primary Key <-> ObjectId



ElastiCache

- ElastiCache is a fully managed, in-memory data store and cache service.
- ElastiCache is used to improve database performance by caching results of queries that are made to a database — thus, improving speed and reducing database load.
- ElastiCache provisions node clusters that you can scale.
- ElastiCache can host web sessions.
- Applications use APIs to set and get data.



	Memcached	Redis (cluster mode disabled)	Redis (cluster mode enabled)
Engine versions	1.4.x	2.8.x and later	3.2.x and later
Data types	Simple ‡	2.8.x - Complex *	3.2.x and later - Complex †
		Complex †	
Data partitioning	Yes	No	Yes
Cluster is modifiable	Yes	Yes	3.2.10 and later - Limited
Online resharding	No	No	3.2.10 and later
Encryption	No	3.2.6, 4.0.10 and later	3.2.6, 4.0.10 and later
Compliance certifications			
Compliance Certification		3.2.6, 4.0.10 and later	3.2.6, 4.0.10 and later
FedRAMP	No	3.2.6, 4.0.10 and later	3.2.6, 4.0.10 and later
HIPAA	No	3.2.6, 4.0.10 and later	3.2.6, 4.0.10 and later
PCI DSS	No	3.2.6, 4.0.10 and later	3.2.6, 4.0.10 and later
Multi-threaded	Yes	No	No
Node type upgrade	No	Yes	No
Engine upgrading	Yes	Yes	Yes
High availability (replication)	No	Yes	Yes
Automatic failover	No	Optional	Required
Pub/Sub capabilities	No	Yes	Yes
Sorted sets	No	Yes	Yes
Backup and restore	No	Yes	Yes
Geospatial indexing	No	2.8.x - No	Yes
		3.2.x and later - Yes	



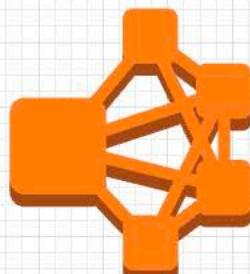
?

Back

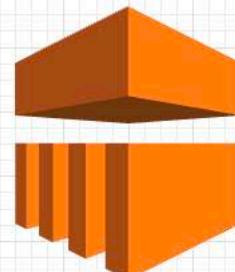
Global Infrastructure

Account & Services

Networking

Well-Architected
Appendix

MapReduce



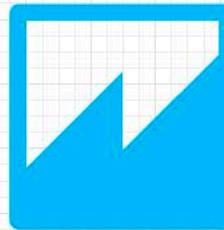
EMR



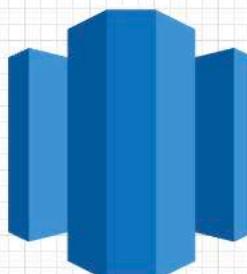
Kinesis



IoT



QuickSight



Redshift



Elasticsearch



MapReduce

MapReduce is a data analysis architecture capable of processing huge quantities of data in a parallel way. It's suitable for data that can be modularized with the same basic operations executed at scale. MapReduce is a process with two main stages: **map** and **reduce**. At a high level, the idea is to split the data into manageable bits, perform operations at scale on those bits, and then recombine for reporting.

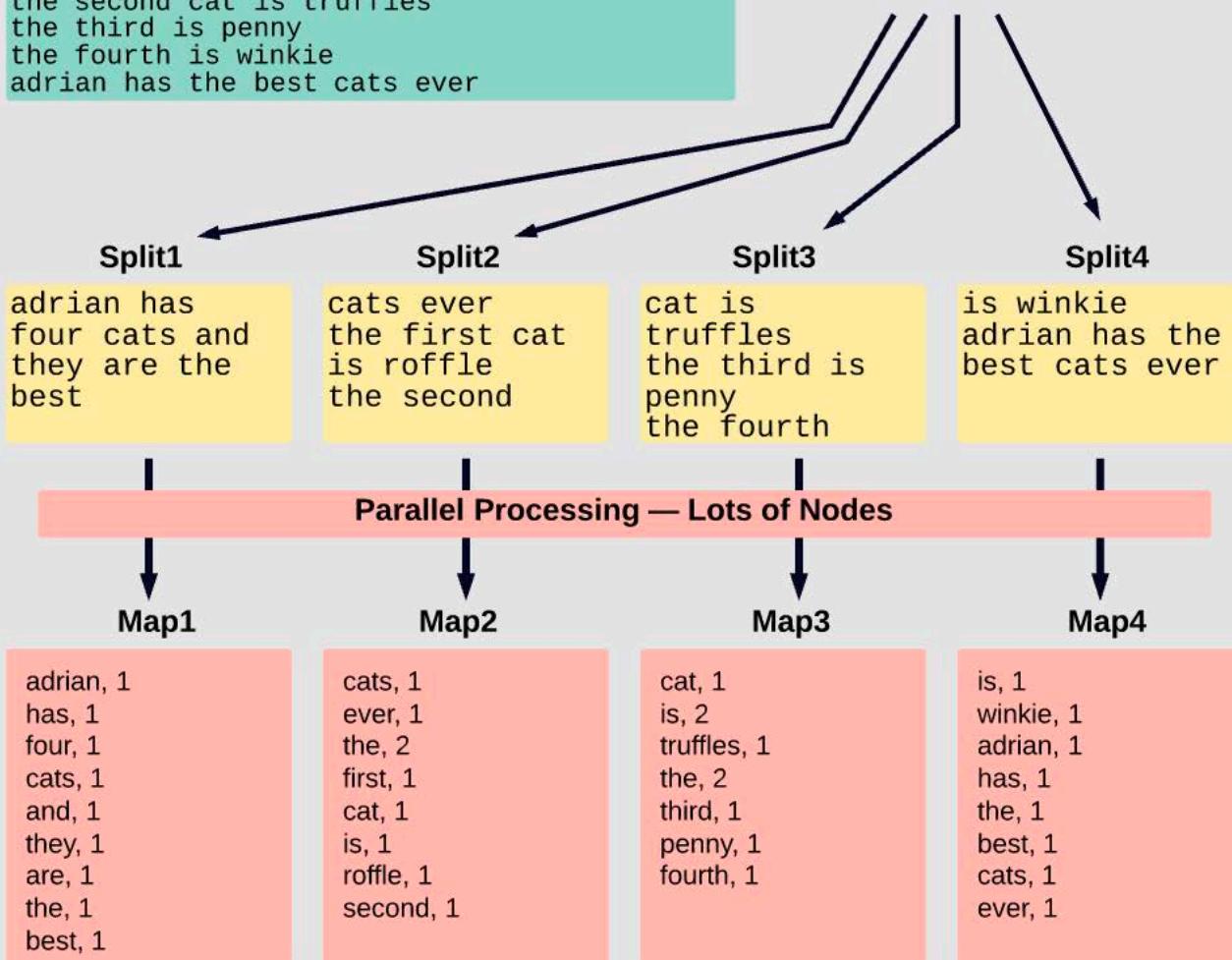
MAP

REDUCE

Input Data

```
adrian has four cats and they are the best
cats ever
the first cat is roffle
the second cat is truffles
the third is penny
the fourth is winkie
adrian has the best cats ever
```

Input data is broken into "splits" based on a split size — e.g., (9)





MapReduce

MapReduce is a data analysis architecture capable of processing huge quantities of data in a parallel way. It's suitable for data that can be modularized with the same basic operations executed at scale. MapReduce is a process with two main stages: **map** and **reduce**. At a high level, the idea is to split the data into manageable bits, perform operations at scale on those bits, and then recombine for reporting.

MAP

REDUCE

MAPS

Shuffle onto Reducers

winkie, 1

fourth, 1

roffle, 1

penny, 1

adrian, 1
adrian, 1cat, 1
cat, 1

are, 1

and, 1

four, 1

is, 1
is, 2
is, 1

truffles, 1

second, 1

has, 1
has, 1

third, 1

best, 1
best, 1ever, 1
ever, 1

they, 1

first, 1

the, 1
the, 2
the, 2
the, 1cats, 1
cats, 1
cats, 1

Reducer

winkie, 1

forth, 1

penny, 1

roffle, 1

and, 1

are, 1

adrian, 2

cat, 2

is, 4

third, 1

truffles, 1

second, 1

they, 1

first, 1

four, 1

has, 2

ever, 2

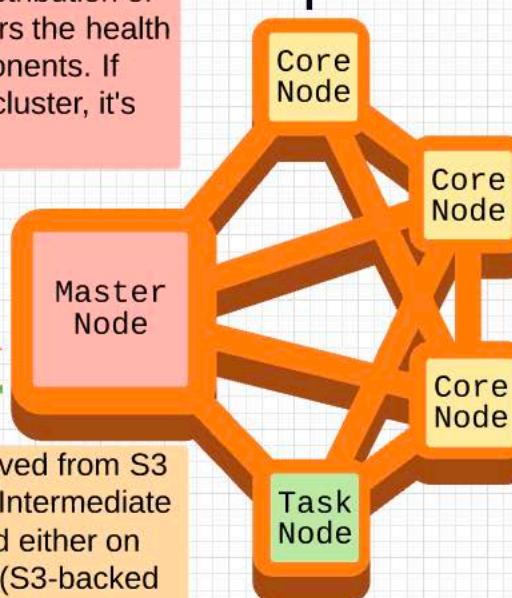
the, 6

cats, 3

winkie 1, fourth 1, truffles 1, second 1, roffle 1, penny 1, has 2, third 1, adrian 2, cat 2, best 2, ever 2, are 1, and 1, they 1, first 1, four 1, is 4, the 6, cats 3



The master node manages the cluster. It acts as the HDFS name node, it controls the distribution of workloads, and monitors the health of all the cluster components. If you need to SSH to a cluster, it's via the master node.



EMR clusters can have zero or more core nodes that are managed by the master node. They act as HDFS data nodes, run the task tracker, and can execute mapping and reduce tasks inside the cluster.

Spot instances can be used as core nodes but risk HDFS data loss if they fail.

Data can be retrieved from S3 and output to S3. Intermediate data can be stored either on HDFS or EMRFS (S3-backed cluster storage), which offers persistence beyond the lifetime of the cluster and resilience from core node failure.



Task nodes are optional in an EMR cluster — they can be used to execute tasks but don't have any involvement in HDFS cluster storage.

Task nodes are ideal when spot instances are used. The processes managing task execution run on core nodes — the compute can occur on task nodes.

Public or Private Subnet

By default, all cluster instances are within the same Availability Zone to allow fast and efficient intra-cluster communications.



Master / Core / Task Nodes



Back

Global Infrastructure

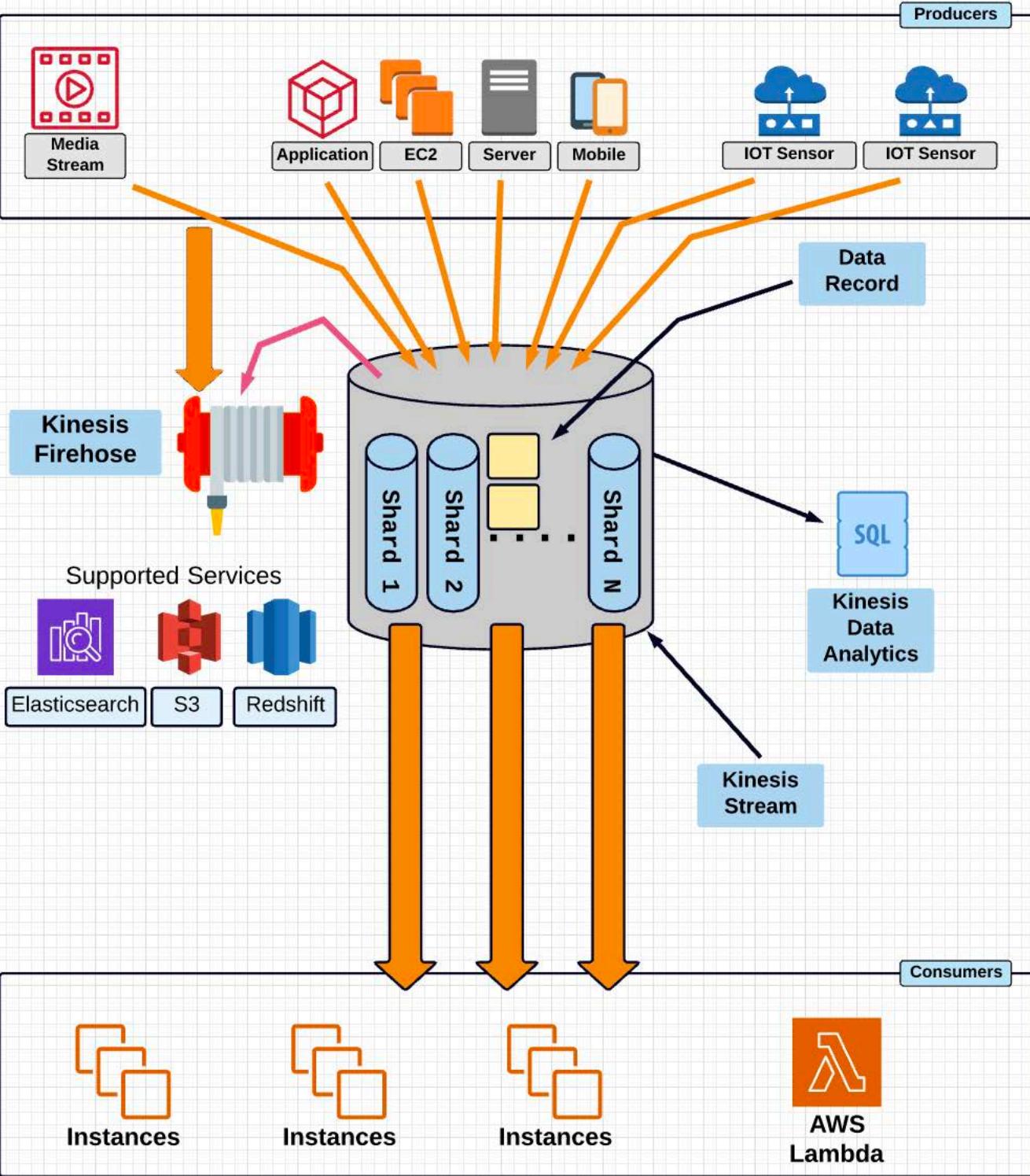
Account & Services

Networking

Well-Architected

Appendix

Kinesis allows the collection of large streams of data records and live video streams in real time. Data can be consumed by multiple consumers .





Kinesis Data Firehose



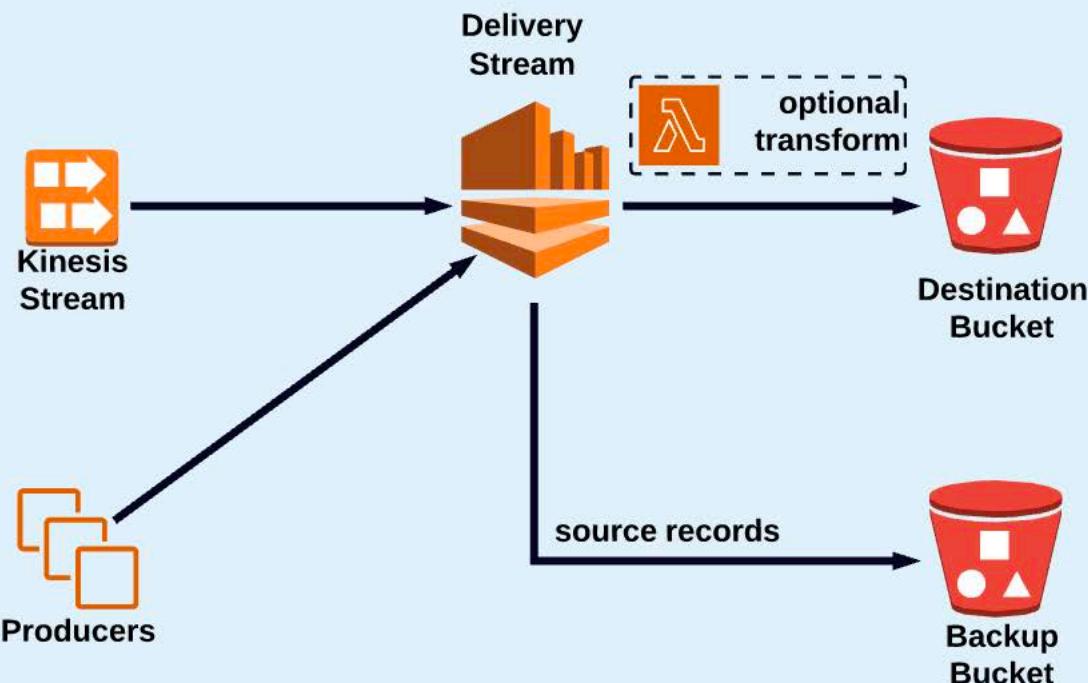
Data Firehose is a managed service that can deliver real-time data to supported destinations. It consumes data from either a Kinesis stream or traditional producers.

Firehose currently supports a number of destinations, including:

- Simple Storage Service (S3)
- Redshift
- Elasticsearch
- Splunk

It also allows for data transformation during the movement between producer and destination using Lambda functions. Examples include:

- Apache log to JSON or CSV
- Syslog to JSON or CSV





?

Back

Global Infrastructure

Account & Services

Networking

Well-Architected
Appendix

Kinesis Stream

X

A Kinesis data stream can be used to collect, process, and analyze a large amount of incoming data. A data stream is accessible from inside AWS VPCs or from the public internet by an unlimited number of producers.

Kinesis streams can scale from low amounts of incoming data to an unlimited level of load using a multi-shard architecture.

Kinesis streams scale to meet capacity requirements and include storage for all incoming data for the stream retention period — 24 hours is standard, with the ability to increase to seven days for an additional charge.

Data Records (see below) are added by **producers** to a Kinesis stream and can be read by **consumers**.

Kinesis Shard

Kinesis shards are how Kinesis streams scale. A stream has at least one shard that provides 1 MiB of ingestion capacity and 2 MiB of consumption capacity. Shards can be added to streams to scale the performance on that stream.

`shards_required = max (ingestion_in_KiB/1024, read_requirements_in_KiB/2048)`

Per producer bandwidth = stream bandwidth / number of producers

Per consumer bandwidth = stream bandwidth / number of consumers (* **enhanced fanout**)

1,000 written data records per second per shard - 1 MiB per record.

Kinesis Data Record

A data record is the basic entity written to and read from Kinesis streams. A data record consists of a **sequence number**, a **partition key**, and a **data blob**. The data blob can be up to 1 MB in size and is not altered by Kinesis in any way. The sequence number is allocated by Kinesis and is unique within its shard.

[Back](#)[Global Infrastructure](#)[Account & Services](#)[Networking](#)[Well-Architected Appendix](#)

Kinesis Stream



A Kinesis data stream can be used to collect, process, and analyze a large amount of incoming data. A data stream is accessible from inside AWS VPCs or from the public internet by an unlimited number of producers.

Kinesis streams can scale from low amounts of incoming data to an unlimited level of load using a multi-shard architecture.

Kinesis streams scale to meet capacity requirements and include storage for all incoming data for the stream retention period — 24 hours is standard, with the ability to increase to seven days for an additional charge.

Data Records (see below) are added by **producers** to a Kinesis stream and can be read by **consumers**.

Kinesis Shard

Kinesis shards are how Kinesis streams scale. A stream has at least one shard that provides 1 MiB of ingestion capacity and 2 MiB of consumption capacity. Shards can be added to streams to scale the performance on that stream.

`shards_required = max (ingestion_in_KiB/1024, read_requirements_in_KiB/2048)`

Per producer bandwidth = stream bandwidth / number of producers

Per consumer bandwidth = stream bandwidth / number of consumers (* **enhanced fanout**)

1,000 written data records per second per shard - 1 MiB per record.

Kinesis Data Record

A data record is the basic entity written to and read from Kinesis streams. A data record consists of a **sequence number**, a **partition key**, and a **data blob**. The data blob can be up to 1 MB in size and is not altered by Kinesis in any way. The sequence number is allocated by Kinesis and is unique within its shard.



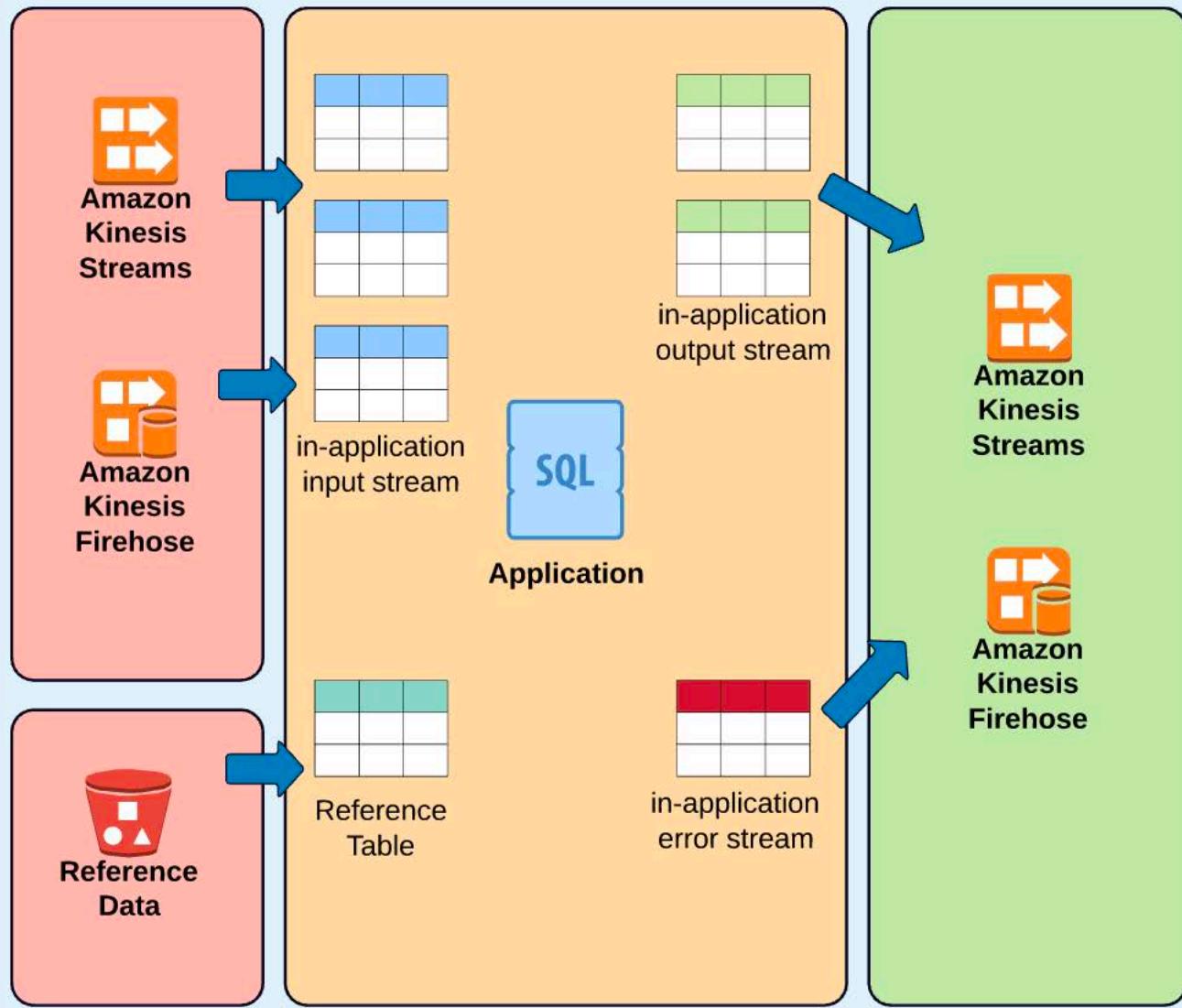
Kinesis Data Analytics:



Kinesis Data Analytics for SQL applications is a Kinesis feature allowing real-time processing and analysis of stream data using standard Structured Query Language (SQL).

Data Analytics can be used for a few types of scenarios:

- Time-series analytics
- Real-time dashboards
- Real-time metrics



[Back](#)[Global Infrastructure](#)[Account & Services](#)[Networking](#)[Well-Architected](#)[Appendix](#)

Kinesis Producers



A Kinesis producer is any device that puts **data records** into a Kinesis stream.

Examples of producers include:

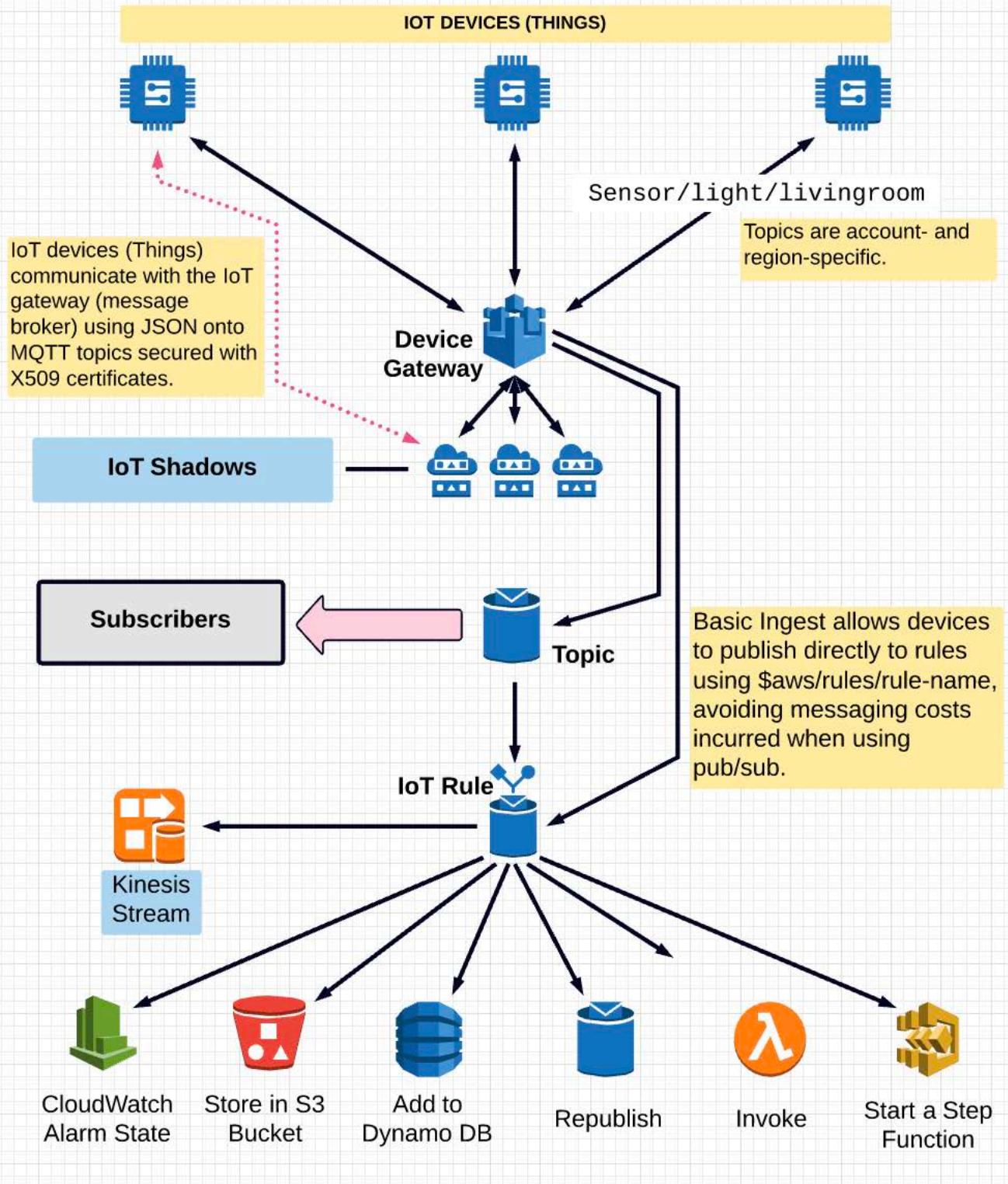
- IoT devices
- Mobile applications
- Applications
- EC2 instance
- On-premises server

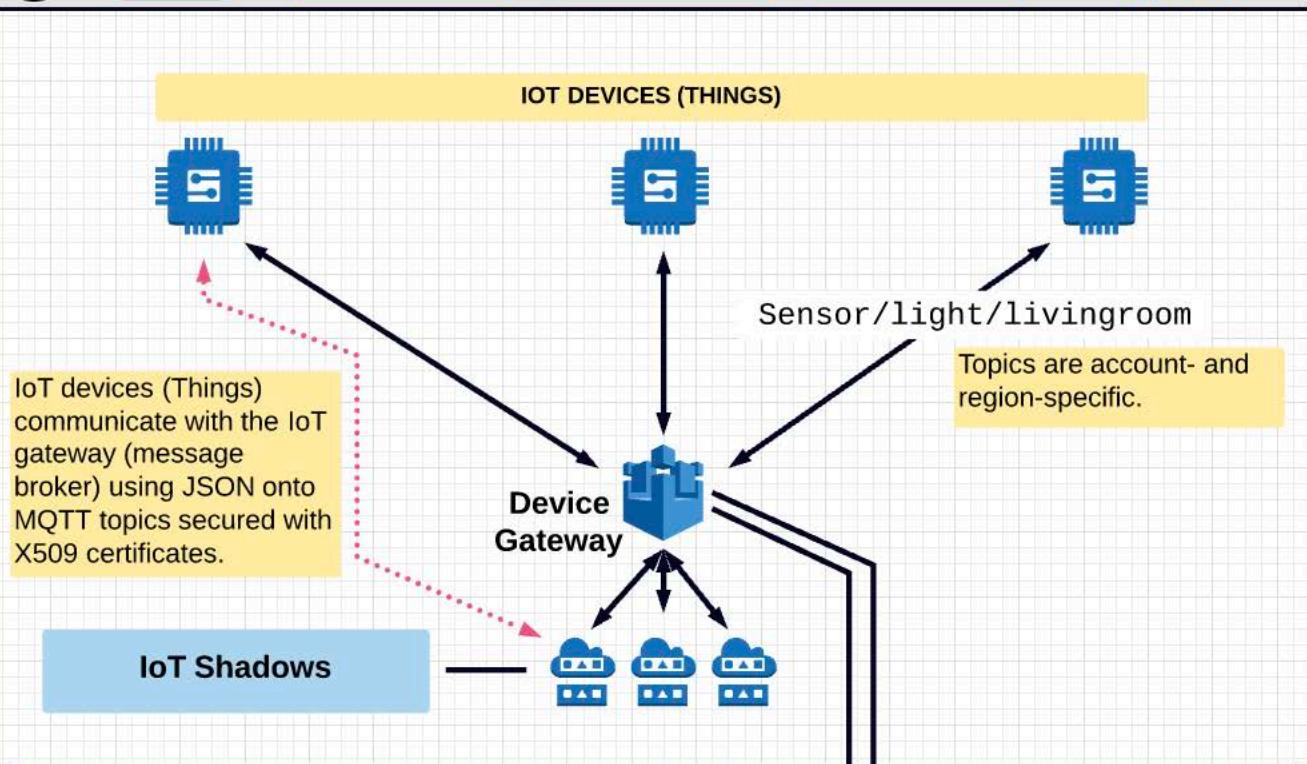
A producer puts data into a stream by specifying the name of the stream, a partition key, and the data to be added. The partition key allows Kinesis to select the **stream** used to ingest the data.

Kinesis Consumers

A consumer is any entity that consumes records from a Kinesis stream. Examples of consumers include:

- EC2 instances running the Kinesis Consumer Library (KCL)
- Lambda functions set to invoke based on stream data records
- Kinesis Firehose





IoT Device Shadows

Communication between an IoT device and the IoT gateway may be sporadic and unreliable. IoT maintains a device shadow that stores the state of the device as last reported. The shadow can be queried rather than communicating directly. The device shadow can also be written to, for example, to set a desired state, which the device will receive the next time it communicates with the gateway.

```
$aws/things/<thing-name>/shadow/update  
$aws/things/<thing-name>/shadow/update/documents
```

```
{  
  "reported" : {  
    "thingname" : "catfood_bowl",  
    "location" : {  
      "lat" : 64.7511  
      "lon" : -147.3494  
    },  
    "status" : "full"  
  }  
}
```



Amazon QuickSight

Amazon QuickSight is a service that provides data visualization. It can integrate with other services to obtain data and present it in a visual and interactive way.

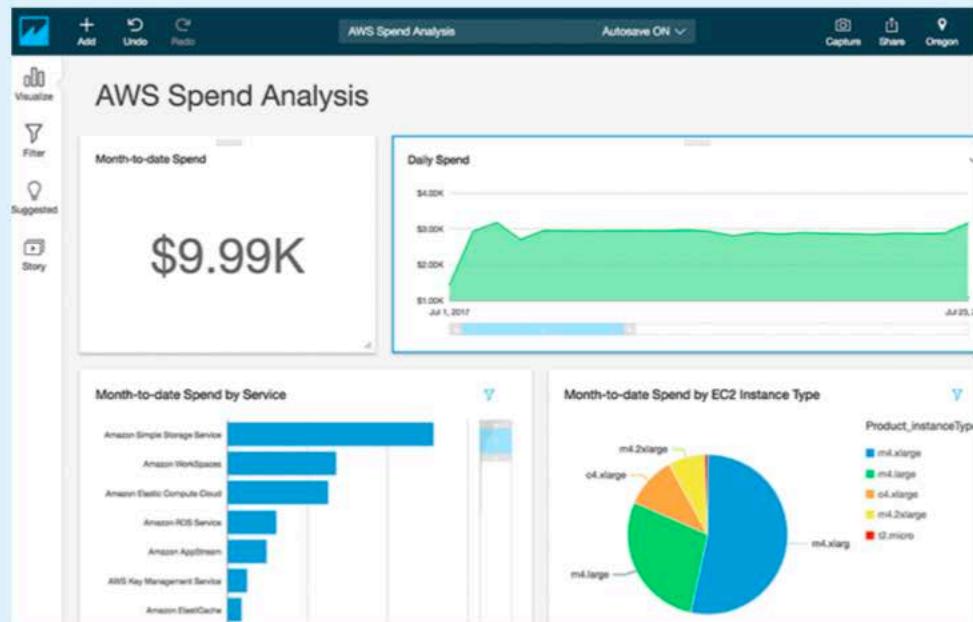
QuickSight is ideally suited when you need to collate data from multiple sources, interact with data visually, or adjust data in an interactive way.

This could be dashboards for a business process, monitoring and metrics for an infrastructure platform, or a visual indication of a fleet of IoT devices.

QuickSight can interact with a large number of data sources, including:

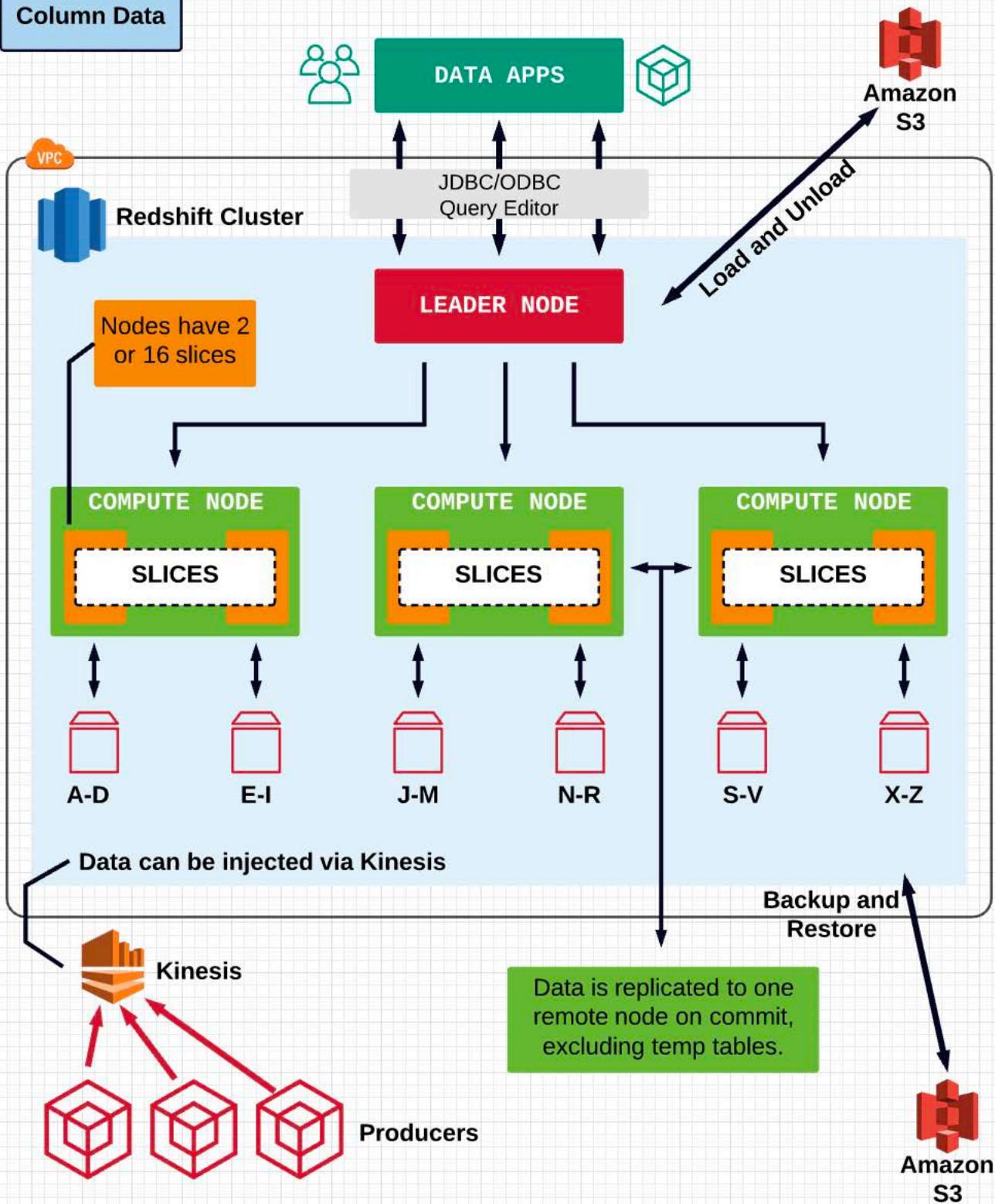
- Amazon Athena
- Amazon Aurora
- Amazon Redshift
- Amazon S3
- Apache Spark 2.0
- MariaDB
- MS SQL 2012+
- MySQL 5.1+

QuickSight can also interact with unsupported data sources by using supported integrations as an intermediary.





Column Data



```
select max(Age), min(Age) from cats
```

Column Based

CATID	Name	Age
0001	Roffle	6
0002	Penny	5
0003	Winkie	3
0004	Truffles	7
...
10000000000000001	Mr Wiggles	57

Row Based

CATID	Name	Age
0001	Roffle	6
0002	Penny	5
0003	Winkie	3
0004	Truffles	7
...
10000000000000001	Mr Wiggles	57

Row-based databases store data in a row arrangement on disk.

Column-based databases store data column by column.

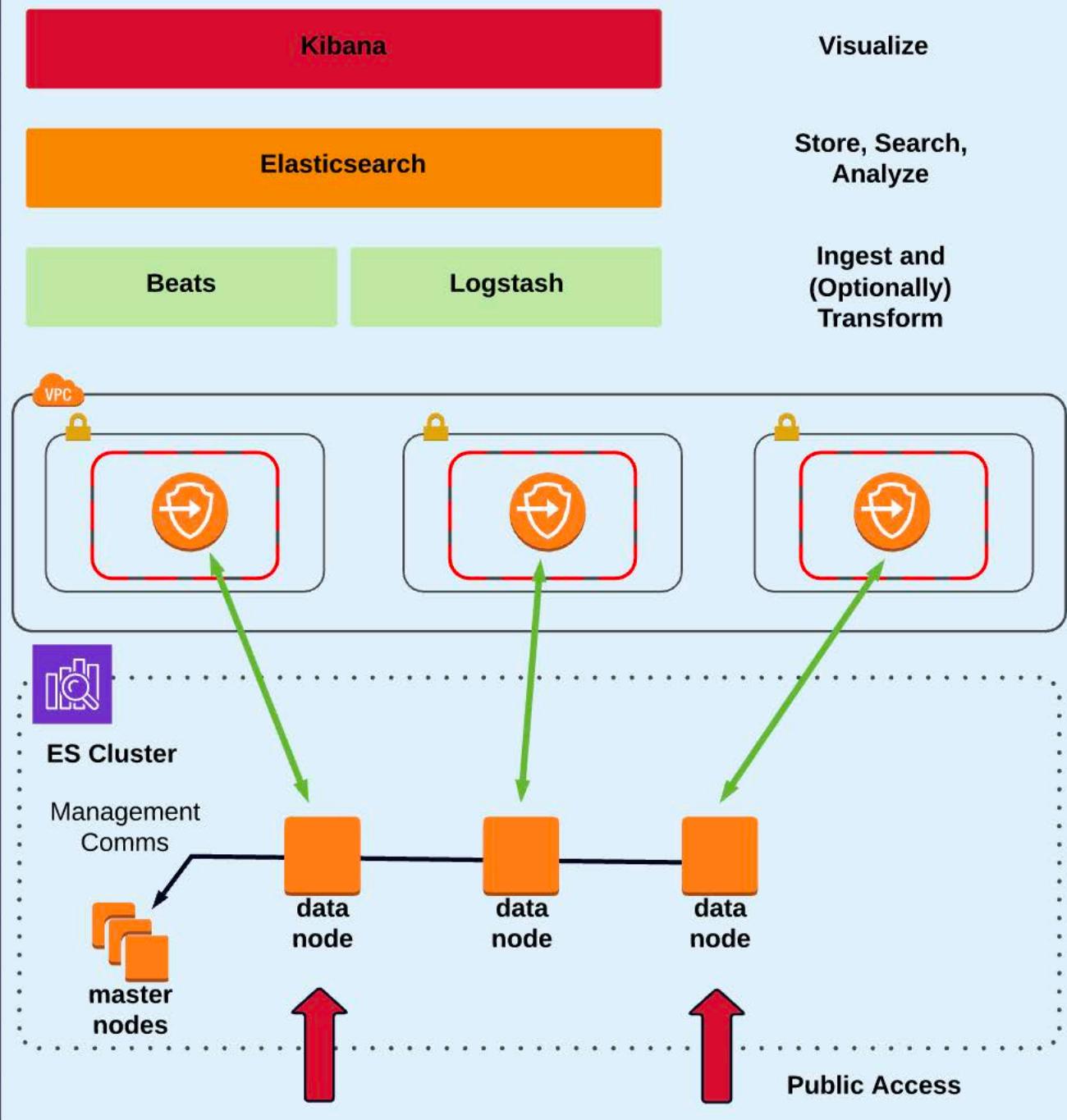
For queries involving specific columns, the data used when querying column-based databases is significantly reduced.

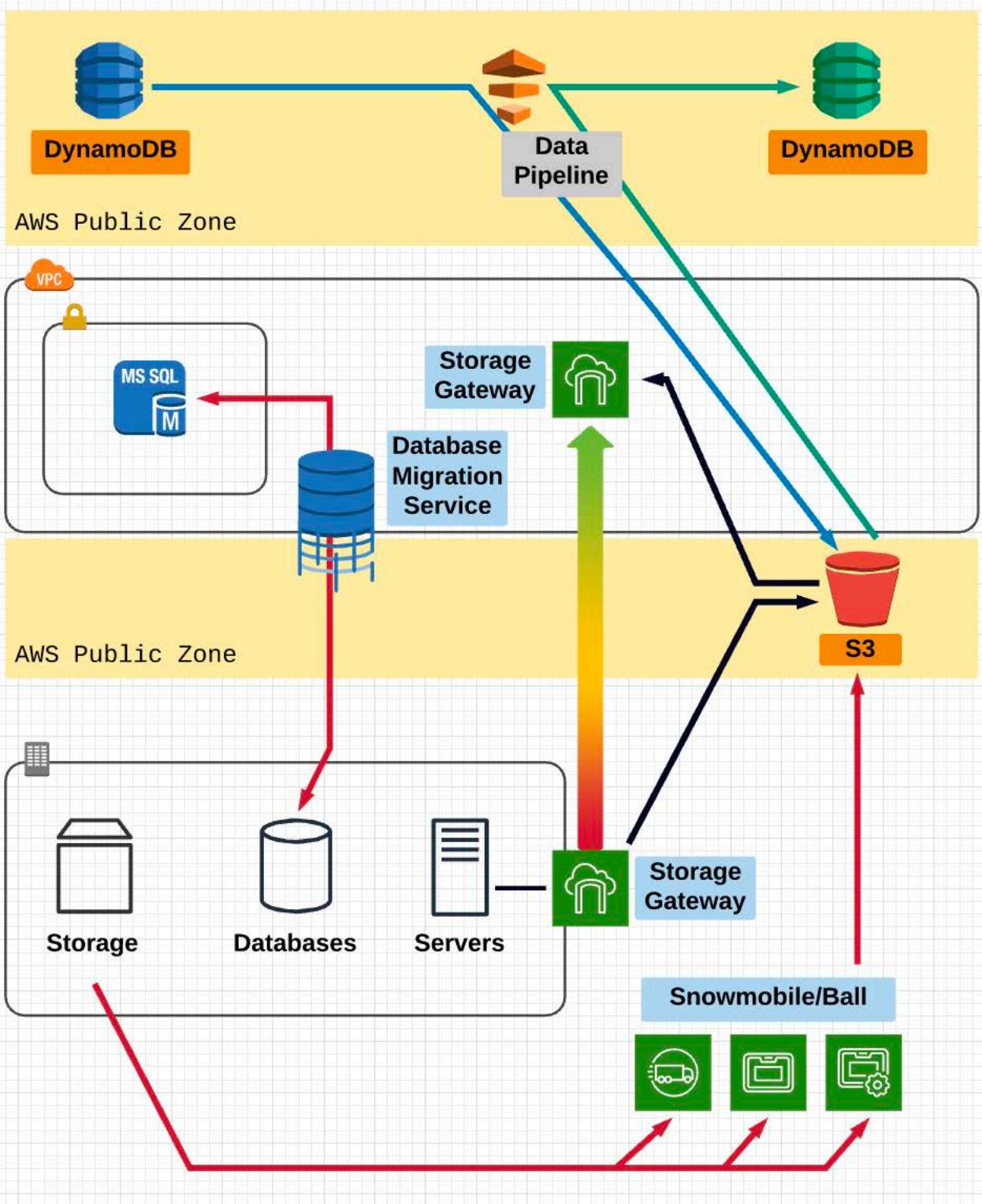
In the example above with five rows, the column-based database reads five attributes. The same query in a row-based database reads 15. The difference would be further amplified with billions of rows.

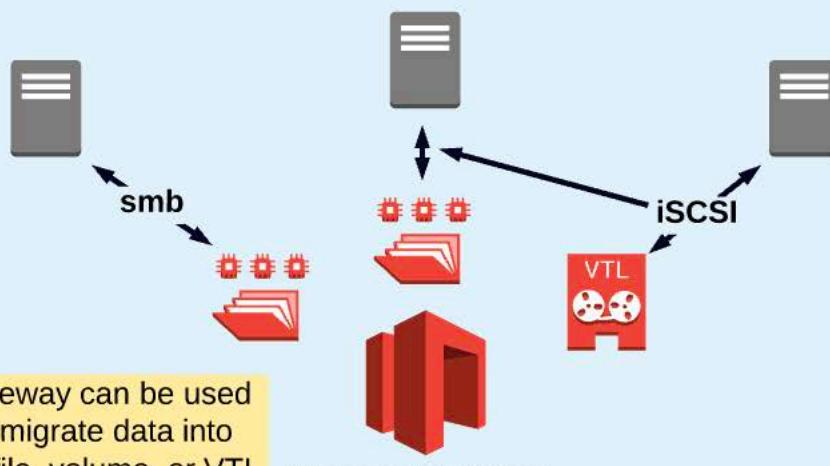


Elasticsearch (ES)

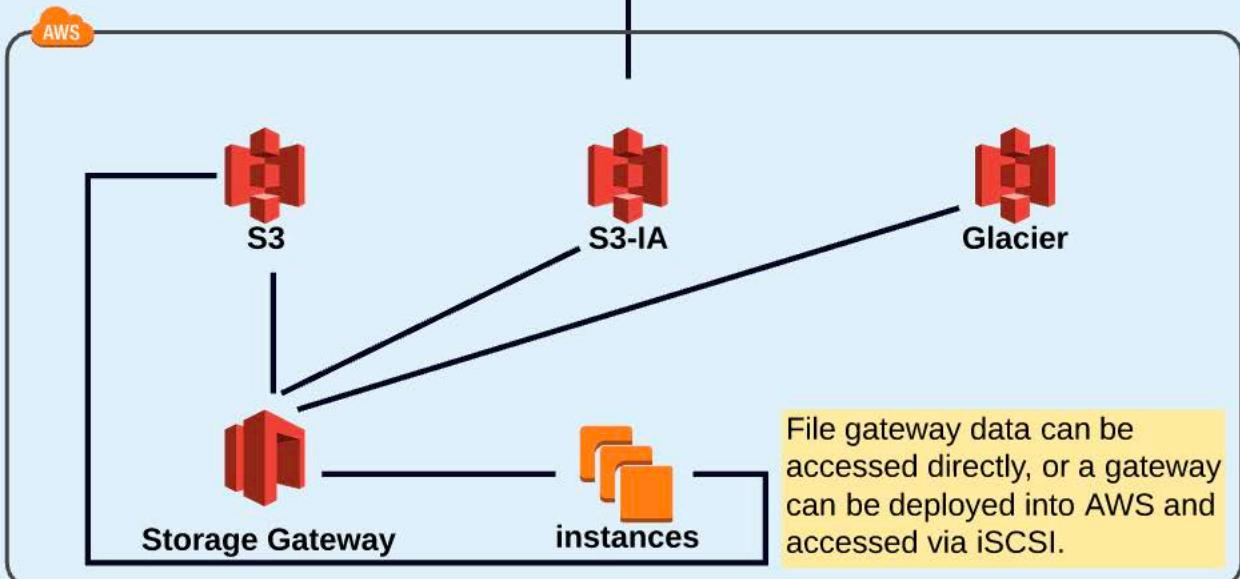
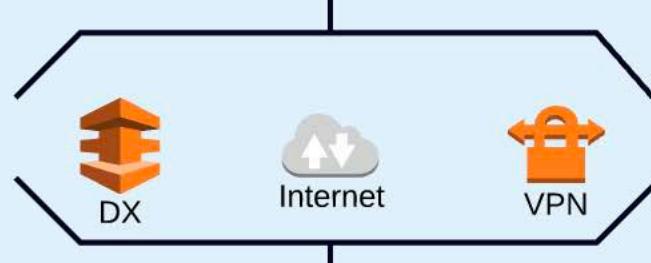
Amazon Elasticsearch is a managed service providing an implementation of the open-source Elasticsearch API and comes with Kibana and Logstash, providing a full implementation of the ELK stack.

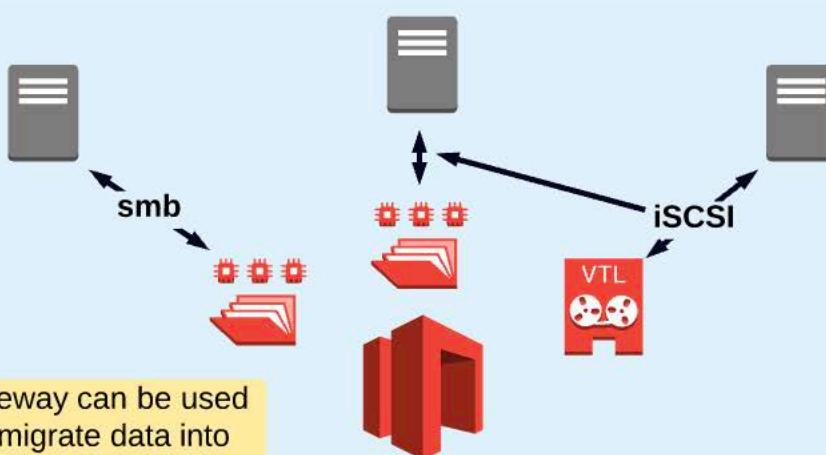






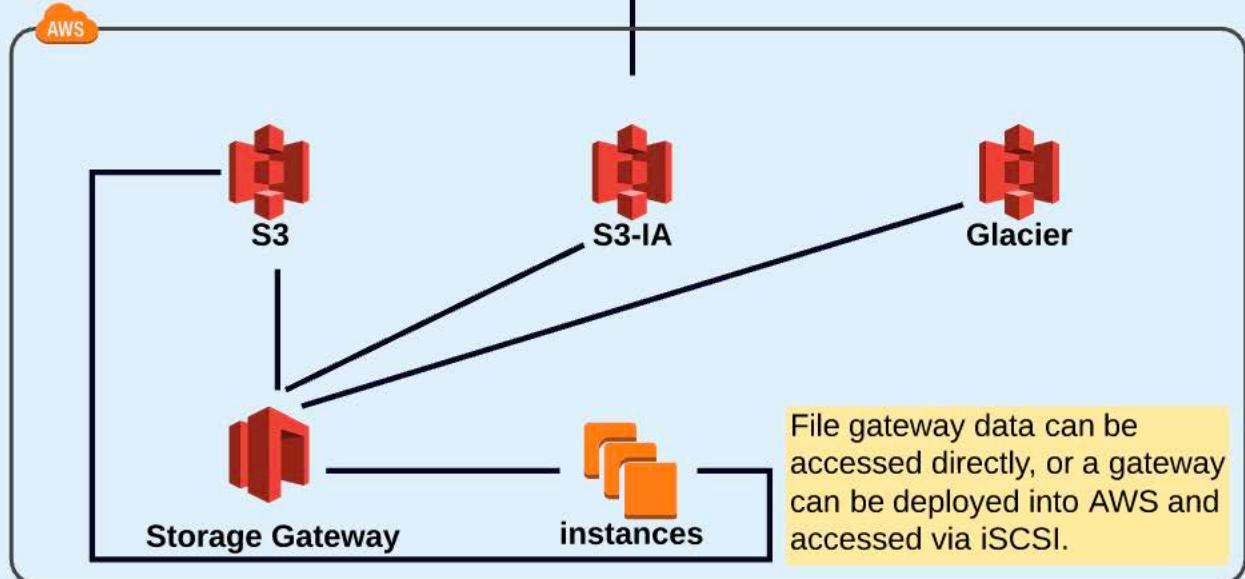
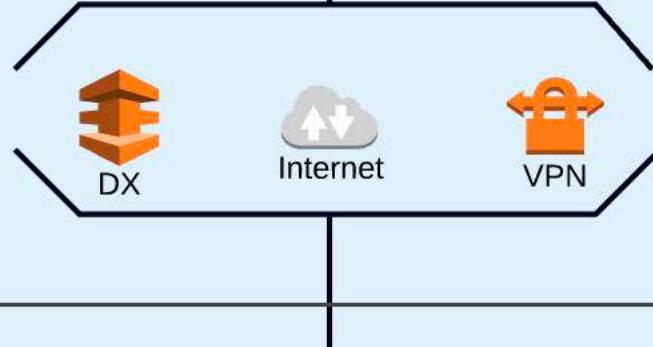
For gradual migrations or lesser data amounts, Storage Gateway is more suited than a physical method such as Snowball or Snowmobile.





Storage Gateway can be used to gradually migrate data into AWS using file, volume, or VTL gateway types.

For gradual migrations or lesser data amounts, Storage Gateway is more suited than a physical method such as Snowball or Snowmobile.



File gateway data can be accessed directly, or a gateway can be deployed into AWS and accessed via iSCSI.



Snowball

Snowballs are storage appliances provided by AWS that can store 50 TB or 80 TB of data. The device is economical when transferring more than 10 TB of data between an on-premises location and AWS or vice versa. Data is encrypted in transit to/from the Snowball and at rest on the device.

Snowballs are shipped to a customer, loaded up via storage interfaces, and shipped back to AWS where the data is loaded onto S3.

Architecturally, Snowballs should be used when larger quantities of data transfer is required (i.e., anything beyond 10 TB of data). The process of requesting a snowball, loading it with data, and sending it back to AWS can achieve faster effective transfer rates than using the internet/DX/VPN.



Snowball Edge

Snowball Edges are similar to standard Snowballs, but they also provide local compute services, allowing them to run processing at the edge, such as Lambda functions and instances. They are used when data needs to be processed or analyzed during the upload (e.g., remote locations with poor connectivity). There are currently three different Snowball Edge versions: **Snowball Edge Storage Optimized**, **Snowball Edge Compute Optimized**, and **Snowball Edge Compute Optimized with GPU**. Up to 100 TB can be stored per device.

Snowmobile

The Snowmobile is a mobile data center. It can be ordered from AWS, driven to location, connected to your data center, and loaded with huge quantities of data before being returned and loaded into AWS. Generally, the Snowmobile is used for data center migrations, or when 10 PB or more needs to be migrated from a single location. For less than 10 PB or when multiple locations are involved, use one or more Snowball/Snowball Edge. Each Snowmobile can store 100 PB, and they can be used in parallel.





AWS AWS Infrastructure "Container"

AWS Account
(i.e., Production Account)

IAM

AWS Deployment Services

Infrastructure As Code



CloudFormation

Infrastructure Management



OpsWorks

Simple App Deployment



Elastic Beanstalk



S3 Logging



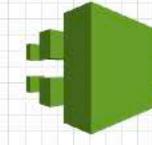
R53 Logging



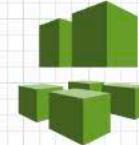
ELB Logging



CloudWatch



CloudTrail



Systems Manager

AWS Operations Services

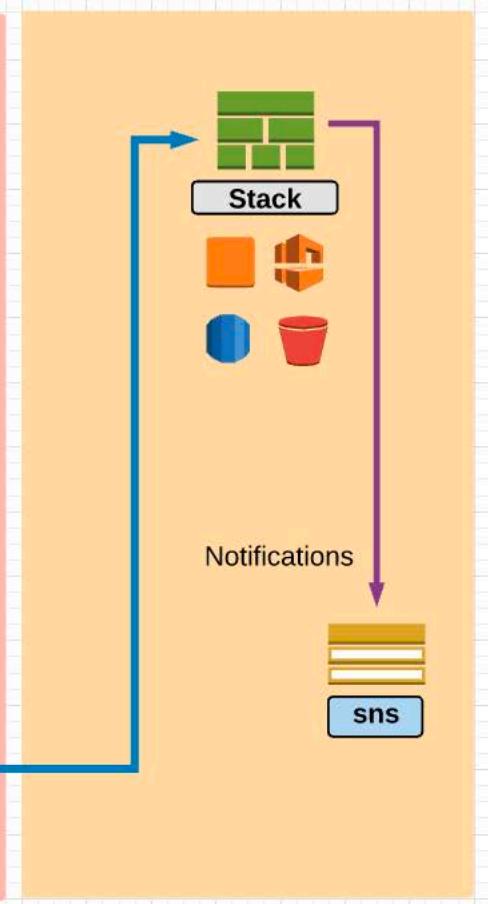
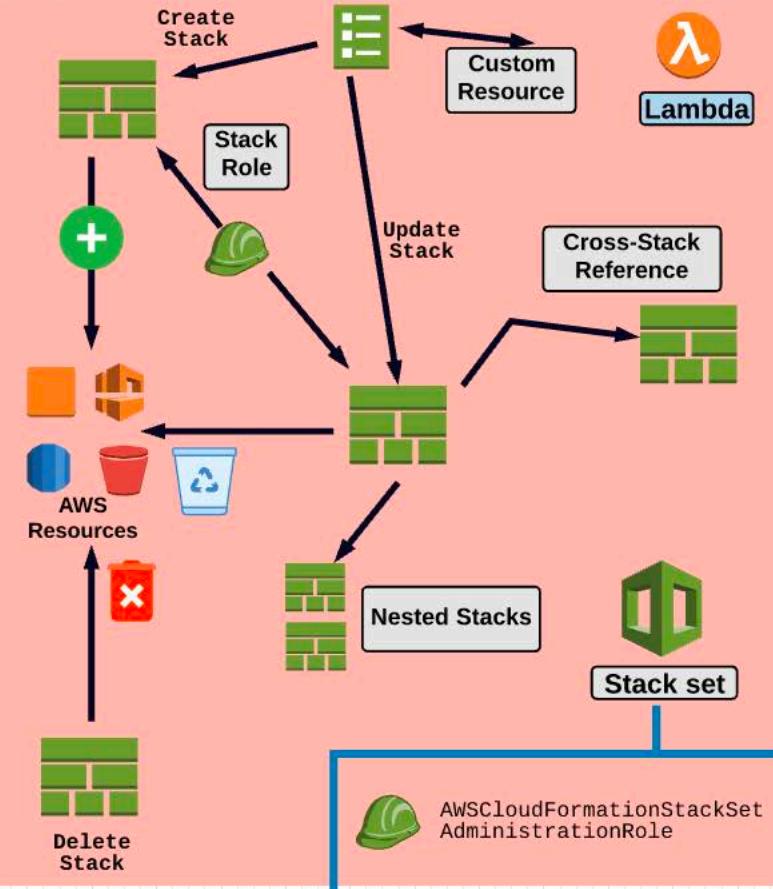




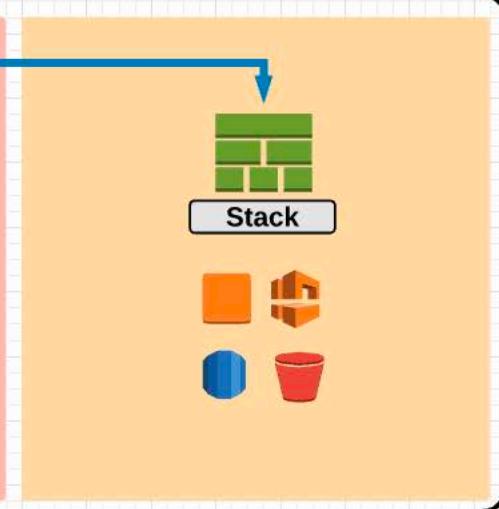
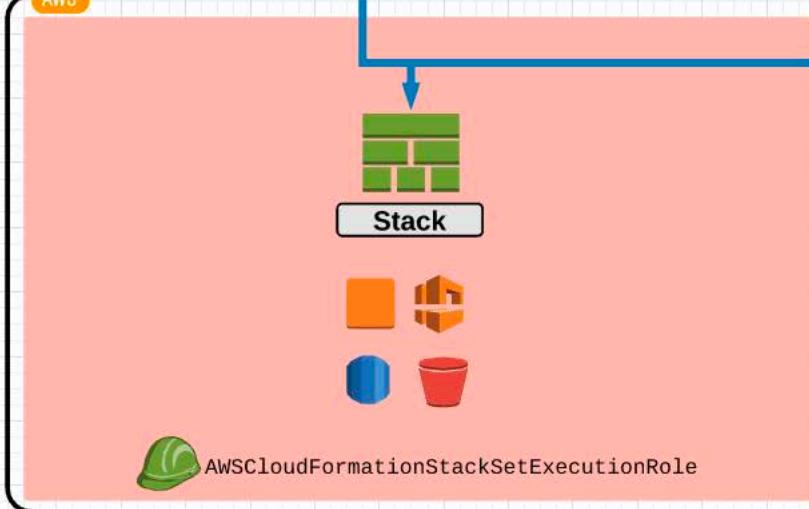
us-east-1

ap-southeast-2

AWS



AWS





AWS

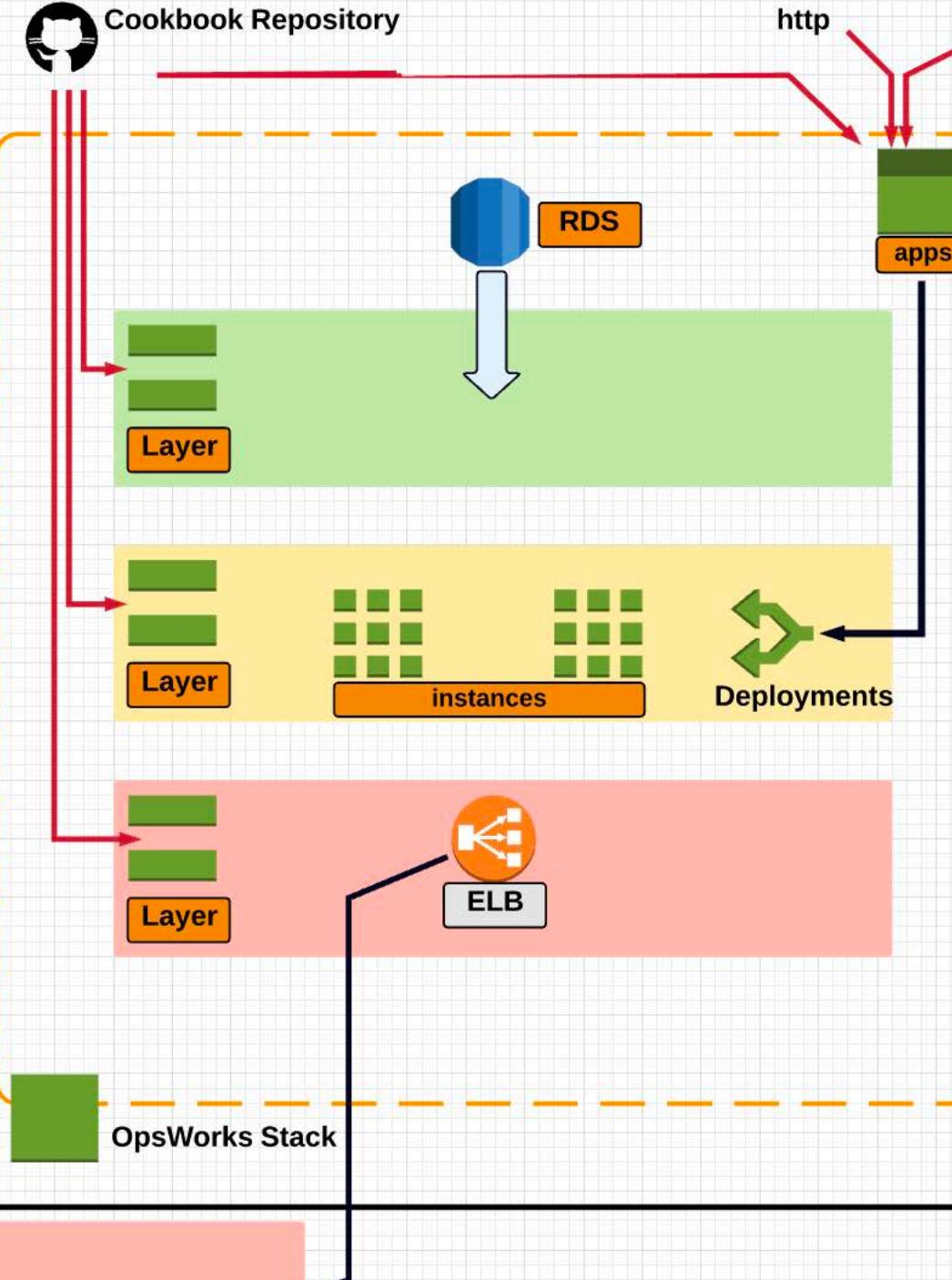


Cookbook Repository

http



S3



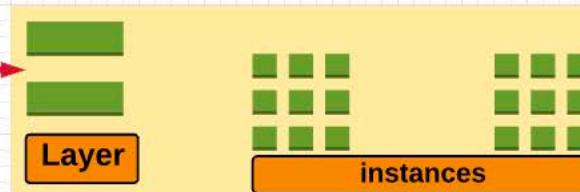
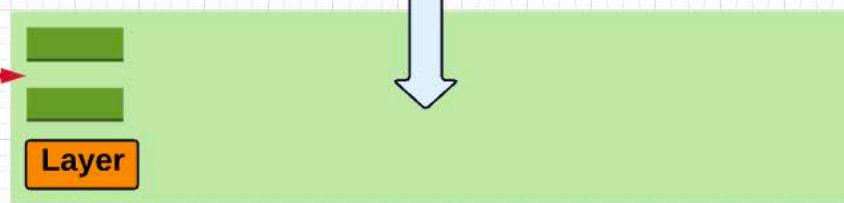
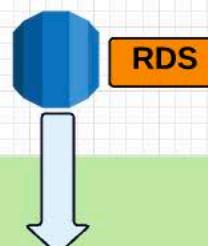


Cookbook Repository

http



S3



Deployments



OpsWorks Stack

Public Internet

A layer within OpsWorks represents a grouping of configurations (recipes). Layers are either an OpsWorks layer, an ECS layer, or an RDS service layer.

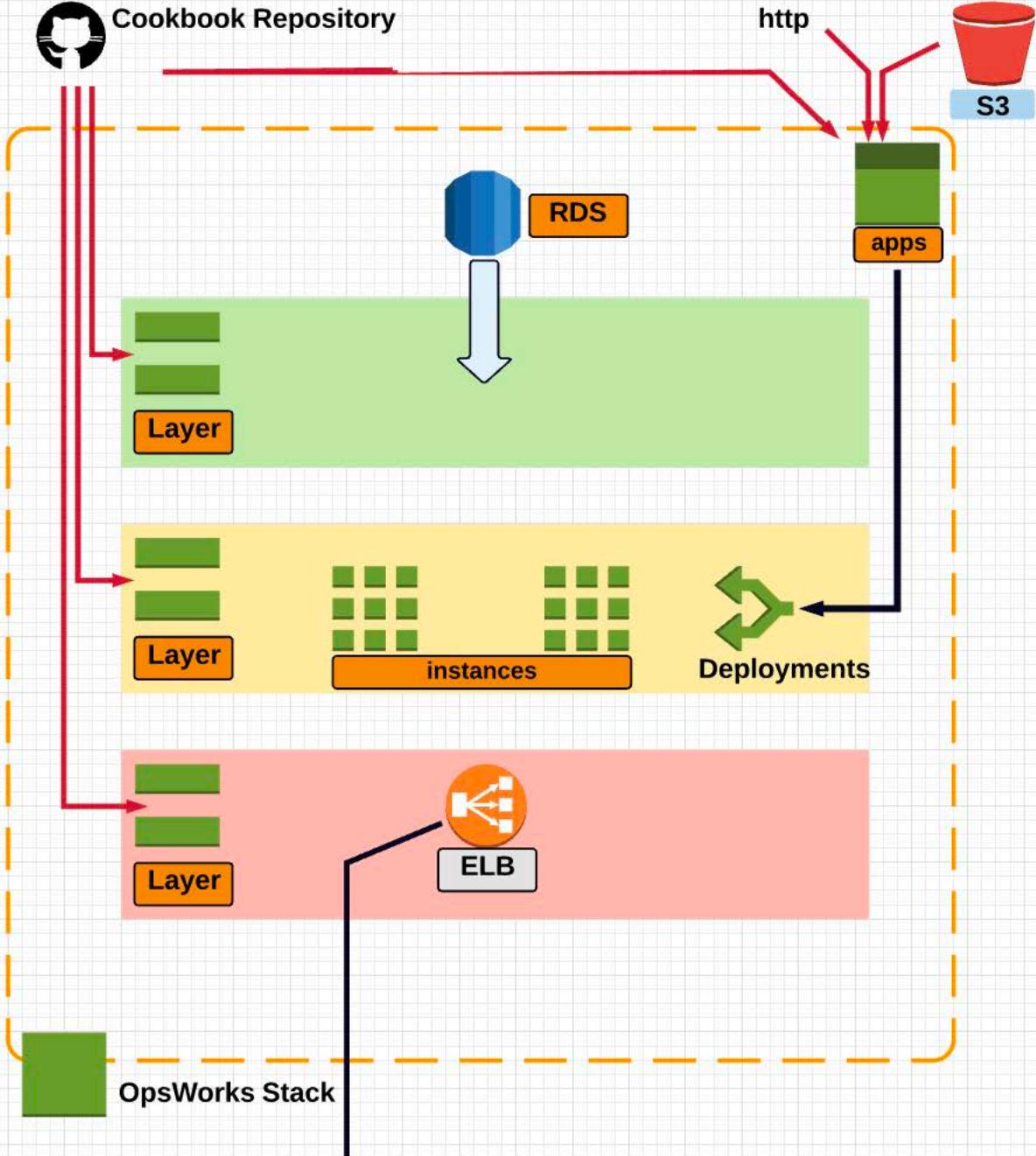




AWS



Cookbook Repository



RDS instances in the same AWS region can be registered with OpsWorks and placed in an RDS service layer. Removing an RDS instance from a layer does *not* delete the RDS instance — it only removes it from the service layer.

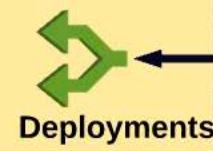
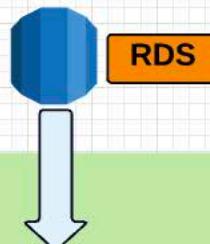


Cookbook Repository

http



S3



ELB

OpsWorks Stack

Public Internet

An app in OpsWorks is code that runs on an OpsWorks instance. Apps can be sourced from GitHub, S3, or downloaded via HTTP. App deployment is done via the deploy recipe. Apps can be connected to an RDS instance.

X

AWS Account
(i.e., Production Account)

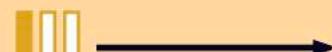
Application: A container of environments, versions, environment configurations



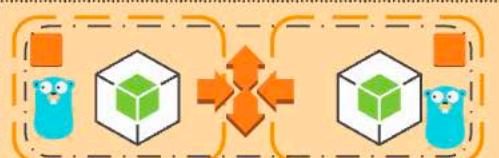
Environment: An isolated environment of a given tier (web server or worker)
Application version: A distinct version of application code (app bundle) that can be deployed into an environment

Elastic Beanstalk Application

ENV - Worker Tier



APP



ENV - Web Server Tier



APP

ELB

Elastic beanstalk Environment

Deployed into

Application Version

APP

ELB

<http://la-blue-env.elasticbeanstalk.com>

<http://la-green-env.elasticbeanstalk.com>

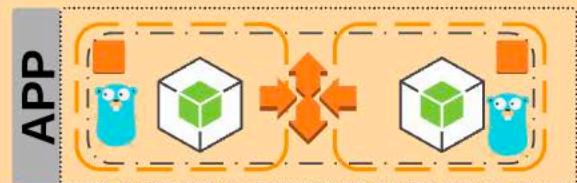
SWAPURL

AWS Account
(i.e., Production Account)

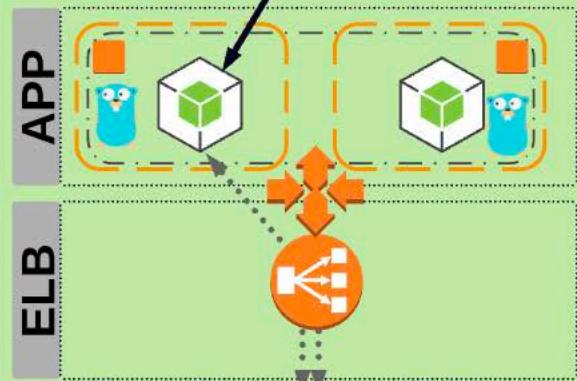
Elastic Beanstalk supports multiple programming languages (Java, PHP, Python, Ruby, Go), web containers (Tomcat, Passenger, and Puma), Docker, and custom platforms you can define and create using Packer. This creates a **platform definition archive**.

bucket

Elastic Beanstalk Application



Elastic beanstalk Environment
Deployed into Application Version



SWAPURL



S3 Logging

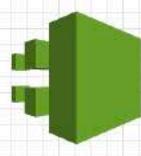


S3 server access logging provides detailed access logs for a specific S3 bucket. Logs for a source bucket are delivered to a target bucket in the same region.

The Log Delivery group needs to be granted access via an entry in the Bucket ACL.



LOG DELIVERY GROUP



AWS Operations Services



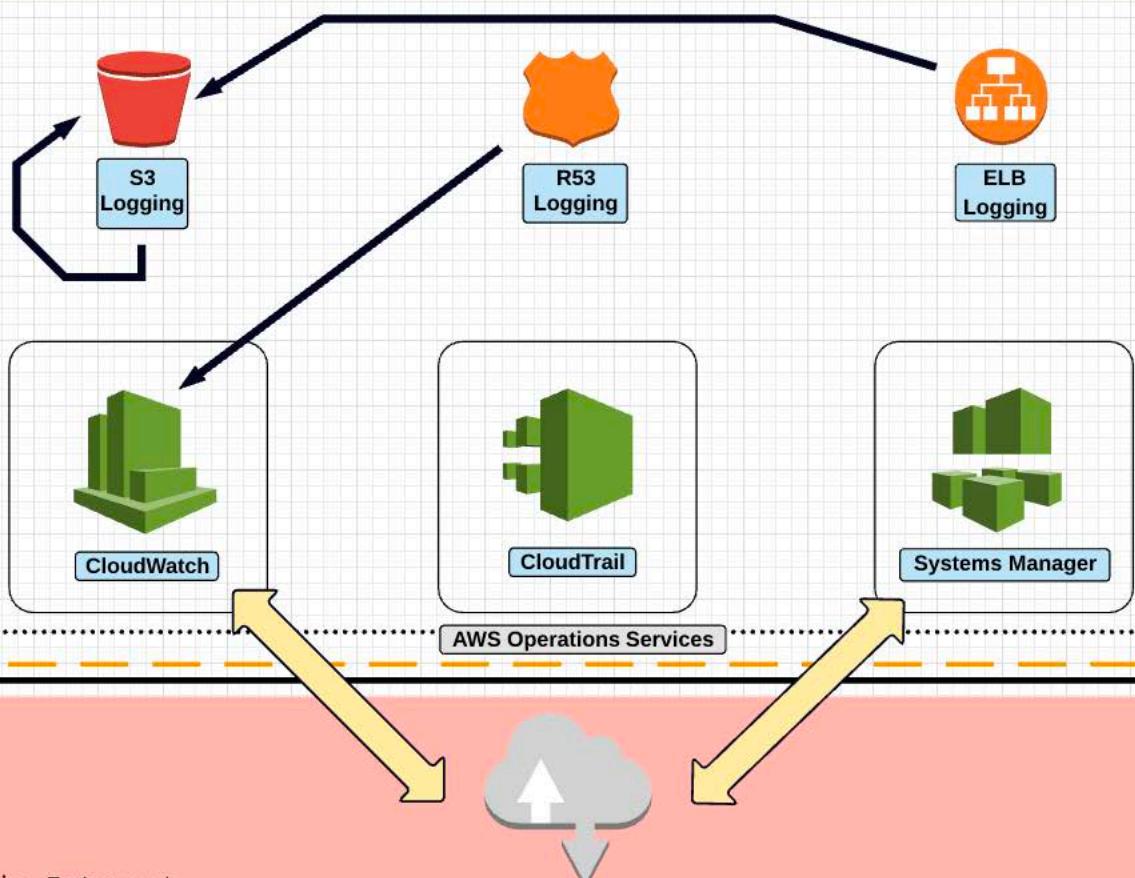
Route 53 Query Logging



Route 53 can log any DNS resolution requests to CloudWatch Logs. Query logging is only available for **public** DNS zones.

Information logged includes:

- The query timestamp
- The DNS zone
- The query type
- The query response code
- The Layer 4 protocol used (TCP/UDP)
- The Route 53 edge location code
- The Resolver IP

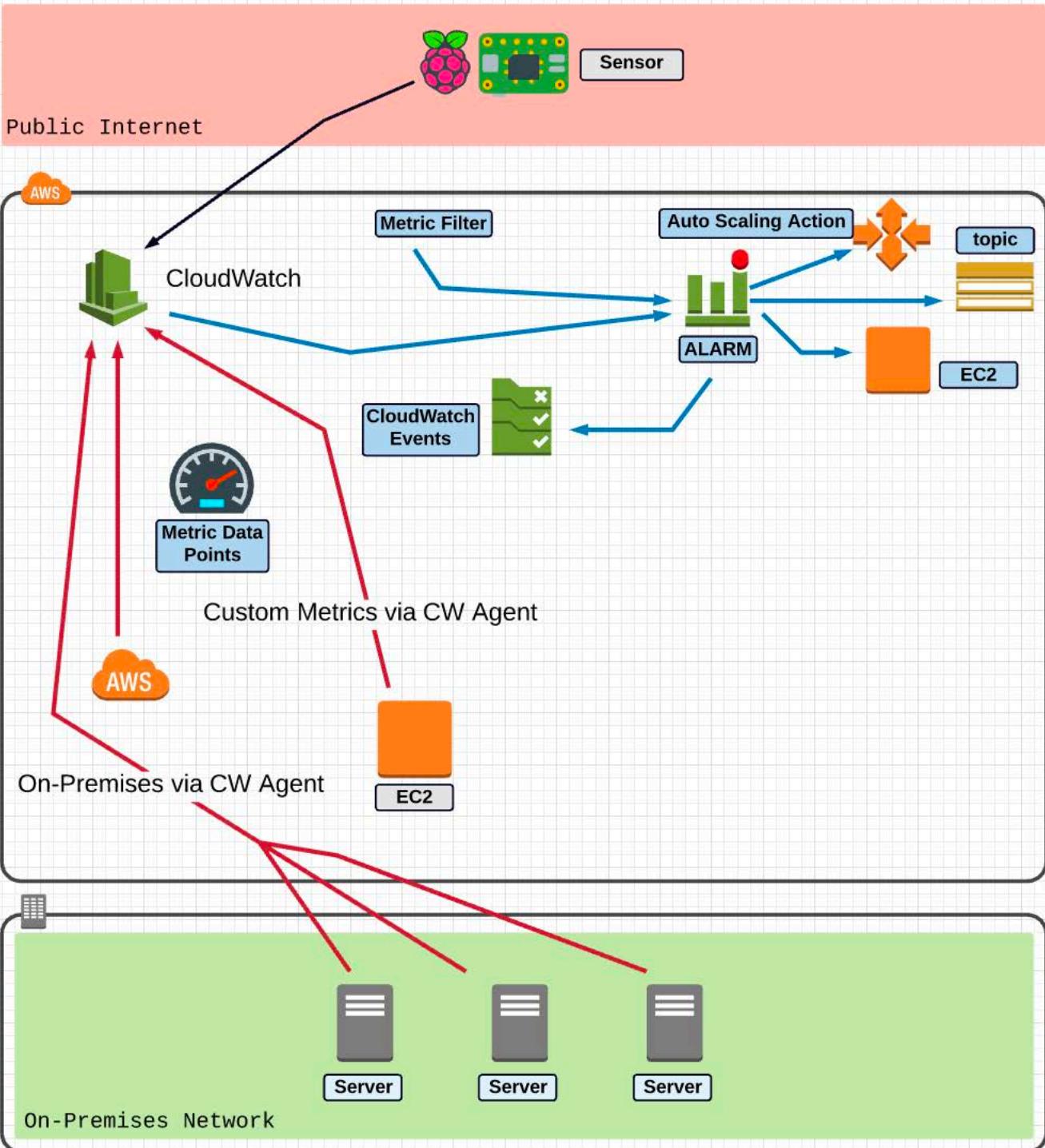




CloudWatch

CloudWatch Logs

CloudWatch Events

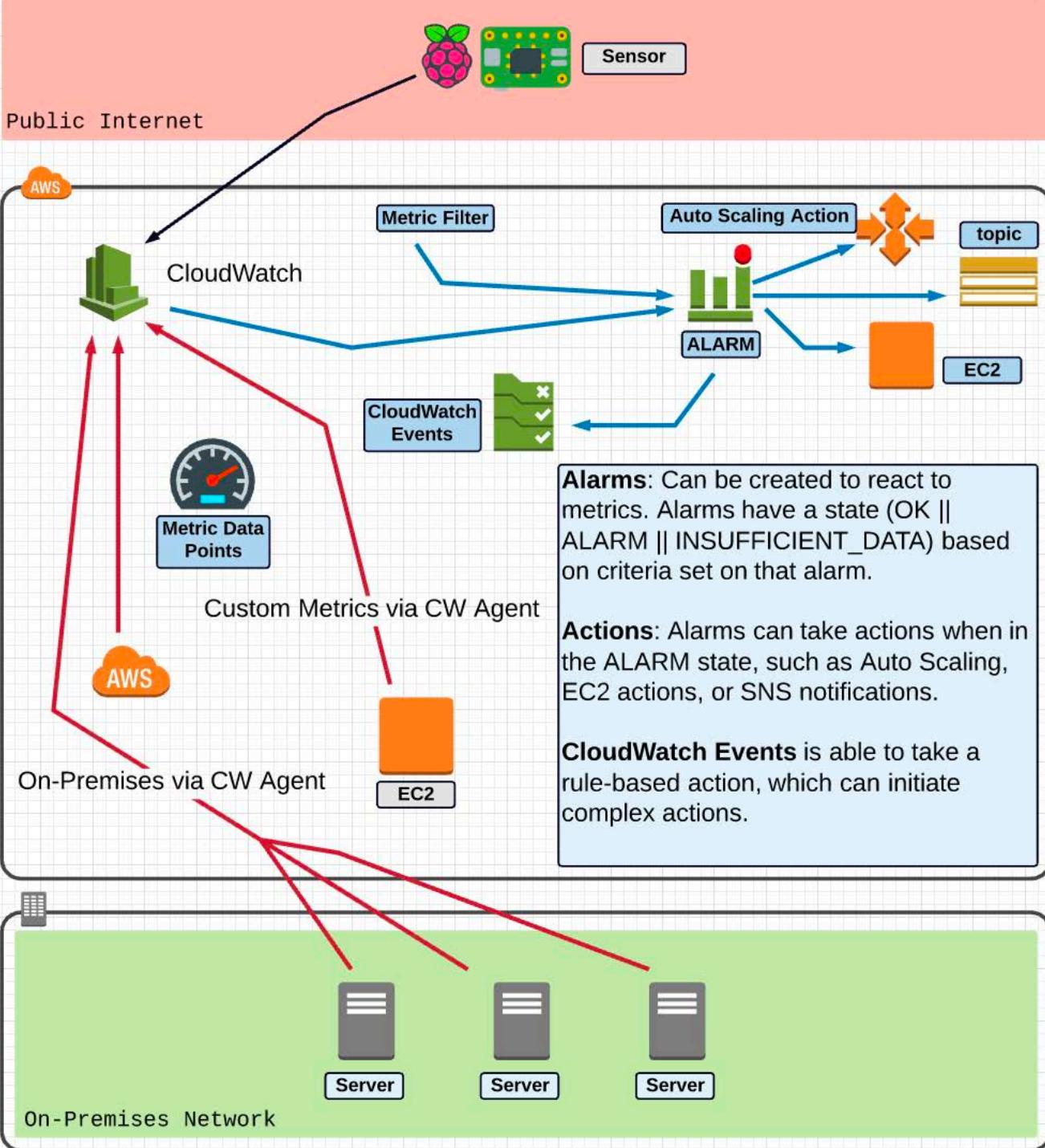




CloudWatch

CloudWatch Logs

CloudWatch Events

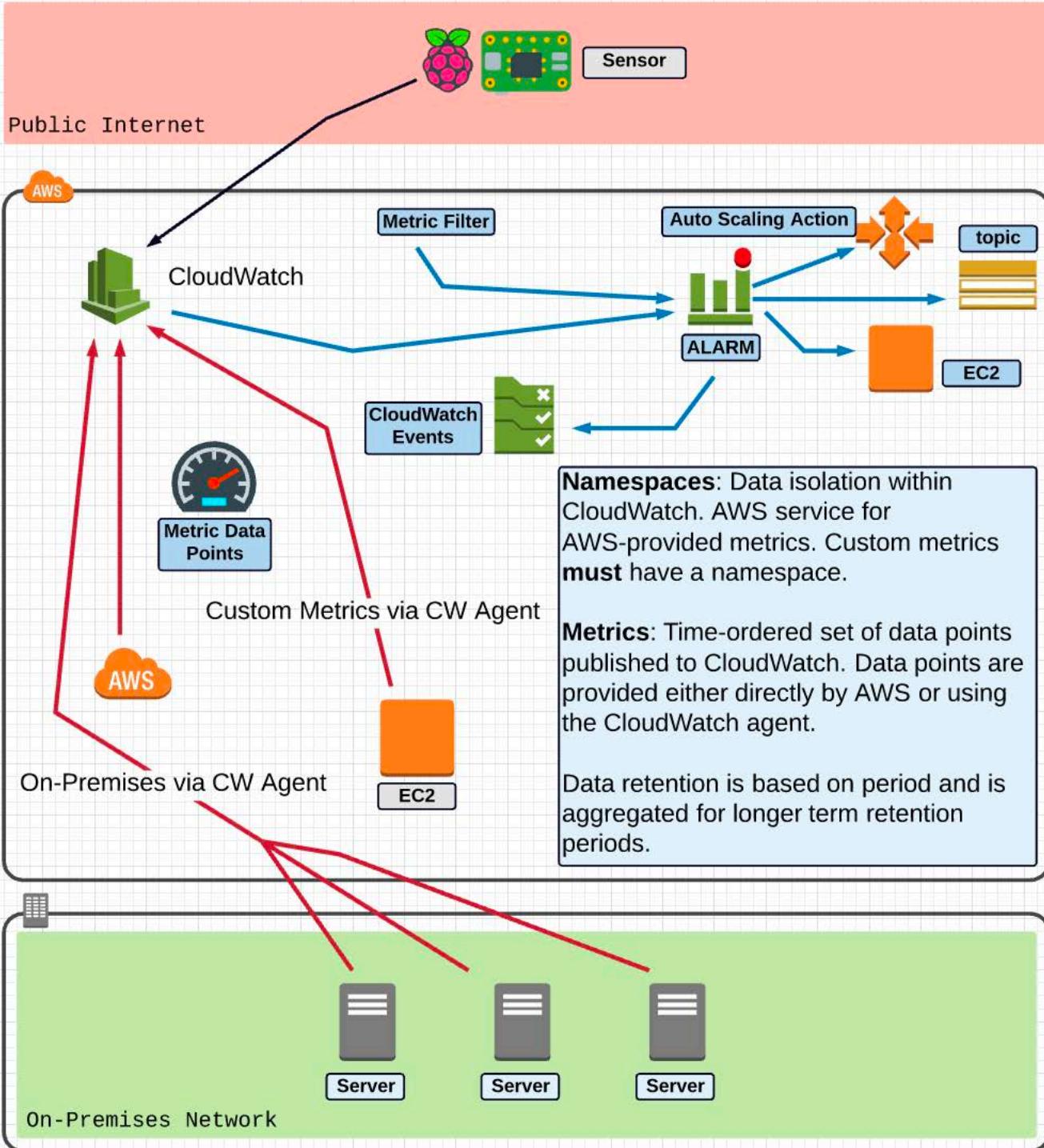




CloudWatch

CloudWatch Logs

CloudWatch Events

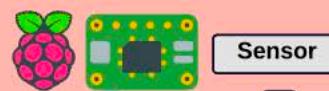




CloudWatch

CloudWatch Logs

CloudWatch Events



Public Internet

AWS

Containers for log streams with same retention, monitoring, and access control



CloudWatch Logs

Metric Filter

Metric filters create metrics and are applied to log groups.

Sensor

CloudWatch Events

ALARM

LOG GROUP

Log Events

Log Stream

Log Stream

Log Stream

YYYYMMDDHHMMSS	MESSAGE

LOG GROUP

A log event is a timestamp and raw message.



AWS services often use CW Logs for log storage.

Sequence of log events with the same source



On-Premises Network



Server



Server



Server



CloudTrail

CloudTrail is the primary service we use for logging in AWS.

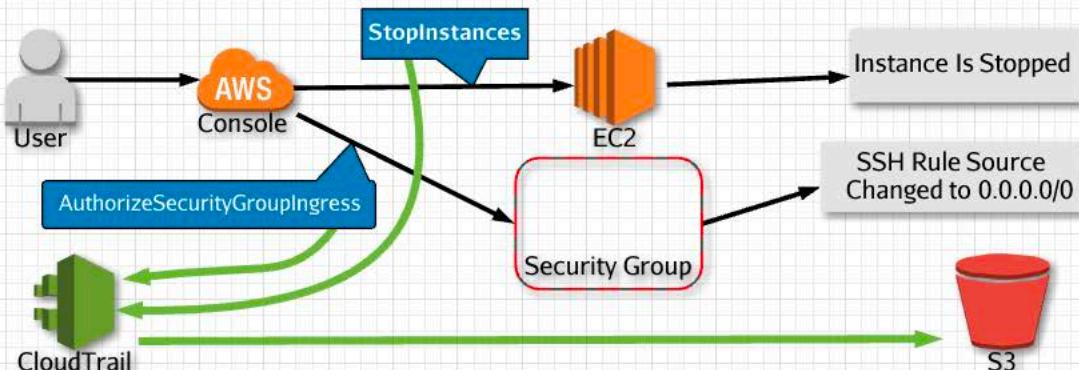
Important Features:

- It logs all the API calls in an AWS account (includes Console, CLI, API/SDK calls)
- It's enabled when your account is created
- Entries can be viewed using the **Event History** (past 90 days)
- **Trail:** A configuration allowing for logs to be sent to an S3 bucket:
 - Single region or multi-region trails can be configured
 - Trails can make multi-account logging possible (more on this later)

Trails have several configuration options:

- **Management events:** Enabling will log control plane events, such as:
 - User login events
 - Configuring security
 - Setting up logging
- **Data events**, which include:
 - Object-level events in S3
 - Function-level events in Lambda
- **Encryption flexibility:**
 - Encrypted in S3 server-side by default but can be changed to KMS
- The logs can be sent to an **S3 bucket** of choice and even prefixed (folders)

Scenario: An IAM user logs in to the AWS Management Console. That user then proceeds to stop an EC2 instance and edit a security group. CloudTrail will log all of these actions. Here is the workflow:





Console

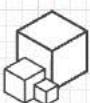


SDKs

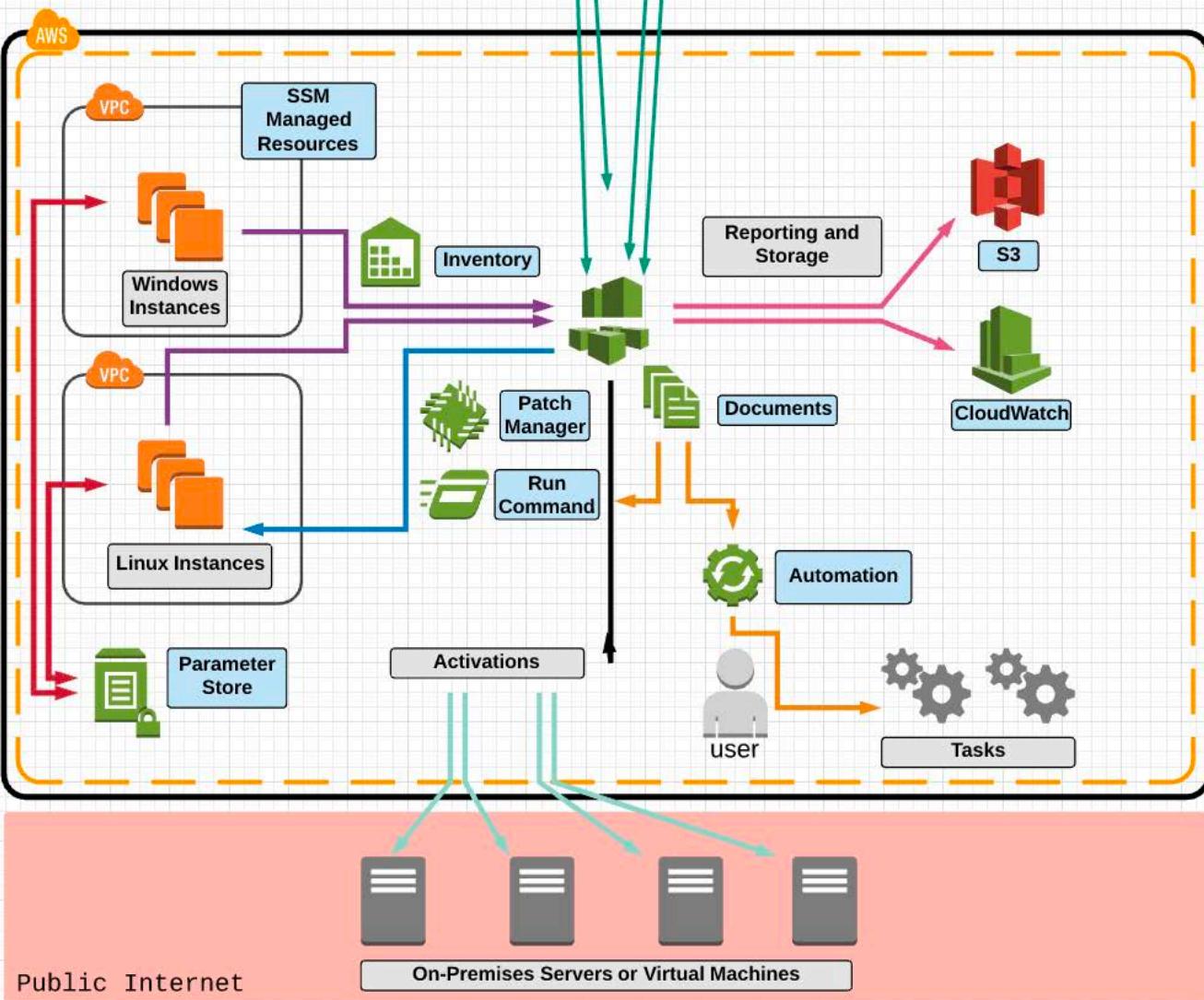
SSM Access Methods



PowerShell



CLI



A **managed instance** is an EC2 instance or on-premises servers or virtual machines configured to use Systems Manager. An EC2 instance or on-premises server becomes a managed instance by installing the SSM Agent and either assigning IAM permissions and an activation code if the compute resource is not an EC2 instance.





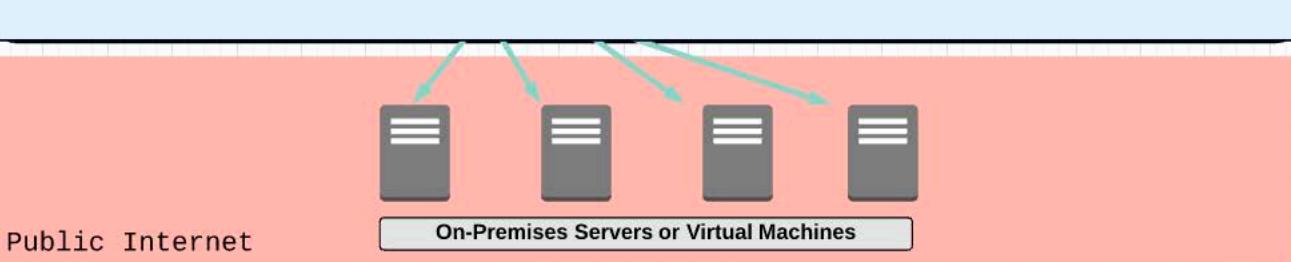
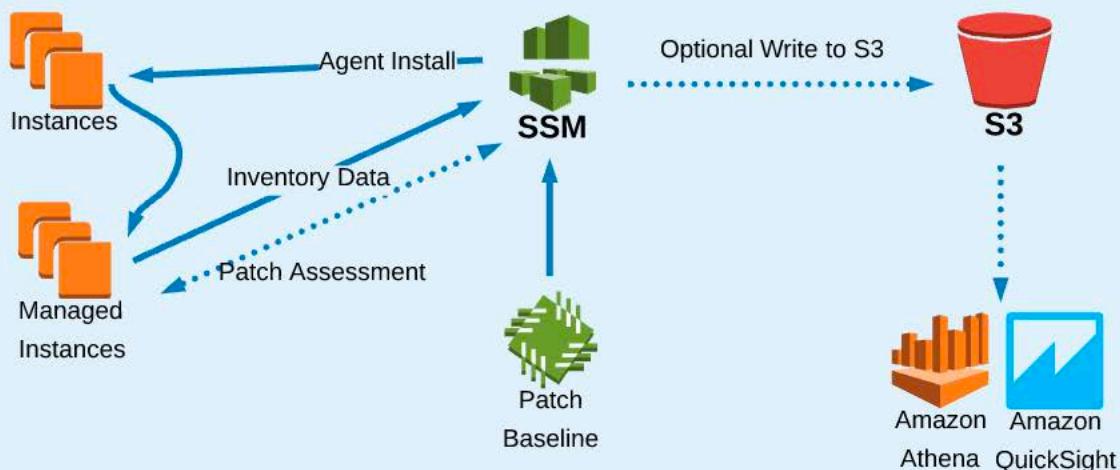
Insights (Inventory and Compliance)



Systems Manager offers two insight features: inventory and compliance. Both are supported by SSM State Manager.

Inventory periodically scans EC2 instances or on-premises servers/VMs, retrieving details of installed applications, AWS components, network config, Windows updates, detailed information on an instance/VM, details on running services, Windows roles, and optional custom data SSM can collect on your behalf.

Compliance allows that data to be compared against a baseline, providing a compliant or non-compliant state to a resource. Compliance uses State Manager, SSM patching, and custom compliance types.

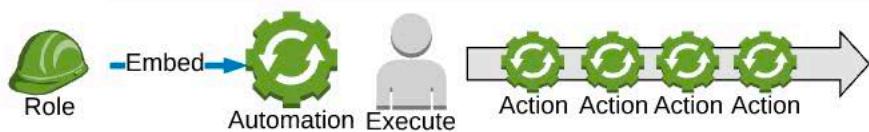




Actions

Actions are the operational engine part of Systems Manager. Actions is the part of Systems Manager that performs collections, runs commands, controls patching, and manages the general state of *managed instances*.

Automation



Run Command



Patch Manager



State Manager is a desired state engine. You define the "desired state" in the form of a Systems Manager document, which can be a command document or a policy document. A command document is used by Run Command and State Manager, while a policy document defines desired states and is only used by State Manager. A document is associated with one or more managed instances.

Public Internet

On-Premises Servers or Virtual Machines





Back

Global Infrastructure

Account & Services

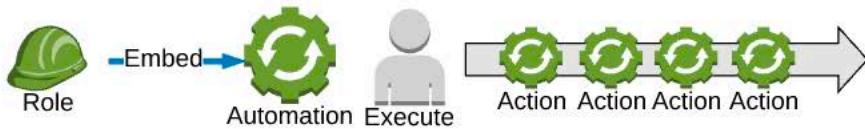
Networking

Well-Architected
Appendix

Actions

Actions are the operational engine part of Systems Manager. Actions is the part of Systems Manager that performs collections, runs commands, controls patching, and manages the general state of *managed instances*.

Automation



Run Command



Patch Manager

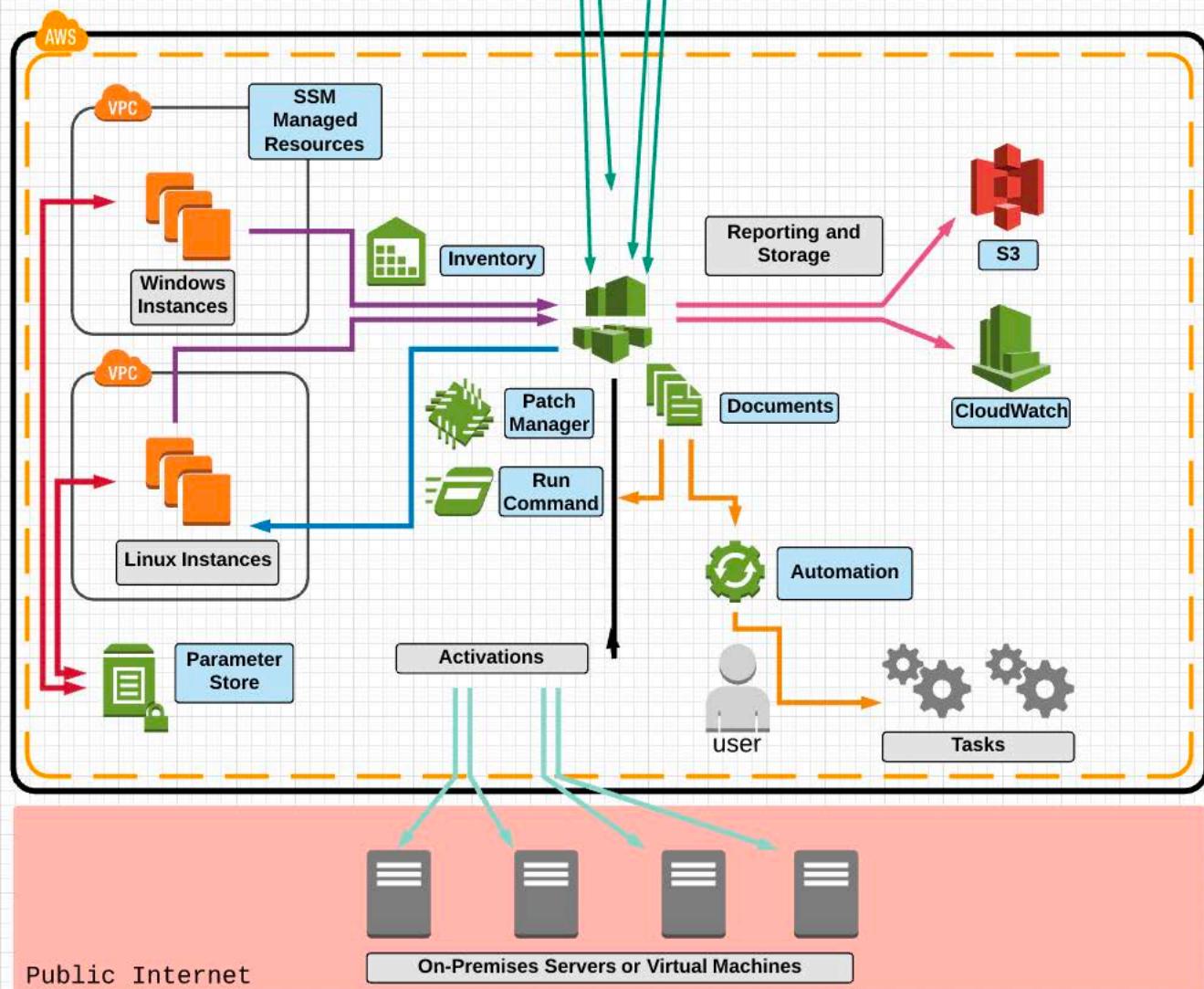
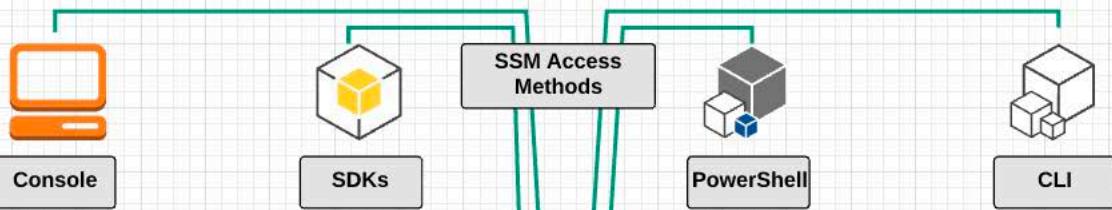


State Manager is a desired state engine. You define the "desired state" in the form of a Systems Manager document, which can be a command document or a policy document. A command document is used by Run Command and State Manager, while a policy document defines desired states and is only used by State Manager. A document is associated with one or more managed instances.



Public Internet

On-Premises Servers or Virtual Machines



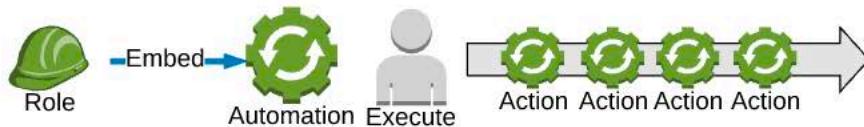
Systems Manager Documents define actions that Systems Manager performs and can be used by State Manager, Run Command, and Automation. They are controlling directives and can be written in YAML or JSON. X



Actions

Actions are the operational engine part of Systems Manager. Actions is the part of Systems Manager that performs collections, runs commands, controls patching, and manages the general state of *managed instances*.

Automation



Run Command



Patch Manager



State Manager is a desired state engine. You define the "desired state" in the form of a Systems Manager document, which can be a command document or a policy document. A command document is used by Run Command and State Manager, while a policy document defines desired states and is only used by State Manager. A document is associated with one or more managed instances.



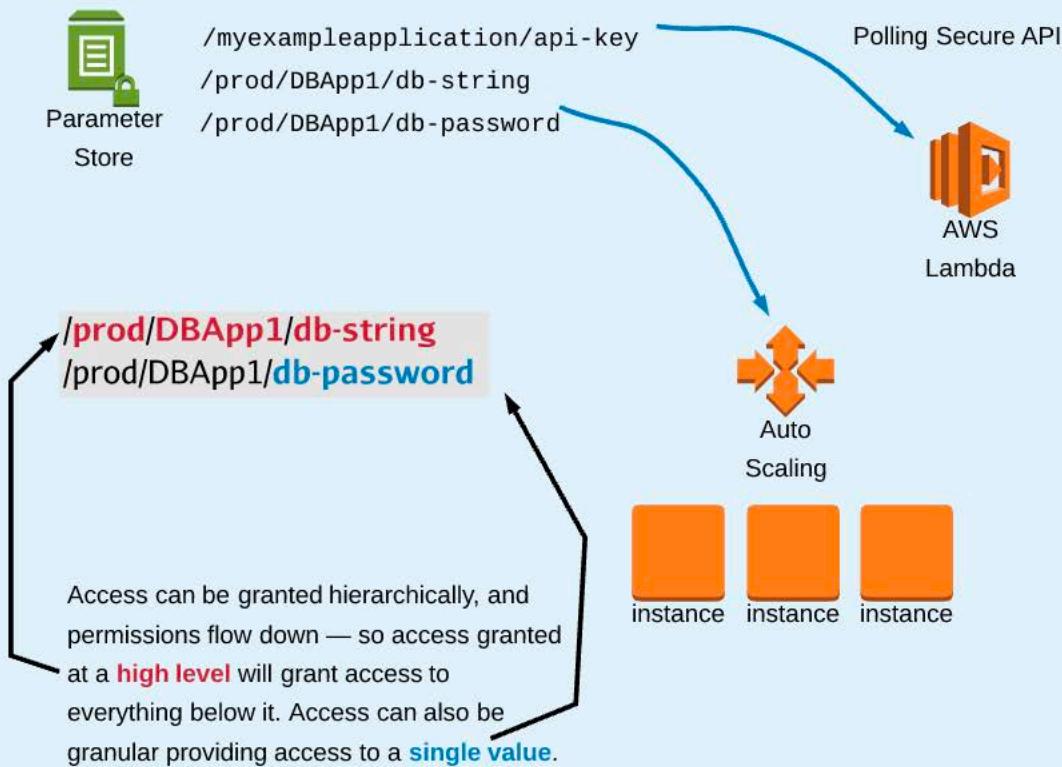


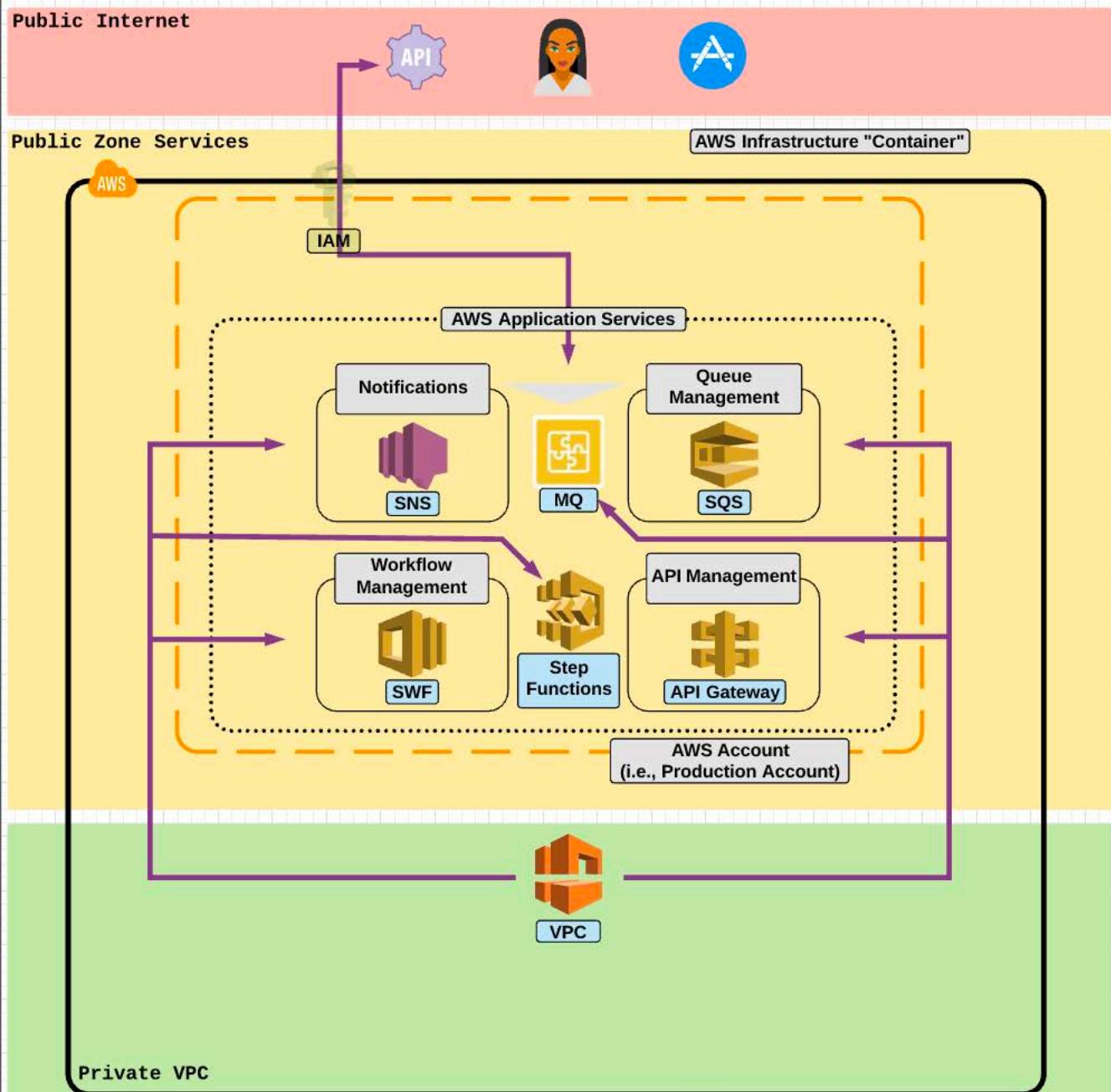
AWS Systems Manager Parameter Store

AWS Parameter Store provides secure storage for configuration data and secrets. Values can be stored as plaintext or as encrypted data using KMS. Data can be referenced using a unique name, providing you have permissions to access the data.

Key Features:

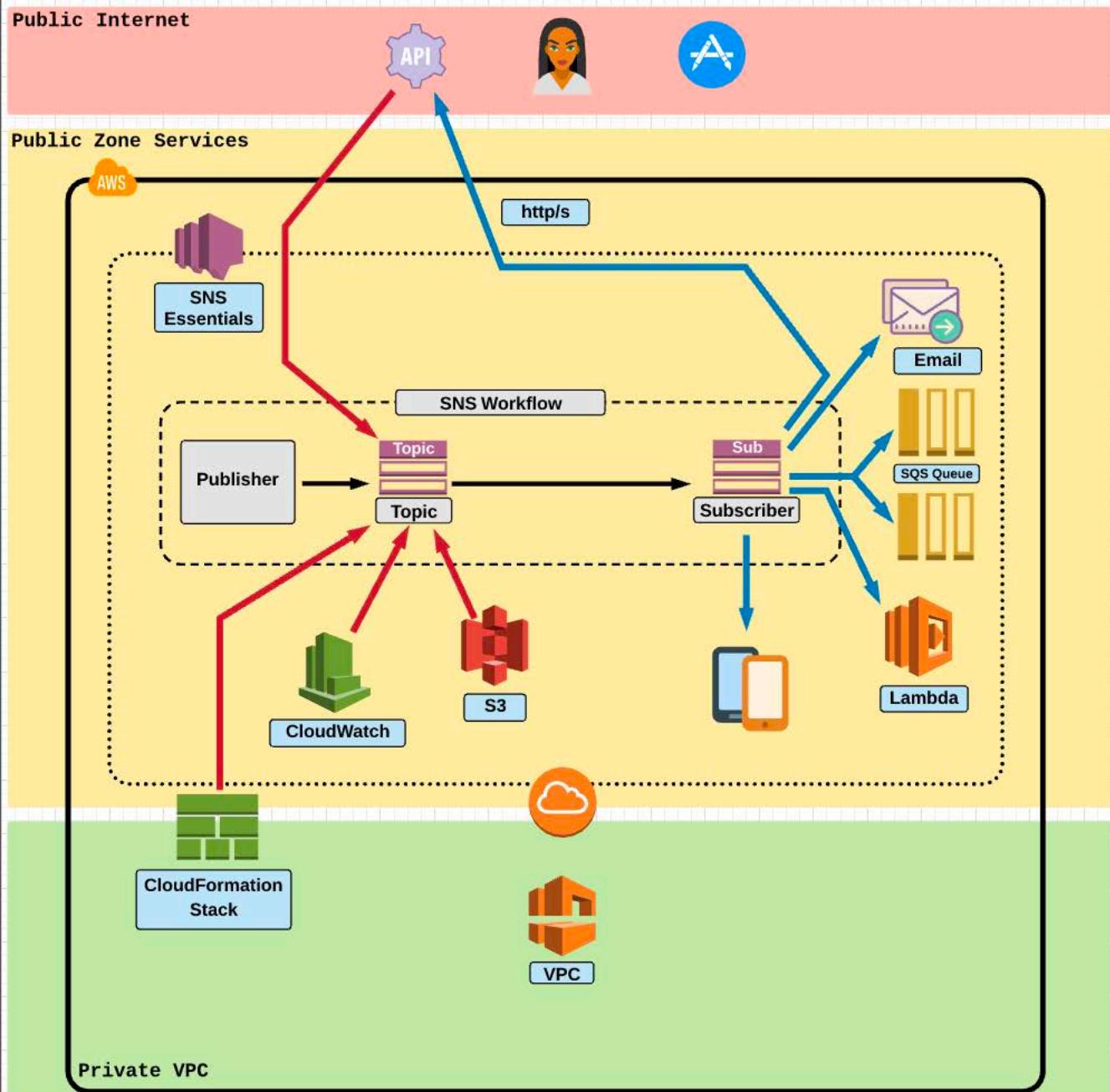
- Configuration and data is separated from code — no chance of leakage via Git
- Data is stored hierarchically, which aids management
- Data is versioned, and access can be controlled and audited
- Parameter Store integrates with many AWS services, including EC2, ECS, Lambda, CodeBuild/Deploy, and many more
- Can also be used for automated deployment using CloudFormation
- Serverless, resilient, and scalable





Application integration services act as supporting functionality to other applications, products, and systems. They generally allow the use of a certain architecture or offload certain parts of application functionality.

Application integration services are generally public zone services, meaning they can be accessed from within private AWS networks or from remote locations over the public internet.





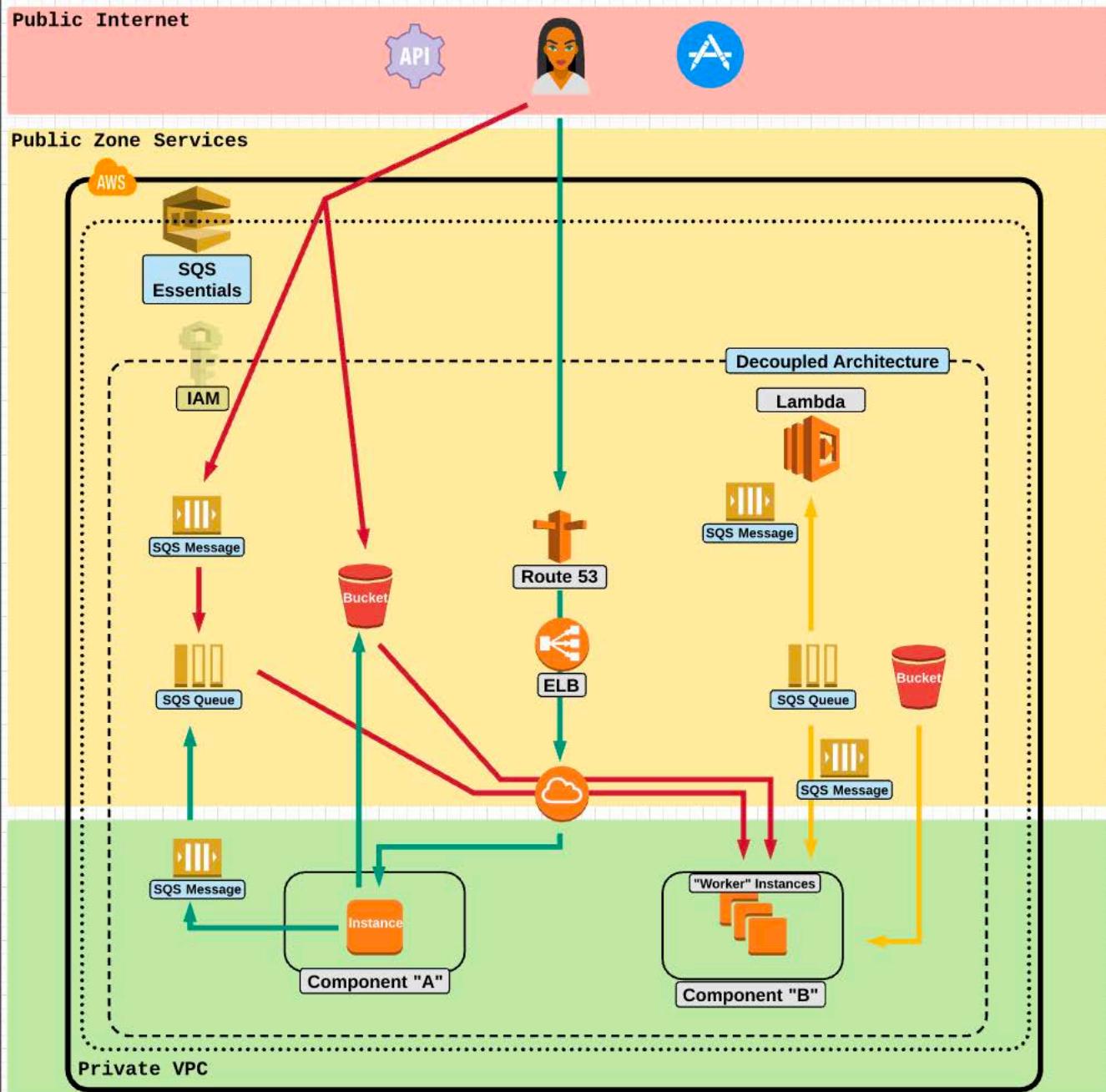
Application Services

SNS Essentials

- SNS coordinates and manages the sending and delivery of messages to subscribers.
- We are able to use SNS to receive notifications when events occur in our AWS environment.
- SNS is integrated into many AWS services, so it is easy to set up notifications based on events that occur in those services.
- With CloudWatch and SNS, a full-environment monitoring solution can be created that notifies administrators of alerts, capacity issues, downtime, changes in the environment, and more.
- This service can also be used for mobile push messages.

SNS Components:

- Topic:
 - Object to which you publish your message (<= 256 KB)
 - Subscribers subscribe to topic to receive message.
- Subscriber:
 - An endpoint to which a message is sent. Messages are simultaneously pushed to subscribers.
 - Available endpoints include:
 - HTTP
 - HTTPS
 - Email
 - Email-JSON
 - SQS
 - Application, mobile app notifications (iOS/Android/Amazon/Microsoft)
 - Lambda
 - SMS (cellular text message)
- Publisher:
 - The entity that triggers the sending of a message
 - Examples include:
 - Application
 - S3 event
 - CloudWatch alarm





Application Services



Amazon MQ Essentials

- An open-source message broker
- Managed Apache ActiveMQ
- Compatible with JMS API or protocols such as AMQP, MQTT, OpenWire, and STOMP
- Can function as a queue or topic
- One-to-one or one-to-many

MQ Components:

- Broker:
 - Single instance or
 - Highly available pair (active/standby)
 - Managed updates
- ActiveMQ Web Console:

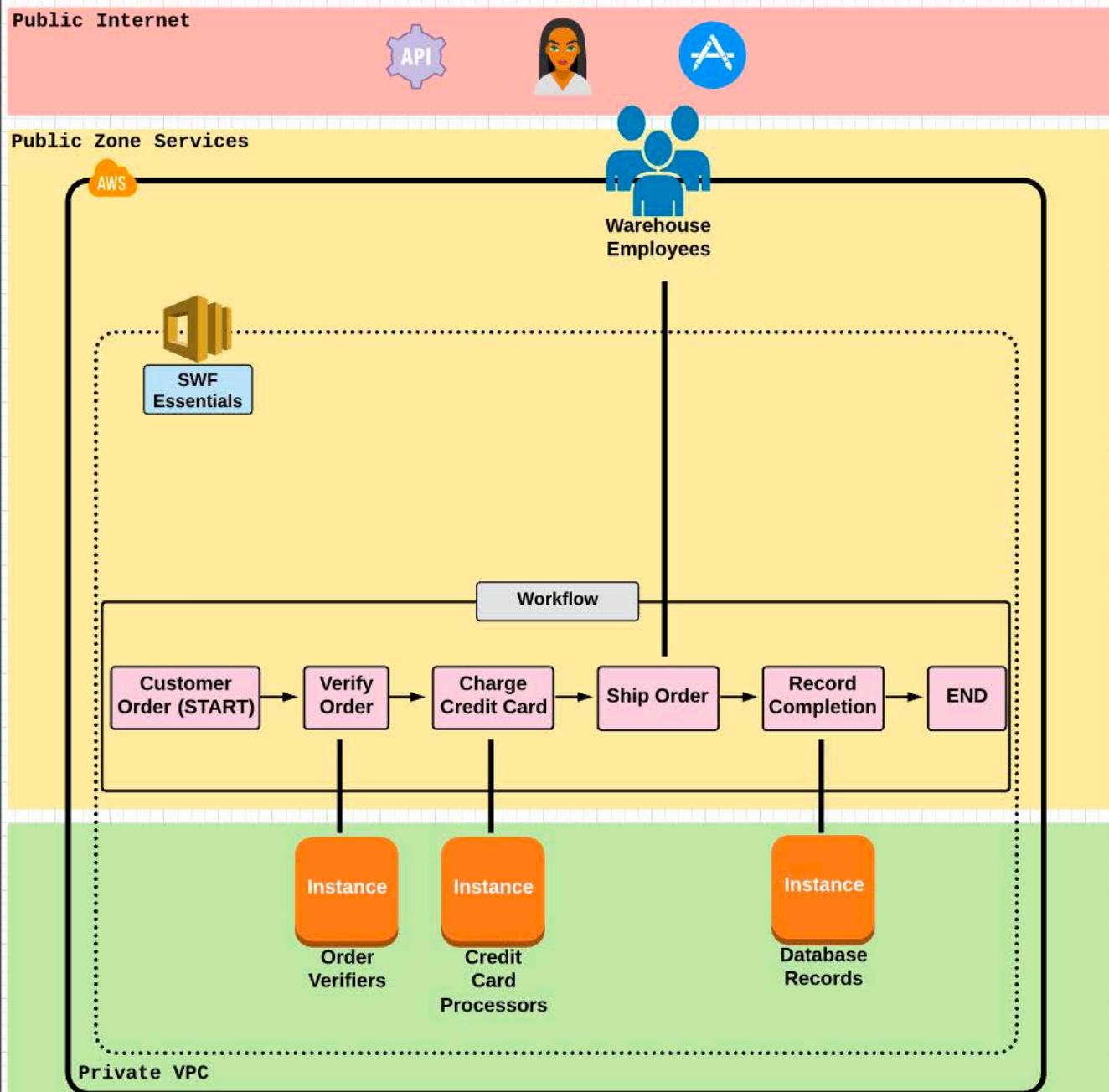
The screenshot shows the Apache ActiveMQ Web Console interface. At the top, there's a navigation bar with links for Home, Queues, Topics, Subscribers, Connections, Network, Scheduled, and Send. On the right side, there's a sidebar with sections for Queue Views (Graph, XML), Topic Views (XML), Subscribers Views (XML), and Useful Links (Documentation, FAQ, Release, Forum). The main content area is titled "Broker" and displays the following configuration details:

Name	localhost
Version	5.14.0
ID	20name-of-your-computer.local-56222-1468240159157-0-1
Uptime	2 minutes
Store percent used	0
Memory percent used	0
Tmps percent used	0

At the bottom of the page, there's a copyright notice: "Copyright 2000-2015 The Apache Software Foundation." Below the browser window, there's a green box labeled "Private VPC" and a blue button labeled "VPC".

Application integration services act as supporting functionality to other applications, products, and systems. They generally allow the use of a certain architecture or offload certain parts of application functionality.

Application integration services are generally public zone services, meaning they can be accessed from within private AWS networks or from remote locations over the public internet.





Application Services

Simple Workflow Essentials

- SWF is a fully managed “workflow” orchestration service provided by AWS.
- A SWF **workflow** allows an architect/developer to implement distributed, asynchronous applications as a workflow.
- A **workflow** coordinates and manages the execution of activities that can be run asynchronously across multiple computing devices.
- SWF has consistent execution.
- Guarantees the order in which tasks are executed.
- There are no duplicate tasks.
- The SWF service is primarily an API into which an application can integrate its workflow service. This allows the service to be used by non-AWS services, such as an on-premises data center.
- A workflow execution can last up to one year.

Components of SWF

- **Starter:** Starts the workflow
- **Workflow:** A sequence of steps required to perform a specific task
 - A workflow is also commonly referred to as a **decider**.
- **Activities:** A single step (or unit of work) in the workflow
- **Tasks:** What interacts with the “workers” that are part of a workflow
 - Activity task: Tells the worker to perform a function
 - Decision task: Tells the decider the state of the workflow execution, which allows the decider to determine the next activity to be performed
- **Worker:** Responsible for receiving a task and taking action on it
 - Can be any type of component, such as an EC2 instance or even a person

AWS Flow Framework for Java

- Simplifies working with SWF by providing objects and classes



Public Internet



Public Zone Services

AWS

State Machine

Start



Lambda

State1



Lambda

State2



Lambda

State3a



Fail

State3b



Succeed

End



Lambda

SES



API GW

JSON State Language



EC2

Private VPC

Step Functions is similar to Simple Workflow. Simple Workflow needs infrastructure — Step Functions doesn't. Step Functions are serverless.

State machines consist of seven types of "state": **Task**, **Choice**, **Parallel**, **Wait**, **Fail**, **Succeed**, and **Pass**. Task states perform the work; Choice adds branching based on input; Parallel splits into multiple or joins data from multiple streams; Wait adds a delay; Fail fails execution; Succeed finishes with a success status; and Pass passes its input to its output.

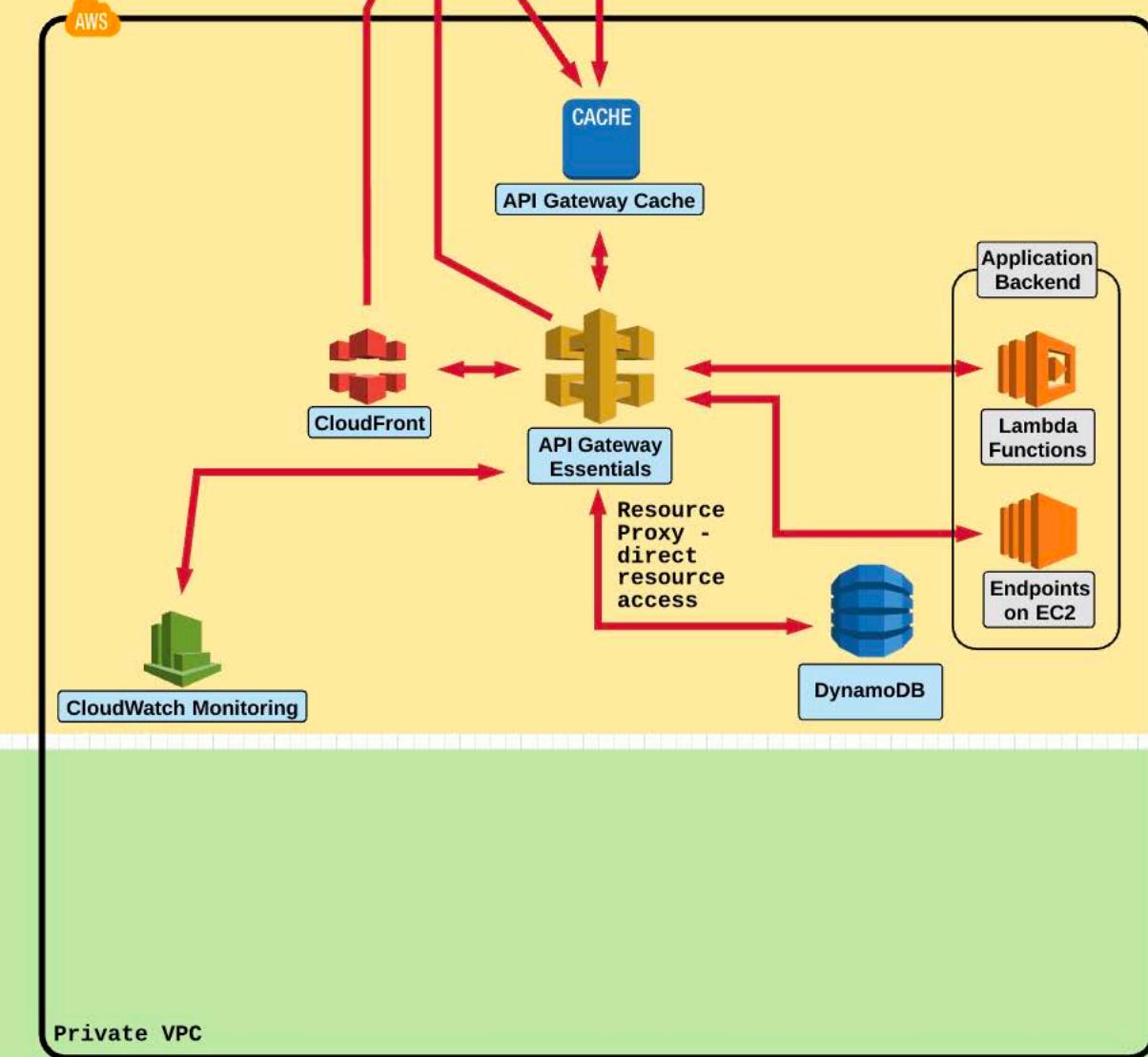
All of the state types are used to construct serverless workflow-style processes.



Public Internet



Public Zone Services

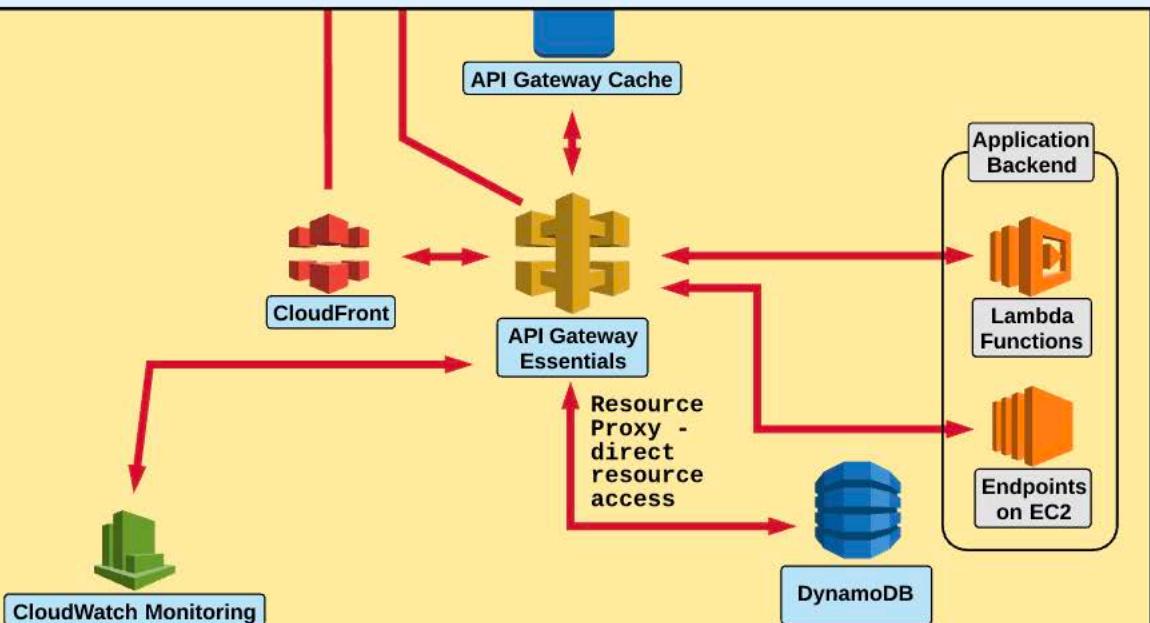




Application Services

API Gateway Caching

- API Gateway will cache API responses so that duplicate API requests do not have to hit your backend.
 - This reduces load on your backend and speeds up calls to your backend.
- You can configure a cache key and time to live (TTL) of the API response.
- Caching can be set up on a per-API or per-stage basis.

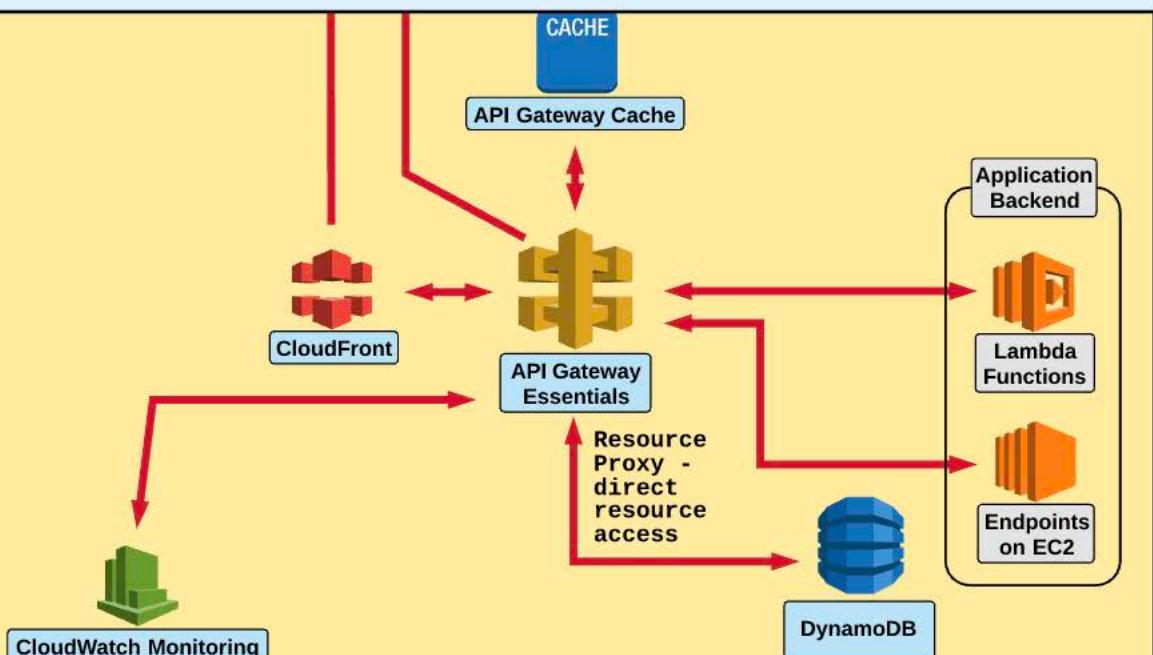




Application Services

API Gateway: CloudFront

- API Gateway benefits from using CloudFront infrastructure:
 - Built-in DDoS attack protection and mitigation
 - All CloudFront Edge Locations become entry points for your API into your backend.
- Summary: Benefits are reduced latency and improved projection.





Application Services

API Gateway Essentials

- API Gateway is a fully managed service that allows you to create and manage your own APIs for your application.
- API Gateway acts as a "front door" for your application, allowing access to data/logic/functionality from your backend services.

API Gateway Main Features

- Build RESTful APIs with:
 - Resources
 - Methods (e.g., GET, POST, PUT)
 - Settings
- Deploy APIs to a "stage" (different environments — e.g., dev, beta, production).
 - Each stage can have its own throttling, caching metering, and logging.
- Create a new API version by cloning an existing one.
 - You can create and work on multiple versions of an API (API version control).
- Roll back to previous API deployments.
 - A history of API deployments are kept.
- Custom domain names
 - Custom domain names can point to an API or stage.
- Create and manage API keys for access *and* meter usage of the API keys through Amazon CloudWatch Logs.
- Set throttling rules based on the number of requests per second (for each HTTP method).
 - Requests over the limit would be throttled (HTTP 429 response).
- Security using Signature Version 4 to sign and authorize API calls
 - Temporary credentials generated through Amazon Cognito and Security Token Service (STS)

Benefits of API Gateway

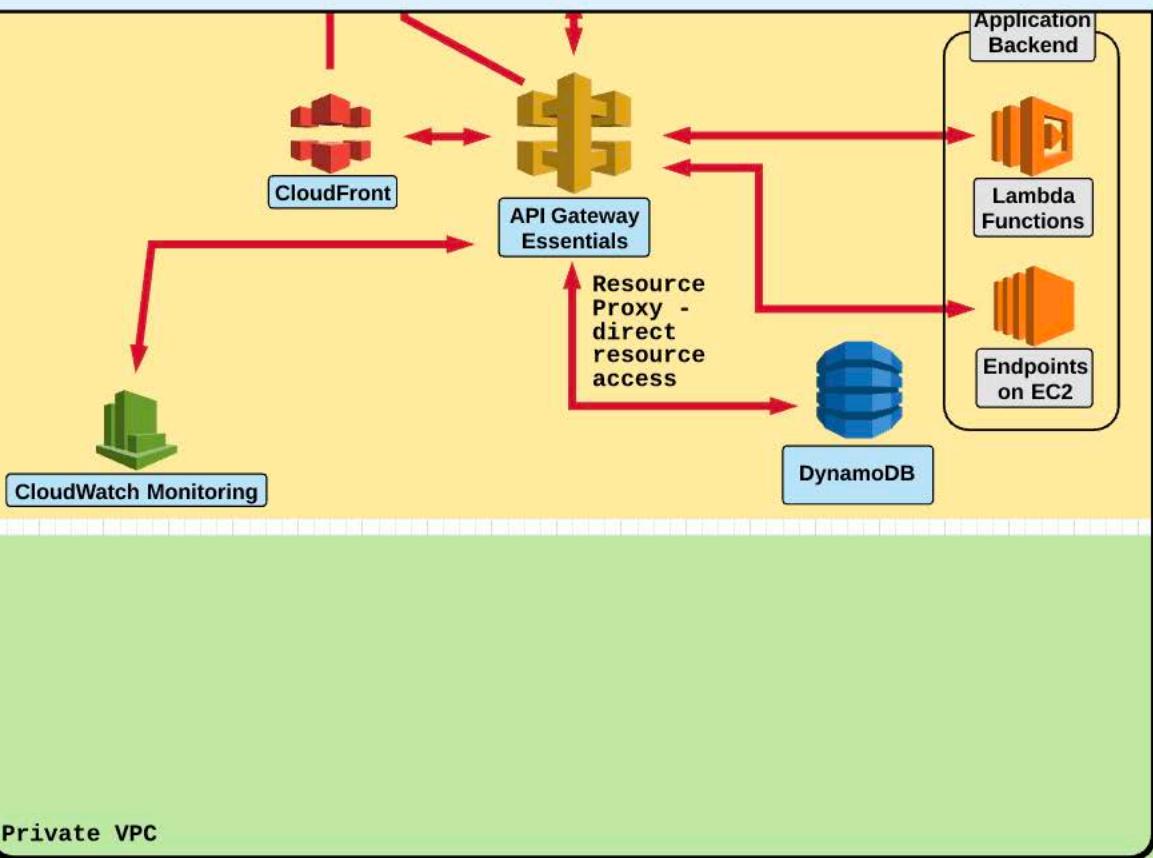
- Ability to cache API responses
- DDoS protection via CloudFront
- SDK generation for iOS, Android, and JavaScript
- Supports Swagger (a very popular framework of API dev tools)
- Request/response data transformation (e.g., JSON IN to XML OUT)



Application Services

API Gateway: CloudWatch

- CloudWatch can be used to monitor API Gateway activity and usage.
- Monitoring can be done on the API or stage level.
- Throttling rules are monitored by CloudWatch
- Monitoring metrics include such statistics as:
 - Caching
 - Latency
 - Detected errors
- Method-level metrics can be monitored.
- You can create CloudWatch alarms based on these metrics.





AWS CSA Professional Blueprint

Overview



AWS provides a blueprint for each of their certification exams. You can use this to guide your studies and focus on certain services and concepts within AWS.

Not all services and concepts are tested equally. There are different weights that add up to a total of 100%.

Below is the breakdown for each domain (five total) on the current exam:

DOMAIN NUMBER	DOMAINS	WEIGHT PERCENTAGE
1	Design for Organizational Complexity	12.5
2	Design for New Solutions	31
3	Migration Planning	15
4	Cost Control	12.5
5	Continuous Improvement for Existing Solutions	29
TOTAL		100

Check out the full blueprint breakdown here: [AWS CSAP Blueprint](#)

Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

The Well-Architected Framework is a series of best-practice recommendations and questions to ask when designing and developing cloud architectures.

The framework consists of the following five pillars:

**Operational Excellence**

Run and monitor systems to provide business value.

Security

Protect and monitor systems.

Reliability

Recover from failure and mitigate disruptions.

Performance Efficiency

Use computing resources efficiently.

Cost Optimization

Avoid or eliminate unnecessary expenses.

Select a resource from above to explore individually or click Services to move on.

[Services](#)



Overview

Operational Excellence

Security

Reliability

Performance Efficiency

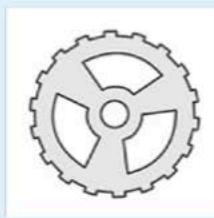
Cost Optimization

AWS Well-Architected Framework

Operational Excellence

"The ability to run and monitor systems to deliver business value and to continually improve supporting processes and procedures."

AWS lays out the following six design principles for operational excellence:



1. **Perform operations as code.**
2. **Annotate documentation.**
3. **Make frequent, small, reversible changes.**
4. **Refine operations procedures often.**
5. **Anticipate failure.**
6. **Learn from all operational failures.**

AWS breaks down cloud-based operational excellence into three areas:

1. **Prepare** Understand your workloads and expected behaviors.
2. **Operate** Measure success with business achievements and customer outcomes.
3. **Evolve** Maintain a continuous lifecycle of improvement over time.

"Make me excellent!"



Check out the operational excellence pillar documentation: [OE Pillar](#)

Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

Operational Excellence

Prepare

You need to be able to understand your workloads and what kinds of behaviors to expect.

There are three areas to consider:



- Operation Priorities
- Design for Operations
- Operational Readiness

Operational Priorities

Understand your workload, roles, and business goals.

Design for Operations

Implement engineering practices to allow for quick, safe fixes.

Operational Readiness

Ensure you have consistent processes and deployments.

Check out the Prepare component in depth here: [Whitepaper](#)



Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

Operational Excellence

Prepare

You need to be able to understand your workloads and what kinds of behaviors to expect.



Operational Priorities

Design for Operations

Operational Readiness

There are three areas to consider:

Utilize AWS services to educate teams involved in your business goals and requirements.

AWS provides the following resources for you:

- [AWS Support Center](#)
- [AWS Developer Forums](#)
- [AWS Knowledge Center](#)

AWS also recommends and provides these services:

- [AWS Cloud Compliance](#)
- [AWS Trusted Advisor](#)
- [Business Support](#)
- [Enterprise Support](#)



Trusted Advisor

Ensure you have consistent processes and deployments.

Check out the Prepare component in depth here: [Whitepaper](#)



Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

Operational Excellence

Prepare

You need to be able to understand your workloads and what kinds of behaviors to expect.



There are three areas to consider:

- Operation Priorities
- Design for Operations
- Operational Readiness

Operational Priorities

Design for Operations

Operational Readiness

Your workload design should include the following:

- ▶ How you will deploy it
- ▶ How you will update it
- ▶ How you will keep it operating

AWS suggests that you treat your workload as code.

→ AKA Infrastructure as Code! ←

They also suggest setting up CI/CD deployment pipelines.
That last major recommendation is to enable custom metric logging.



CloudFormation

AWS Key Services:

- ▶ [AWS CloudFormation](#)
- ▶ [AWS Developer Tools](#)
 - ▶ [AWS X-Ray](#)
- ▶ [Amazon CloudWatch](#)



X-Ray

Check out the Prepare component in depth here: [Whitepaper](#)



Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

Operational Excellence

Prepare

You need to be able to understand your workloads and what kinds of behaviors to expect.



There are three areas to consider:

- Operation Priorities
- Design for Operations
- Operational Readiness

Operational Priorities

Design for Operations

Operational Readiness

Check ou

Use consistent processes to mark when you are able to go live with your workload!

Use some type of checklist.



Script your deployments and utilize parallel environments.



Config



Systems Manager

AWS Key Services

▶ [AWS Config](#)

▶ [Amazon EC2 Systems Manager](#)

▶ [AWS Lambda](#)



Lambda

Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

Operational Excellence

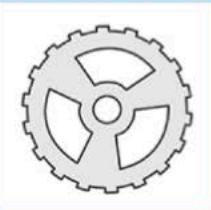
Operate

Success in this area is "measured by the outcomes and metrics you define."

You need to consider two main concepts:

- Understanding Operational Health
- Responding to Events

Grasping both of these points leads to quicker identification of impacts to your workload and faster response times.



Operational Health

Understand your workload's health easily!

Respond to Events

Anticipate events — both planned and unplanned.

Check out the Operate component in depth here: [White Paper](#)

Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

Operational Excellence

Operate

Success in this area is "measured by the outcomes and metrics you define."

You need to consider two main concepts:

- Understanding Operational Health
- Responding to Events

X

Grasping both of these points

Understand your workload!

Use the appropriate metrics for your needs and requirements.

AWS Example: Send CloudWatch logs to Elasticsearch, and use Kibana to visualize your operational health.

AWS Key Services

- ▶ [Amazon CloudWatch Logs](#)
 - ▶ [Amazon ES](#)
- ▶ [Personal Health Dashboard](#)
- ▶ [Service Health Dashboard](#)



Operational Health

Respond to Events

Anticipate events — both planned and unplanned.

Check out the Operate component in depth here: [White Paper](#)

Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

Operational Excellence

Operate

Success in this area is "measured by the outcomes and metrics you define."

You need to consider two main concepts:

- Understanding Operational Health

X

Grasping both of these points

Operational Health

Respond to Events

Design your workloads with events in mind!

Planned events: site promotions, advertising traffic, or DR tests.

Unplanned events: AZ losses and hardware issues.

Isolate your resources to determine how business will be impacted if any of them go down.

Do your best to automate processes and fixes!



Key AWS Services

- [Amazon CloudWatch](#)
- [CloudWatch Events](#)
- [Amazon SNS](#)
- [Auto Scaling](#)
- [EC2 Systems Manager](#)
- [AWS Lambda](#)



Check out the Operate component in depth here: [White Paper](#)



Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

Operational Excellence

Evolve



AWS defines the evolution component as a "continuous cycle of improvement over time."

You should always try to improve your workloads and architectures by making small, incremental changes. Don't make a few massive changes.

AWS lays out the following two principles:

- Learning from Experience
- Sharing Learnings

Having a firm grasp of these topics will help your operations evolve smoothly over time!

Learning from Experience

Learn from mistakes and failures!

Sharing Learnings

Always be sure to participate in knowledge sharing.

Check out the Evolve component in depth here: [Whitepaper](#)



Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

Operational Excellence

Evolve



AWS defines the evolution component as a "continuous cycle of improvement over time".

You should always try to implement incremental changes.

AWS recommends:

Having a firm grasp of these tools:

Learning from Experience

Sharing Learnings

Learn from all of your failures because they *will* happen.
Analyze failures, and then work on improvements.

AWS recommends using mirrored, temporary environments for testing deployments, etc.

Use *all* logging services, such as CloudTrail and CloudWatch.

Lastly, AWS suggests you perform cross-team reviews that cover *all* aspects of the business.



AWS Key Services

- ▶ [Amazon QuickSight](#)
- ▶ [Amazon Athena](#)
- ▶ [Amazon S3](#)
- ▶ [Amazon CloudWatch](#)
- ▶ [Amazon ES](#)



CloudWatch

Check out the Evolve component in depth here: [Whitepaper](#)

Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

Operational Excellence

Evolve



AWS defines the evolution component as a "continuous cycle of improvement over time."

You should always try to improve your workloads and architectures by making small incremental changes. Don't make a few massive changes.

X

AWS lay

All knowledge learned should be shared!

Knowledge sharing helps you avoid future challenges that may affect your workloads.

Having a firm grasp of these to

Once again, AWS encourages you to treat infrastructure as code. This allows for easier sharing.

Learning from Experience



Sharing Learnings

Remember the least privilege principle!

AWS Key Services



- ▶ [IAM](#)
- ▶ [Amazon SNS](#)
- ▶ [AWS CodeCommit](#)
- ▶ [AWS Lambda](#)
- ▶ [AMIs](#)



CodeCommit

Check out the

Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

Security

"The ability to protect information, systems, and assets while delivering business value through risk assessments and mitigation strategies."

The security pillar involves protecting all information and assets while ensuring business risks are low and mitigated rapidly.

AWS recommends the following guidelines for maintaining strong security:



- 1 Implement a strong identity foundation.
- 2 Enable traceability.
- 3 Apply security at every layer.
- 4 Automate security.
- 5 Protect data in transit and at rest.
- 6 Prepare for security events.

Review the Shared Security Model documentation: [LINK](#)

Shared Responsibility Model

AWS breaks down security into five areas:

IAM

Detective Controls

Infrastructure Protection

Data Protection

Incident Response

Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

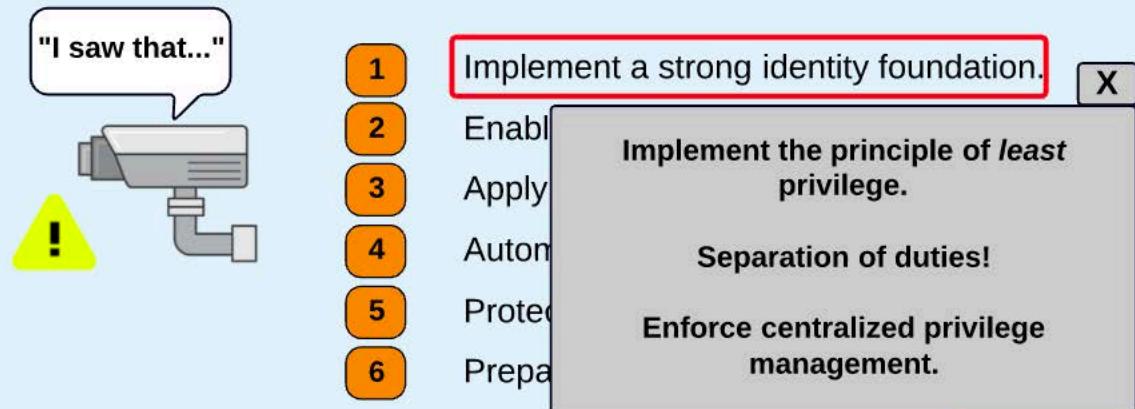
AWS Well-Architected Framework

Security

"The ability to protect information, systems, and assets while delivering business value through risk assessments and mitigation strategies."

The security pillar involves protecting all information and assets while ensuring business risks are low and mitigated rapidly.

AWS recommends the following guidelines for maintaining strong security:



Review the Shared Security Model documentation: [LINK](#)

Shared Responsibility Model

AWS breaks down security into five areas:

IAM

Detective Controls

Infrastructure Protection

Data Protection

Incident Response

Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

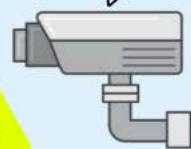
Security

"The ability to protect information, systems, and assets while delivering business value through risk assessments and mitigation strategies."

The security pillar involves protecting all information and assets while ensuring business risks are low and mitigated rapidly.

AWS recommends the following guidelines for maintaining strong security:

"I saw that..."



1

Implement a strong identity foundation.

2

Enable traceability.

3

Apply security at every

4

Automate security.

5

Protect data in transit a

6

Prepare for security eve

X

Set up monitoring, alerts, and auditing for any changes or activities within your environments!

In addition, heavily leverage logging and metrics services.

Review the Shared Security Model documentation: [LINK](#)

Shared Responsibility Model

AWS breaks down security into five areas:

IAM

Detective Controls

Infrastructure Protection

Data Protection

Incident Response

Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

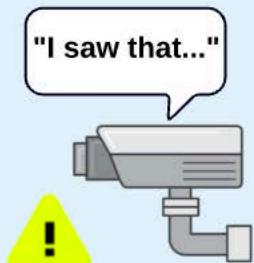
AWS Well-Architected Framework

Security

"The ability to protect information, systems, and assets while delivering business value through risk assessments and mitigation strategies."

The security pillar involves protecting all information and assets while ensuring business risks are low and mitigated rapidly.

AWS recommends the following guidelines for maintaining strong security:



- 1
- 2
- 3
- 4
- 5
- 6

Implement a strong identity foundation.

Enable traceability.

Apply security at every layer.



You should take a layered defense (i.e., "defense in depth") approach to security.

No service or resource should be exempt from this practice.

Shared Responsibility Model

Review the Shared Security

AWS breaks down security into five areas.

IAM**Detective Controls****Infrastructure Protection****Data Protection****Incident Response**

Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

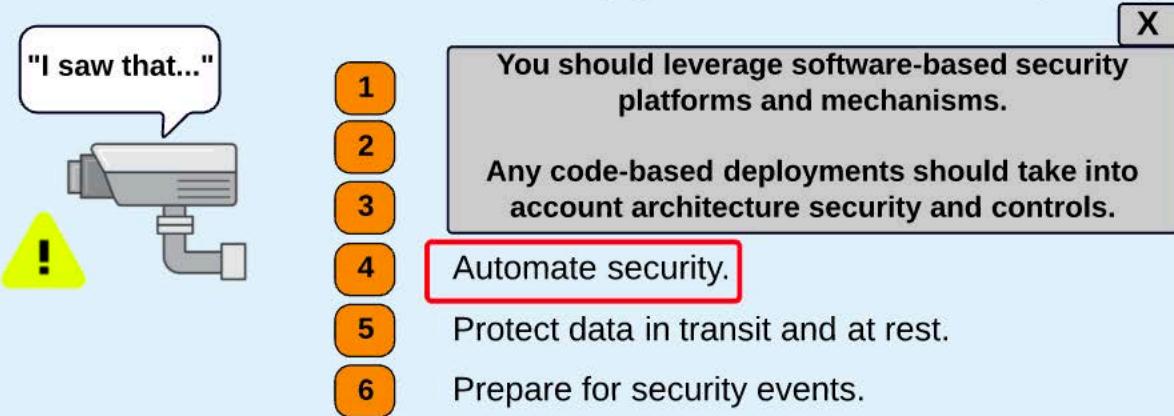
AWS Well-Architected Framework

Security

"The ability to protect information, systems, and assets while delivering business value through risk assessments and mitigation strategies."

The security pillar involves protecting all information and assets while ensuring business risks are low and mitigated rapidly.

AWS recommends the following guidelines for maintaining strong security:



Review the Shared Security Model documentation: [LINK](#)

Shared Responsibility Model

AWS breaks down security into five areas:

IAM

Detective Controls

Infrastructure Protection

Data Protection

Incident Response

Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

Security

"The ability to protect information, systems, and assets while delivering business value through risk assessments and mitigation strategies."

The security pillar involves protecting all information and assets while ensuring business risks are low and mitigated rapidly.

AWS recommends the following guidelines for maintaining strong security:

"I saw that..."



1

You should classify and categorize data by sensitivity.

2

After classification, you should use encryption, tokenization, etc., where necessary.

3

Reduce human access to data streams.

4

Protect data in transit and at rest.

5

Prepare for security events.

6

X

Review the Shared Security Model documentation: [LINK](#)

Shared Responsibility Model

AWS breaks down security into five areas:

IAM

Detective Controls

Infrastructure Protection

Data Protection

Incident Response

Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

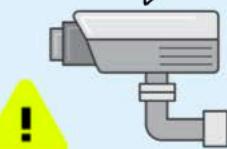
Security

"The ability to protect information, systems, and assets while delivering business value through risk assessments and mitigation strategies."

The security pillar involves protecting all information and assets while ensuring business risks are low and mitigated rapidly.

AWS recommends the following guidelines for maintaining strong security:

"I saw that..."



1

Implement a strong identity foundation.

2

Enable traceability.

X

3

Have an incident management process set up!

4

Test for situations that may occur, such as DR or application failures.

5

Prepare for security events.

6

Review the Shared Security Model documentation: [LINK](#)

Shared Responsibility Model

AWS breaks down security into five areas:

IAM

Detective Controls

Infrastructure Protection

Data Protection

Incident Response

Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

Security

Identity and Access Management (IAM)



IAM is one of the most critical components of AWS!

You control *all* access and authentication to your resources this way.

The AWS documentation discusses best practices for implementing and using IAM in the following sections:

- Protecting AWS Credentials
- Fine-Grained Authorization

Protecting AWS Credentials

You want to protect all credentials within your account.

Follow best practices such as rotating access keys, enforcing password policies, and requiring MFA.



Fine-Grained Authorization

Always follow the principle of least privilege.

Specifically define roles that are required for users and applications.

Use AWS Organizations as a central account management point.



Organizations



Defined Needs

Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

Security

Detective Controls

You can use detective controls to identify potential threats.



Using these tools can help you meet compliance requirements.



You want to keep track of all resources and audit *all* changes.

The AWS docs lay out a few ways to accomplish this in the following sections:

- Capture and Analyze Logs
- Integrate Auditing Controls with Notification and Workflow

Capture and Analyze

AWS allows for easier asset management.

You can programmatically describe instances, monitor logs, etc.

You can use API-driven services to easily collect, filter, and analyze logs.



Auditing Controls

Logging is a *critical* tool for analyzing risks and correcting misbehaviors.

AWS best practice is to integrate security logging with notifications and event handling.



Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

Security

Infrastructure Protection

This entails all control methods needed to comply with your industry best practices and AWS guidelines.

You want to protect against unauthorized access and attacks.

AWS separates Infrastructure Protection into three parts:

- Protecting Network and Host-Level Boundaries
- System Security Configuration and Maintenance
- Enforcing Service-Level Protection

Protection

Design your topology carefully.

Consider public vs. private components.



VPC



Direct Connect

Config and Maintenance

Keep a strict security posture!

Always maintain least privilege principles.

Automate security procedures.



Inspector



Systems Manager



Automation

Enforcement

Protect service endpoints via IAM.

Least privilege principle!

IAM policies are critical here.



KMS



S3



Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

Security

Data Protection

Data and resource protection should be the foundation of your architecture's design.

AWS breaks down Data Protection into five categories:

- Data Classification
- Encryption/Tokenization
- Protecting Data at Rest
- Protecting Data in Transit
- Data Backup/DR

Data Classification

Classify data based on sensitivity.



Encryption

Encrypt data!

Scope out tokenization approaches.



Data at Rest

Any data persisting for any duration.

Encryption and access control!



Data in Transit

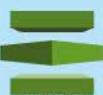
Any data moving from one system to another — *not just external*.



Encryption, VPNs, certificates.



ELB



ACM

Backups and DR

Define your backup, replication, and recovery approach!

Automate when you can.

Test your backups!



S3

Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

Security

Incident Response

All organizations should implement a response and mitigation plan.

Nobody is immune!

Teams should aim to be proactive, not reactive.

AWS provides one main approach for this:

- Clean Room
-



Clean Room

Teams should maintain situational awareness!

Tag your resources.



CloudFormation

If possible, form an incident response team.



Use APIs to automate your routine tasks.



Be able to identify root causes for incidents.

Step
Functions

Overview

Operational Excellence

Security

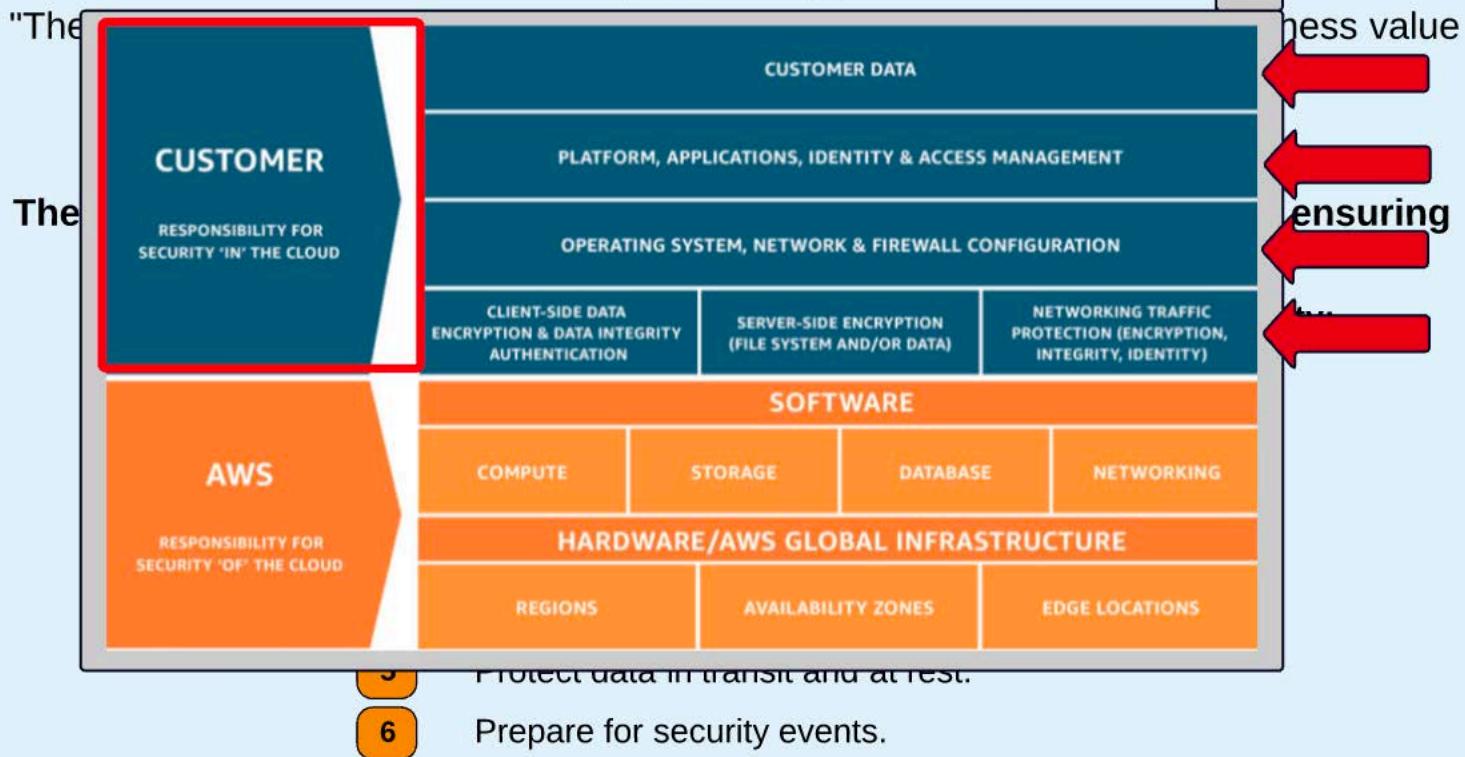
Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

Security



Review the Shared Security Model documentation: [LINK](#)

Shared Responsibility Model

AWS breaks down security into five areas:

IAM

Detective Controls

Infrastructure Protection

Data Protection

Incident Response



Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

Reliability



"The ability to recover from and mitigate service disruptions."

AWS stresses the importance of service availability within your architecture. You need to know how to calculate availability (five nines!) as well as maintain the most cost-efficient architectures.

The reliability pillar is split into a few key points:

Example

- Always test your recovery solutions.
- Design with self-healing in mind.
- Architect around the ability to scale dynamically.
- Focus on automating all deployments.
- Enable monitoring and alarming.
- Calculate your resource needs. Don't guess!
- Remember least privilege!

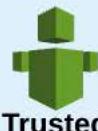
Key Services



IAM



VPC



Trusted Advisor



Shield



ALB



CloudTrail



CloudWatch



S3



CloudFormation

Check out the reliability pillar documentation: [Reliability Pillar](#)



Overview

Operational Excellence

Security

Reliability

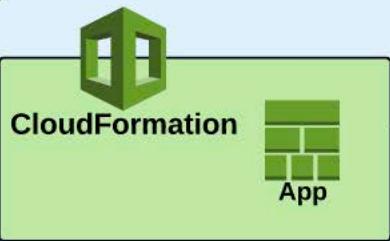
Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

AWS

BACK



Automated Deployment

Sample that meets the suggested design patterns.



ALB

**Auto Scaling Group****DB Tier**

AZ A

AZ B

Replication

AWS



Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

Performance Efficiency

"The ability to use computing resources efficiently to meet system requirements and to maintain that efficiency as demand changes and technologies evolve."



Don't reinvent the wheel! AWS can help you deploy efficient services.

You can reach users worldwide with a single click.

Use event-driven architectures when possible!

AWS defines four areas of performance efficiency:

- Selection
- Review
- Monitoring
- Trade-offs

Selection

Choose the right instance and storage options.

Review

Re-evaluate when AWS announces new features and services.

Monitoring

Verify that resources perform as expected.

Trade-Offs

Consider caching and read replicas.

Check out the performance efficiency pillar documentation: [Performance Efficiency](#)

Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

Performance Efficiency

Selection

Selection involves implementing the correct solution based on your specific workload needs.

There is no one-size-fits-all solution for architectures.

It's important to optimize compute resources for many reasons, including cost and performance. There are different instance types for a reason. Some are better suited than others for certain tasks.

Compute



Auto Scaling

Four areas of selection:

- Compute
- Storage
- Databases
- Network

Storage



S3



EBS

Databases



DynamoDB



RDS

Network



VPC



Route 53



Direct Connect

Read more about the selection component here: [Selection](#)

Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

Performance Efficiency

Review

Continuously reviewing your architecture is critical.

Sooner or later, AWS will release new technologies and services that make your existing architecture less efficient than it could be. It will happen!

AWS recommends a data-driven approach with scheduled performance reviews.

Implement code deployment pipelines! Automate things when you can.

The two key sections here are:

- Benchmarking
- Load Testing

Benchmarking

Test to provide useful data about performance.

Load Testing

Use real workloads to get realistic load test results.

Read more about the review component here: [Review](#)



Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

Performance Efficiency

Review

Continuously reviewing your architecture is critical.

Sooner or later, AWS will release new technologies and services that make your existing architecture less efficient than it could be. It will happen!

AWS recommends a data-driven approach with scheduled performance reviews.

Implement code deployment pipelines! Automate things when you can.

Benchmarking is usually done at the start.

Usually quicker than load testing.

Pre-warm your resources so that results will be realistic.



CloudFormation



CodeDeploy

Benchmarking

Load Testing

Use real workloads to get realistic load test results.

Read more about the review component here: [Review](#)

Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

Performance Efficiency

Review

Continuously reviewing your architecture is critical.

Sooner or later, AWS will release new technologies and services that make your existing architecture less efficient than it could be. It will happen!

AWS recommends a data-driven approach with scheduled performance reviews.

Implement code deployment pipelines! Automate things when you can.

The two key sections here are:

- Benchmarking



Test real workloads in order to gain an understanding of production performance.

Benchmarking

Automate tests during code deployment pipeline.

Use metrics! CloudWatch is king.

Load Testing

Parallelize your processes and workloads if possible.



CloudWatch

Read more about the review component here: [Review](#)

Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

Performance Efficiency

Monitoring

After implementing your solutions, you need to monitor their performance.

Set up alarms so you know when thresholds are breached.

Be proactive in your approach — not reactive.

AWS breaks down monitoring into two sections:

- Active and Passive
- Phases

Active and Passive

Monitoring falls into one of these two categories.

Phases

Five categories fall under this section.

Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

Performance Efficiency

After implementing your architecture, you should:

Set up alarms

Be proactive

AWS best practices

Active and Passive

Phases

Monitoring

Active Monitoring

Continuously simulating user activity.

Should be continuous, lightweight, and predictable.

Can be run in *all* environments!



CloudWatch

Passive Monitoring

Common for web-based workloads.

Collects metrics from a browser.

Used to gain understanding of:

- UX
- Geo performance
- API impact

Read more about the monitoring component here: [Monitoring](#)



Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

Performance Efficiency

Monitoring

After implementing your solutions, you need to monitor their performance.



There are five phases within this category:

Set up alarm

Generation

Your scope of monitoring, metrics, and thresholds



AWS br

Aggregation

Complete view from all sources



Real-Time Processing and Alarming

Recognize and respond to events



Storage

Data management and retention policies



Analytics

Dashboards, reporting, and insights

Active and Passive

Phases

Read more about the monitoring component here: [Monitoring](#)

Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

Performance Efficiency

Trade-Offs

Consider possible trade-offs to determine optimal approaches.

For example, you may have to sacrifice a bit of durability for higher performance.

AWS allows for global deployments within a few minutes!

AWS breaks down trade-offs into four areas:

- Caching
- Partitioning or Sharding
- Compression
- Buffering

Caching

Compression

Increase performance using caching.

Compress to reduce resource requirements.

Partition/Shard

Buffering

Split data across multiple schemas.

Leverage queues for different rate times.

Read more about the trade-offs component here: [Trade-Offs](#)

[Overview](#)[Operational Excellence](#)[Security](#)[Reliability](#)[Performance Efficiency](#)[Cost Optimization](#)

AWS Well-Architected Framework

Databases are a key resource for maintaining truthful data.

Caching improves database efficiency.

There are a few different levels:

Application Level

Trade-off at app level by using app-level caches (in-memory caching).

Database Level

Use read replicas if possible.

Geographic Level

CDN usage!

[Caching](#)[Compression](#)

Increase performance using caching.

Compress to reduce resource requirements.

[Partition/Shard](#)[Buffering](#)

Split data across multiple schemas.

Leverage queues for different rate times.

Read more about the trade-offs component here: [Trade-Offs](#)

Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

Trade longer computing times for reduced storage and network requirements.

Compression can happen at different levels, including the file system, data files, and even web-based resources.

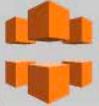
CloudFront is excellent for this, as it can compress at the edge location!

Consider non-network-based data transfer solutions for large-scale jobs.

Redshift allows for better compressions due to the column-based storage.



Amazon
Redshift



Amazon
CloudFront



Edge



AWS
Snowball

Caching

Compression

Increase performance using caching.

Compress to reduce resource requirements.

Partition/Shard

Buffering

Split data across multiple schemas.

Leverage queues for different rate times.

Read more about the trade-offs component here: [Trade-Offs](#)

[Overview](#)[Operational Excellence](#)[Security](#)[Reliability](#)[Performance Efficiency](#)[Cost Optimization](#)

AWS Well-Architected Framework

Some RDS implementations are limited to vertical scaling only.

Once you reach the vertical scaling limitations, consider sharding data.

Sharding into multiple schemas offers performance gains for write-heavy loads.

Any sharding changes require work at the application layer as well.

DynamoDB does this for you!



Amazon
DynamoDB



Amazon
DynamoDB
Accelerator

[Caching](#)[Compression](#)

Increase performance using caching.

Compress to reduce resource requirements.

[Partition/Shard](#)[Buffering](#)

Split data across multiple schemas.

Leverage queues for different rate times.

Read more about the trade-offs component here: [Trade-Offs](#)

Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

Buffering leverages queues for accepting work.

Always use durable storage when you can.

Queuing allows you to go at the customer's required pace.

Amazon SQS is the main service for queue-based messaging.

You can also use Kinesis since it uses data streams.

When using queues, be sure to keep latency requirements in mind.



Amazon
SQS



Amazon
Kinesis

Caching

Compression

Increase performance using caching.

Compress to reduce resource requirements.

Partition/Shard

Buffering

Split data across multiple schemas.

Leverage queues for different rate times.

Read more about the trade-offs component here: [Trade-Offs](#)

Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

Cost Optimization

"The ability to avoid or eliminate unneeded cost or suboptimal resources."

Cost optimization is a continuous process of refinement.

To be fully optimized, you should be fully utilizing *all* resources.

AWS has five design principles to follow:

- Adopt a consumption model
- Measure overall efficiency
- Stop spending money on data centers
- Analyze and attribute expenditure
- Use managed services

Cost-Effective Resources

Choosing the right instance and storage options

Matching Supply and Demand

Scale according to load



Expenditure Awareness

Use cost allocation tags

Optimizing Over Time

Continually reevaluate

Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

Performance Efficiency

Cost-effective Resources

This is all about choosing the correct services and resources for your use case.

You can leverage AWS discounted resources like spot and reserved instances.

Be sure to use AWS geographic tools as well.

Managed services take care of a lot of overhead spending! Use them!

Key Services



Amazon
CloudFront



AWS
Lambda



Amazon
SES



Amazon
SQS



Amazon
Route
53



Spot
instance



Amazon
SNS



Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

Performance Efficiency

Meeting Supply and Demand

The inviting feature about the cloud is the "pay for what you use" pricing points.

There is a fine line between accurately planning for need and still maintaining high availability, etc.

AWS lays out three main approaches for us:

- Demand based
- Buffer based
- Time based

Demand Based

Using this approach usually entails leveraging Auto Scaling services.

You want to focus on elasticity here.

You can leverage CloudWatch metrics to scale accurately.



Auto Scaling

Amazon CloudWatch

Application Load Balancer

Buffer Based

This approach uses a queue to buffer your requests.

SQS is a great choice for this.

You need to keep in mind acceptable delay times here.



Amazon SQS

Amazon Kinesis

Amazon DynamoDB

Demand Based

Use this when you have a predictable workload timing.

You can schedule resources to scale during the highest point of need.

Make sure your workload is consistent!



Auto Scaling

AWS CloudFormation

Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

Performance Efficiency

Expenditure Awareness

The key here is to understand your business cost drivers!

Be aware of where your costs are coming from. It is easy to lose track when you have multiple environments deployed.

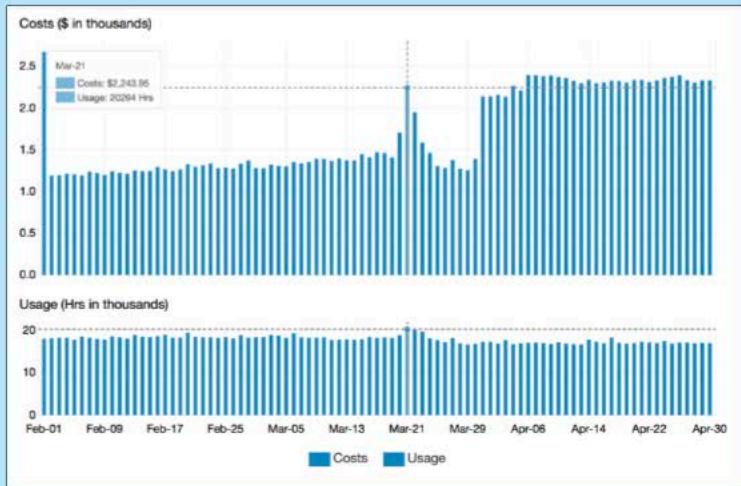
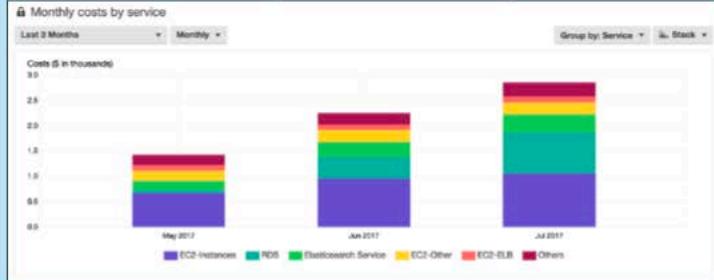
Keep in mind the following key factors:

- Stakeholders
- Visibility and Controls
- Cost Attribution
- Tagging
- Entity Lifecycle Tracking

Structure your accounts appropriately.

Always use resource tagging!

Use Cost Explorer to track your costs.



Overview

Operational Excellence

Security

Reliability

Performance Efficiency

Cost Optimization

AWS Well-Architected Framework

Performance Efficiency

Optimizing Over Time

AWS encourages you to use different approaches over time.

You want to always be checking your implementations for potentially better solutions.

- Measure, Monitor, Improve
- Staying Ever Green

Measure, Monitor, Improve

There are four key concepts:

- Establish a cost optimization function
 - Establish goals and metrics
- Gather insight and perform analysis
 - Report and validate

Plenty of tools are available for you to use to perform these tasks.



Amazon CloudWatch



Auto Scaling



AWS Trusted Advisor

Staying Ever Green

This simply means to *always* review the new services offered.

Leverage managed services when you can!

AWS News Blog

Amazon SageMaker Adds Batch Inference | by Randall Hunt | on 20 JUL 2018 | In Amazon SageMaker

At the New York Summit a few weeks ago, we announced Amazon SageMaker Batch Inference. This allows customers to make predictions in real-time using pre-trained models. SageMaker remains the easiest way to build, train, and deploy machine learning models at any scale.

Read More

Amazon Polly Update – Text-to-Speech | by Jeff Barr | on 17 JUL 2018 | In Amazon Polly

I hope that you are enjoying the Challenge and the Storage Gateway posts in order to get a better sense of how AWS services work together.

Read More

THE ORION PAPERS

Appendix

What Is a Solutions Architect?

Terminology

About the Exam

Helpful Links

Learning Activities

Exit

Welcome to the Appendix for the Orion Papers

Here you will find helpful resources and links
to help you explore AWS.

Select a resource in the navigation panel above
to explore the different parts of this appendix.

THE ORION PAPERS

Appendix

[What Is a Solutions Architect?](#)[Terminology](#)[About the Exam](#)[Helpful Links](#)[Learning Activities](#)[Exit](#)

What Is a Solutions Architect?

- In a broad sense, a solutions architect is responsible for interpreting customer requirements and defining a solution following architectural design principles.
- They provide implementation guidance based upon best practices throughout the lifecycle of the project.
- Being a successful solutions architect on AWS means being competent in the following areas:
 - Designing resilient architectures that are highly available and/or fault tolerant
 - Achieving maximum performance efficiency by leveraging caching and the inherent scalability and elasticity of AWS
 - Identifying the most cost-efficient solutions for compute and storage
 - Knowing how to secure applications, data, and networks
 - Choosing design features that enable operational excellence

A cloud architect is an IT professional who is responsible for overseeing a company's cloud computing strategy. This includes cloud adoption plans, cloud application design, and cloud management and monitoring. Cloud architects oversee application architecture and deployment in cloud environments — including public cloud, private cloud, and hybrid cloud. Additionally, cloud architects act as consultants to their organization and need to stay up to date on the latest trends and issues.

—TechTarget.com

THE ORION PAPERS

Appendix

[What Is a Solutions Architect?](#)[Terminology](#)[About the Exam](#)[Helpful Links](#)[Learning Activities](#)[Exit](#)

CSA Terminology

High Availability

Refers to architectures that continue to remain available to end users in the event of a component or systems failure. On AWS, multi-AZ architectures allow your applications to remain available in the event of an AZ outage.

Fault Tolerance

Refers to architectures that not only remain available during an outage (high availability) but also suffer no degradation in performance. Fault-tolerant architectures usually require extra redundancy and should be traded off with cost concerns.



Scalability

Refers to the ability of a system to easily increase in size and capacity in a cost-effective way (usually based on usage demand). Scaling can be **vertical** (increase the capacity of a single instance or server) or **horizontal** (add or terminate the number of instances).

Elasticity

Refers to the ease of a system's ability to change or adapt. Examples include automatically scaling out or in, updating firewall rules, and remapping IP addresses.

Cost-Efficient

Refers to making the trade-offs required to make a system as inexpensive as possible while meeting all functional requirements.

Secure

Refers to following proper security guidelines and practices to secure a system at every layer.

THE ORION PAPERS

Appendix

[What Is a Solutions Architect?](#)[Terminology](#)[About the Exam](#)[Helpful Links](#)[Learning Activities](#)[Exit](#)

The Certified Solutions Architect Exam (Released February 2018)

About the Exam

- **Length:** 130 minutes
- **Number of Questions:** 65
- **Format:** Multiple choice
- **Cost:** \$150 USD
- **Register for the exam:** <http://amzn.to/22V3swr>

- The exam must be taken at a designated testing center (located in/near most major cities)
- Results are provided immediately after completing the exam

[What is covered on the exam?](#)

How to Prepare for the Exam

- Watch and follow along with all video lessons
- Complete every Live Environment hands-on lab (at least once)
- Pass every section quiz (strive for 100%)
- Pass the final (practice) exam a total of three times
- Memorize the instructor flash card deck (or create and memorize your own)
- Read the provided AWS whitepapers (this should not be overlooked)
- Use the Orion Papers for review and study
- Participate in the Linux Academy Community or start/participate in a study group

Test Day — Exam Tips and Tricks

"Tips and Tricks for Taking an AWS Certification Exam" (blog post): <http://bit.ly/2nxE1Su>



Exam Content

The exam questions are divided up into five domains.

Each domain corresponds to a pillar of the Well-Architected Framework.

Domain	% of Exam	Framework Pillar
Domain 1: Design Resilient Architectures	34%	Reliability
Domain 2: Define Performant Architectures	24%	Performance Efficiency
Domain 3: Specify Secure Applications and Architectures	26%	Security
Domain 4: Design Cost-Optimized Architectures	10%	Cost Optimization
Domain 5: Define Operationally Excellent Architectures	6%	Operational Excellence
TOTAL	100%	

THE ORION PAPERS

Appendix

[What Is a Solutions Architect?](#)[Terminology](#)[About the Exam](#)[Helpful Links](#)[Learning Activities](#)[Exit](#)

Helpful Links

Here are some useful links that will assist you in your journey to learn AWS and become certified.

Linux Academy Resources

- **Linux Academy:** linuxacademy.com
- **Certified Solutions Architect (associate) course page:** <http://bit.ly/2nhTt3e>
- **Linux Academy Community:** <http://bit.ly/2nhUKaI>
 - Share your success with the community when you pass the exam
 - Share any feedback, thoughts, and/or tips and tricks
- **Linux Academy blog:** <http://bit.ly/2nN6QMU>
- **Instructor LinkedIn:** <https://bit.ly/2Hp0wWd>
 - Add me as a contact
 - Let's endorse each other's AWS skills and grow our network

AWS Resources

- **Amazon Web Services:** aws.amazon.com
- **AWS Documentation:** <http://amzn.to/1T9k1Og>
- **AWS CLI Reference Guide:** <http://amzn.to/1l1HXoW>
- **AWS Whitepapers:** <http://amzn.to/2nBp2sp>
 - Architecting for the Cloud: AWS Best Practices <http://bit.ly/2iJG1Ht>
 - AWS Well-Architected Framework: <http://amzn.to/2hhUCVH>
- **AWS certification exams info page:** <http://amzn.to/20k8G2u>
- **AWS certification exams sign-up page:** <http://bit.ly/1Yi4KgL>

THE ORION PAPERS

Appendix

[What Is a Solutions Architect?](#)[Terminology](#)[About the Exam](#)[Helpful Links](#)[Learning Activities](#)[Exit](#)

Linux Academy Hands-On Labs

Here is a comprehensive list of all live hands-on labs provided by Linux Academy as part of the Certified Solutions Architect (associate) level course. The purpose of live labs are to provide you with real AWS environments to practice what you have learned in the video lessons and help develop your real-world AWS skills.

Building a VPC from Scratch: <http://bit.ly/2oi20Fj>

Recap: Provisioning an EC2 Instance: <http://bit.ly/2mQytp7>

EC2 Backup Solutions with AMIs and Snapshots: <http://bit.ly/2o8Tb3p>

Accessing an Instance's User Data and Metadata: <http://bit.ly/2nB4SyB>

Setting up an ELB and Auto Scaling Group: <http://bit.ly/2oigdSK>

Building a More Secure Application with a Bastion Host and NAT Gateway: <http://bit.ly/2nwyo74>

Using S3 for Static Web Hosting: <http://bit.ly/2nwBEPE>

Configuring Backup and Archiving Solutions in S3: <http://bit.ly/2nBbD3j>

Configuring Route 53 DNS Records Sets: <http://bit.ly/2nwva3i>

Configuring a CloudFront Distribution: <http://bit.ly/2oi3bod>

VPC Peering: <http://bit.ly/2nhDMco>

RDS Lab: <http://bit.ly/2nhMbWJ>

Create and Use an SNS Topic with S3 Events: <http://bit.ly/2ozEAdG>

CloudWatch Sandbox: <http://bit.ly/2nAZ3B8>

CloudTrail Sandbox: <http://bit.ly/2nwEILJ>

CloudFormation Lab: <http://bit.ly/2mQvQU0>

Elastic Beanstalk Lab: <http://bit.ly/2nwEOD5>