Linux Academy

The Orion Papers

AWS Solutions Architect (Associate) Exam Course Manual

Enter

Linux Academy

Linux Academy
Keller, Texas
United States of America

March 31, 2017

To All Linux Academy Students:

Welcome to Linux Academy's AWS Certified Solutions Architect (associate level) prep course. As part of this course, we are introducing an exciting innovation in AWS instruction - called ***The Orion Papers***.

The Orion Papers is a non-linear, visual, interactive guide designed to enhance your learning and understanding of AWS. This guide can be used independently of the video lessons, but is meant to be supplemental and used in conjunction with the video lessons and live labs provided on linuxacademy.com.

Thank you for joining us on this AWS adventure!

Sincerely,

Thomas B. Haslett

Thomas B. Haslett
Course Author

Continue

Appendix

Welcome to the Appendix for the Orion Papers

Here you will find helpful resources and links to aid in your exploration of AWS.

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

Appendix

CSA Concepts You Need To Know

To be a good solutions architect (and before you take the exam) it is very important to know and understand the concepts outlined in this section.

Appendix

CSA Terminology

High Availability:
Refers to systems that are durable and likely to operate continuously without failure for a long time. For a solutions architect this means making sure your application (hosted in AWS) is always available when a user/customer tries to access it.

Fault Tolerance:
Is the property that enables a system to continue operating properly in the event of the failure of one or more of its components. A fault tolerant application (in AWS) would be where one of its web server can fail, and still serve traffic to visitors (and ever repair itself).

Scalability:
The ability of a system to easily increase in size and capacity in a cost effective way (usually based on usage demand).

Elasticity:
The ability of a system to increase and decrease in size (usually based on usage demand). In architecting applications, this usually refers to the ability of an application to increase and decrease server capacity on demand.

Cost Efficient:
Choosing the correct options to make a system as inexpensive as possible.

Secure:
Following proper security guidelines and practices to secure a system.

AWS Best Practices:
A set of guidelines outlined by AWS that should be followed when provisioning and using their services.

Appendix

The Certified Solutions Architect Exam:

About the Exam

- ***Length***: 80 minutes
- ***Number of Questions***: 60
- ***Format:*** Multiple choice
- ***Cost***: $150 USD
- ***Register for the exam***: http://bit.ly/1Yi4KgL

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

How to prepare for the exam

- Watch and follow along with all video lesson
- Complete every Live Lab (at least once)
- Pass every section quiz (strive for 100%)
- Pass the final (practice) exam a total of three times
- Memorize the instructor notecard 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 & Tricks for Taking and AWS Certificaiton Exam (blog post): http://bit.ly/2nxE1Su

Appendix

Helpful Links:

Here are usefull links that will assist you in your journey to learn AWS and become certified:

Linux Acadaemy 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 & tricks.
- **Linux Academy Blog**: http://bit.ly/2nN6QMU
- ***Instructor LinkedIn:*** http://bit.ly/2li1CZr
  - Add me as a contact.
  - Let's endorse each others 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
- **AWS Certificaiton exams (info page)**: http://amzn.to/20k8G2u
- **AWS Certification exam (sign-up page)**: http://bit.ly/1Yi4KgL

Appendix

CSA Concepts | Terminology | About the Exam | Helpful Links | Live Labs | Exit

Linux Academy Live Labs:

Here is a comprehensive list of all live 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 to 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 Instances 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
CHALLENGE LAB: Troubleshooting Connectivity Issues #1: http://bit.ly/2oi3P4O
CHALLENGE LAB: Troubleshooting Connectivity Issues #2: http://bit.ly/2oc1lIz
CHALLENGE LAB: Troubleshooting Connectivity Issues #3: http://bit.ly/2nkhXtE
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 an 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

Implementation & Development:

X

How to Design Cloud Services & Best Practices:

- ***Design for failure***, and create self-healing application environments.
- Always design applications with instances ***in at least two availability zones***.
- Guarantee that you have “reserved” capacity in the event of an emergency by purchasing reserved instances in a designated recovery availability zone (AWS does not guarantee on-demand instance capacity).
- Rigorously test to find single points of failure and apply high availability.
- Always enable RDS Multi-AZ and automated backups (InnoDB table support only for MySQL).
- Utilize Elastic IP addresses for fail over to “stand-by” instances when auto scaling and load balancing are not available.

- Use Route 53 to implement failover DNS techniques that include:
  - Latency based routing
  - Failover DNS routing

- Have a disaster recovery and backup strategy that utilizes:
  - Multiple Regions
  - Maintain up to date AMI’s (and copy AMI’s from one region to another)
  - Copy EBS snapshots to other regions (use CRON jobs that take snapshots of EBS)
  - Automate everything in order to easily re-deploy resources in the event of a disaster
  - Utilize bootstrapping to quickly bring up new instances with minimal configuration and allows for “generic” AMI’s

- Decouple application components using services such as SQS (when available).
- “Throw away” old or broken instances.
- Utilize CloudWatch to monitor infrastructure changes and health.
- Utilize MultiPartUpload for S3 uploads (for objects over 100MB).
- Cache static content on Amazon CloudFront using EC2 or S3 Origins.

- Protect your data in transit by using HTTPS/SSL endpoints.
- Protect data at rest using encrypted file systems or EBS/S3 encryption options.
- Connect to instances inside of the VPC using a bastion host or VPN connection.
- Use IAM roles on EC2 instances instead of using API keys; Never store API keys on an AMI

Implementation & Development:

Monitoring your AWS Environment:

Use CloudWatch for

- Shutting down inactive instances.
- Monitoring changes in your AWS environment with CloudTrail integration.
- Monitor instances resources and create alarms based off of usage and availability:
  - EC2 instances have “basic” monitoring which CloudWatch supports out of the box, and includes all metrics that can be monitored at the hypervisor level.
  - Status Checks which can automate the recovery of failed status checks by stopping and starting the instance again.
  - EC2 metrics that include custom scripts to work with CloudWatch:
    - Disk Usage; Available Disk Space
    - Swap Usage; Available Swap
    - Memory Usage; Available Memory

Use CloudTrail for

- Security and compliance.
- Monitoring all actions taken against the AWS account.
- Monitoring (and being notified) of changes to IAM accounts (with CloudWatch/SNS Integration)
- Viewing what API Keys/Users performed any given API action against an environment (i.e view what user terminated a set of instances or an individual instance).
- Fulfilling auditing requirements inside of organizations.

Use AWS Config for

- Receiving detailed configuration information about an AWS environment.
- Taking a point in time “snapshot” of all supported AWS resources to determine the state of your environment.
- Viewing historical configurations within your environment by viewing the "snapshots".
- Receiving notifications whenever resources are created, modified, or deleted.
- Viewing relationships between resources, I.E what EC2 instances an EBS volume is attached to.

Implementation & Development:

Architectural Trade-off Decisions:

Storage Trade-off Options

- S3 Standard Storage
  - 99.999999999% durability and 99.99% availability, but is the most expensive.
- S3 RRS
  - Reduce redundancy durability is 99.99%, but the storage costs is cheaper.
  - Should be used for easily reproducible data, and you should take advantage of lost object notification using S3 events.
- Glacier
  - Requires and extended timeframe to check-in and check-out data from archiving.
  - Costs are significantly reduced compared to S3 storage options.

Database Trade-off Options

- Running databases on EC2 instances:
  - Have to manage the underlying operating system.
  - Have to build for high availability.
  - Have to apply your own backups.
  - Can use additional software to cluster MySQL.
  - Requires more time to manage than RDS.

- Managed RDS database provides:
  - Fully managed database updates and does not require managing of the underlying OS.
  - Provides automatic point in time backups.
  - Easily enable Multi-AZ failover, and when a failover occurs the DNS is switched from the primary instance to the standby instance.
  - If Multi-AZ is enabled then backups are taken against the stand-by to reduce I/O freezes and updates are applied to the standby then is switched to the primary.
  - Easily create read replicas.

Implementation & Development:

X

Elasticity and Scalability:

- ***Proactive Cycle Scaling***: Scaling that occurs at a fixed interval.
- ***Proactive Event-based scaling***: Scaling that occurs in anticipation of an event.
- ***Auto-scaling based on demand***: Scaling that occurs based off of increase in demand for the application.

- Plan to scale out rather than up (Horizontal scaling):
  - Add more EC2 instances to handle increases in capacity rather than increasing instance size.
  - Be sure to design for the proper instance size to start.
  - Use tools like Auto Scaling and ELB.
  - A scaled service should be fault tolerant and operationally efficient.
  - Scalable service should become more cost effective as it grows.

- DynamoDB is a fully managed NoSQL services from AWS:
  - With high availability and scaling already built in.
  - All the developer has to do is specify required throughput for the tables.

- RDS requires scaling in a few different ways:
  - RDS does not support a cluster of instances to load balance traffic across.
  - Because of this there are a few different methods to scale traffic with RDS:
    - Utilize read replicas to offload heavy read only traffic.
    - Increase the instance size to handle increase in load.
    - Utilize ElastiCache clusters for caching database session information.

Architecting for Security in AWS:

X

Shared Security Responsibility Model:

- AWS is responsible for portions of the cloud, and you as the customer have portions of the cloud that you are responsible for - thus creating shared security responsibility.

- Reduces the operational burden (on you) as AWS operates, manages, and controls the components from the host operating system and virtualization layer, down to the physical security of the facilities in which the services operate.

- As the customer (you), using AWS means you assume the responsibility and management of the guest operating system (including, updates and security patches), other associated applications software, as well as the configuration of the AWS-provided security group firewall.

- You are also responsible for your own coded applications and custom applications built on top of the cloud.

AWS is responsible for (EC2 example)

- Facilities
- Physical security of hardware
- Network infrastructure
- Virtualization infrastructure

You (as the customer) are responsible for (EC2 example)

- Amazon Machine Images (AMIs)
- Operating systems
- Applications
- Data-in-transit
- Data-at-rest
- Data stores
- Credentials
- Policies and configuration

Architecting for Security in AWS:

AWS Platform Compliance and Security Services:

- The AWS cloud infrastructure has been architected to be flexible and secure with world-class protection, by using it's built-in security features:
  - ***Secure access*** – Use API endpoints, HTTPS, and SSL/TLS
  - ***Built-in firewalls*** – Virtual Private Cloud (VPC)
  - ***Unique users*** – AWS Identity and Access Management (IAM)
  - ***Multi-factor authentication*** (MFA)
  - ***Private subnets*** – AWS allowing private subnets on your VPC
  - ***Encrypted data storage*** – Encrypt your data in EBS, S3, Glacier, Redshift, and SQL RDS
  - ***Dedicated connection option*** – AWS Direct Connect
  - ***Perfect Forward Secrecy*** – ELB and CloudFront offer SSL/TLS cipher suites for PFS
  - ***Security logs*** – AWS CloudTrail
  - ***Asset identification and configuration*** – AWS Config
  - ***Centralized key management*** – Centralized key management service
  - ***Isolated GovCloud*** – US ITAR regulations using AWS GovCloud
  - ***CloudHSM*** – Hardware Security Model (HSM) hardware based cryptographic storage
  - ***Trusted Advisor*** – With premier support (identify security holes)

Architecting for Security in AWS:

Incorporating Common Conventional Security Products:

OS-side Firewalls

- IPTABLES
- FirewallD
- Windows Firewall

AntiVirus Software

- TrendMicro:
  - Integrates into AWS EC2 instances

Architecting for Security in AWS:

X

DDoS Mitigation:

- When mitigating against DOS/DDOS attacks, use the same practice you would use on your on-premise components:
  - Firewalls:
    - Security groups
    - Network access control lists
    - Host-based firewalls
  - Web application Firewalls (WAFS)
  - Host-based or inline IDS/IPS (Trend Micro)
  - Traffic shaping/rate limiting

- Along with your traditional approaches for DOS/DDOS attack mitigation, AWS provides capabilities based on its elasticity:
  - You can potentially use CloudFront to absorb DOS/DDOS flooding attacks.
  - A potential attackers trying to attack content behind a CloudFront distribution is likely to send most requests to CloudFront edge locations, where the AWS infrastructure will absorb the extra requests with minimal to no impact on the back-end customer web servers.

- We MUST have permission to do Port Scanning on any of your EC2 instances.
- INGRESS Filtering on all incoming traffic onto their network.

Architecting for Security in AWS:

Encryption Solutions:

- S3 has built-in features that allow you to encrypt your data:
  - AES-256 bit encryption that encrypts data-at-REST in an S3 bucket.
  - AWS will decrypt the data and sent to you when you download it.

- EBS encrypted volumes:
  - You can select to have all data encrypted that is stored on an EBS for volume.
  - If a snap-shot it taken, that snap-shot is automatically encrypted.

- RDS encryption:
  - Aurora, MySQL, Oracle, PostegreSQL, and MS SQL all support this feature.
  - Encrypts the underlying storage space for the instance.
  - Automated Backups are encrypted (as well as snap-shots).
  - Read Replicas are encrypted.
  - RDS provides SSL endpoints to encrypt a connection to a DB instance.

Architecting for Security in AWS:

X

Complex Access Control:

- Through IAM policies, AWS gives us the ability to create extremely complex and granular permission policies for our users (all the way down to the resource level).
- IAM polices with resource level permissions:
  - EC2: Create permissions for instances such as reboot, start, stop, or terminate based all the way down to the instance ID.
  - EBS volumes: Attach, Delete, Detach.
  - EC2 actions that are not one of these above are not governed by resource-level at this time.
- This is not EC2 limited can also include services such as RDS, S3, etc.

- Additional security measures, such as MFA authentication are also available when acting on certain resources:
  - For example, you can require MFA before an API request to delete an object within an S3 bucket.

Architecting for Security in AWS:

CloudWatch for the Security Architect:

CloudWatch Security

- Request are signed with an HMAC-SHA1 signature, calculated from the request and the user's private key.
- CloudWatch control API is only accessible via SSL encrypted endpoints.
- CloudWatch access is given via IAM permission policies, essentially giving users permissions that are only needed (only give access to CloudWatch if they need access to CloudWatch).
- Use CloudWatch and CloudTrail to monitor changes inside the AWS environment.
    - We can ask CloudWatch to notify us (via SNS) if there has been changes, for example:
        - Changes to IAM security credentials.
        - Assigning access policies to users.
        - Adding/deleting users.
- It is important to know how we can use CloutWatch for security in our AWS environment.

Architecting for Security in AWS:

X

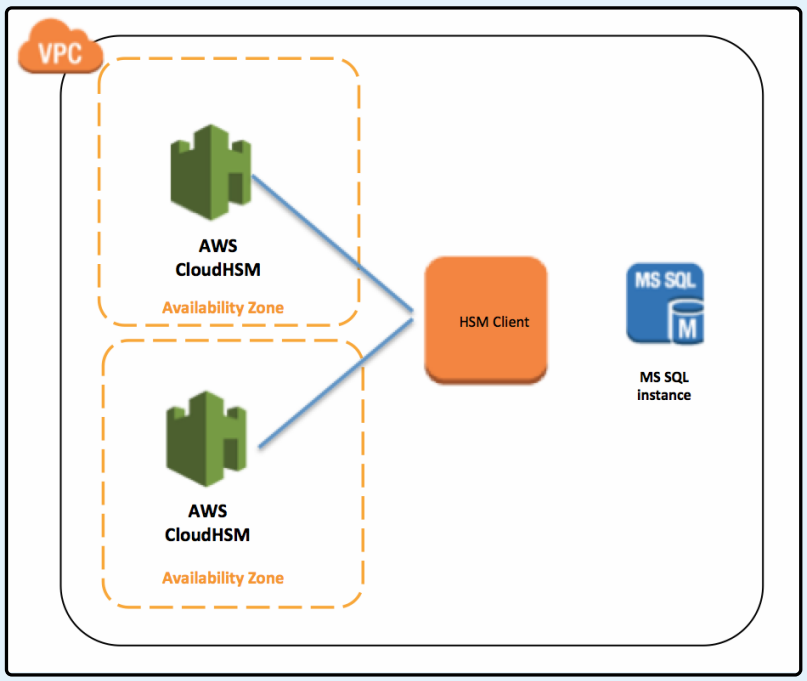
CloudHSM:

- HSM (Hardware Security Module) is a dedicated physical machine/appliance isolated in order to store security keys and other types of encryption keys used within an application.
- The key is used within the domain of the HSM appliance instead of being exposed outside the appliance.

- HSM Appliances have special security mechanisms to make them more secure:
  - The security key is only used within the HSM.
  - An HSM client is used to expose the APIs of the HSM.
  - So an application can communicate with HSM to do the encryption (or decryption) of the data that we are requesting.
  - They appliance is physically isolated from other resources.
  - Tamper resistant (build to notify via advanced logging).
  - On AWS, even though they are hosting the appliance, AWS engineers have NO access to the keys (only to manage and update the appliance).
  - If the keys are lost or reset (to access the appliance), you will never be able to access the data stored on the appliance.

- Some types of keys that might be stored on HSMs:
  - Keys used to encrypt file systems
  - Keys used to encrypt databases
  - Keys used to provide DRM
  - Used with S3 encryption

- When to use CloudHSM instead of something like Key Management Service?
  - Generally, compliance requirements require it or internal security policy require it.
  - Not even AWS engineers have access to the keys on the CloudHSM appliance, only access to “manage” the appliance.



Architecting for Security in AWS:

Disaster Recovery:

Business disaster recovery key words: Very important for AWS CSA Exam

Recovery time objective (RTO): Time it takes after a disruption to restore operations back to its regular service level, as defined by the companies operational level agreement. (i.e If the RTO is 4 hours, you have 4 hours to restore service back to an acceptable level).

Recovery point objective (RPO): Acceptable amount of data loss measured in time. (i.e if the system goes down at 10PM, and RPO is 2 hours, then you should recover all data as part of the application as it was before 8PM).

Not only should you design for disaster recovery for your applications running on AWS, you can also use AWS as a disaster recovery solution for your on-premise applications or data. The AWS services used should be determined based off of the business RTO and RPO operational agreement.

Pilot Light: A minimal version of your production environment that is running on AWS. This allows for replication from on-premise servers to AWS, and in the event of a disaster the AWS environment spins up more capacity (elasticity/automatically) and DNS is switch from on-premise to AWS. It is important to keep up to date AMI and instance configurations if following pilot light protocol.

Warm Standby: Has a larger foot print than a pilot light setup, and would most likely be running business critical applications in “standby”. This type of configuration could also be used as a test area for applications.

Multi-Site Solution: Essentially clones your “production” environment, which can either be in the cloud or on premise. Has an active-active configuration which means instances size and capacity are all running in full standby and can easily convert at the flip of a switch. Methods like this could also be used to “load balance” using latency based routing or Route 53 failover in the event of an issue.

Services Examples:

- Elastic Load Balancer and Auto Scaling
- Amazon EC2 VM Import Connector
- AMI’s with up to date configurations
- Replication from on-premise database servers to RDS
- Automate the increasing of resources in the event of a disaster
- Use AWS Import/Export to copy large amounts of data to speed up replication times (also used for off site archiving)
- Route 53 DNS Failover/Latency Based Routing Solutions
- Storage Gateway (Gateway-cached volumes/Gateway-stored volumes)

CSA Concepts:

X

What is a Solutions Architect?

- In a broad sense, a solutions architect is responsible interpreting requirements (through analysis) and turning it into architecture that can be used by others to build/implement the solution.

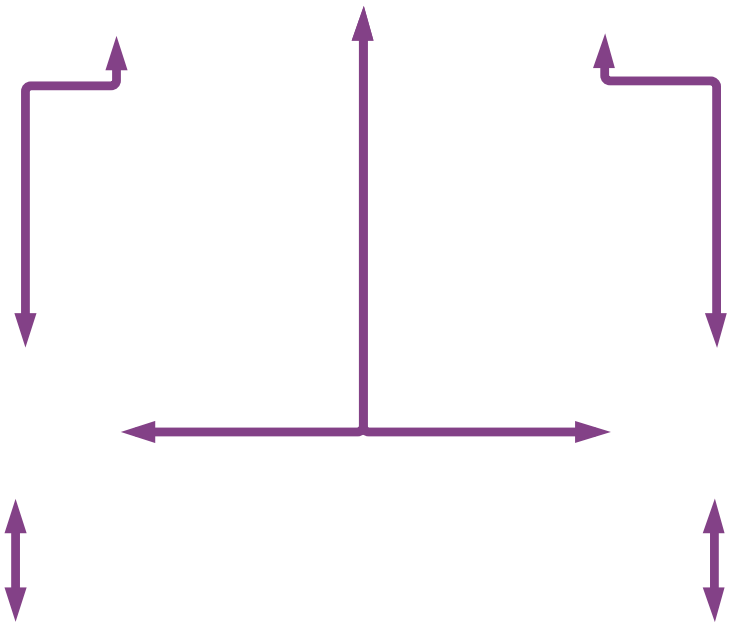
- In the world of AWS (and cloud computing in general) it means being competent in the following areas: (source: AWS)
  - Designing and deploying scalable, highly available, and fault tolerant systems on AWS.
  - Lift and shift of an existing on-premises application to AWS.
  - Ingress and egress of data to and from AWS.
  - Selecting the appropriate AWS service based on data, compute, database, or security requirements.
  - Identifying appropriate use of AWS architectural best practices.
  - Estimating AWS costs and identifying cost control mechanisms.

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***

Account & Services Layer

Physical & Networking Layer

Appendix



On-premises Data Center

On-Premises Servers

Hybrid Environments

Account & Services Layer

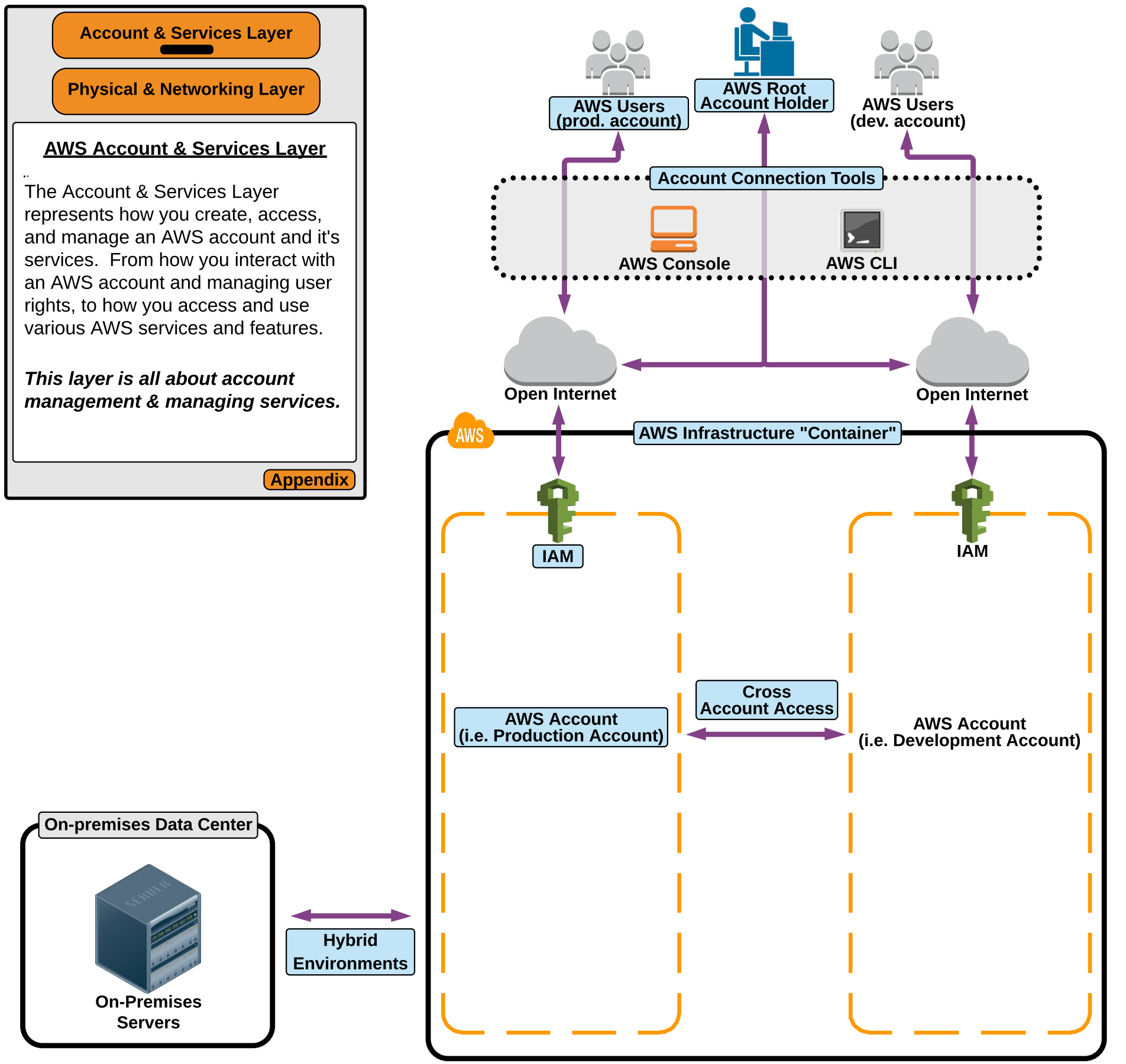
Physical & Networking Layer

AWS Account & Services Layer

The Account & Services Layer represents how you create, access, and manage an AWS account and it's services. From how you interact with an AWS account and managing user rights, to how you access and use various AWS services and features.

This layer is all about account management & managing services.

Appendix



Account & Services Layer

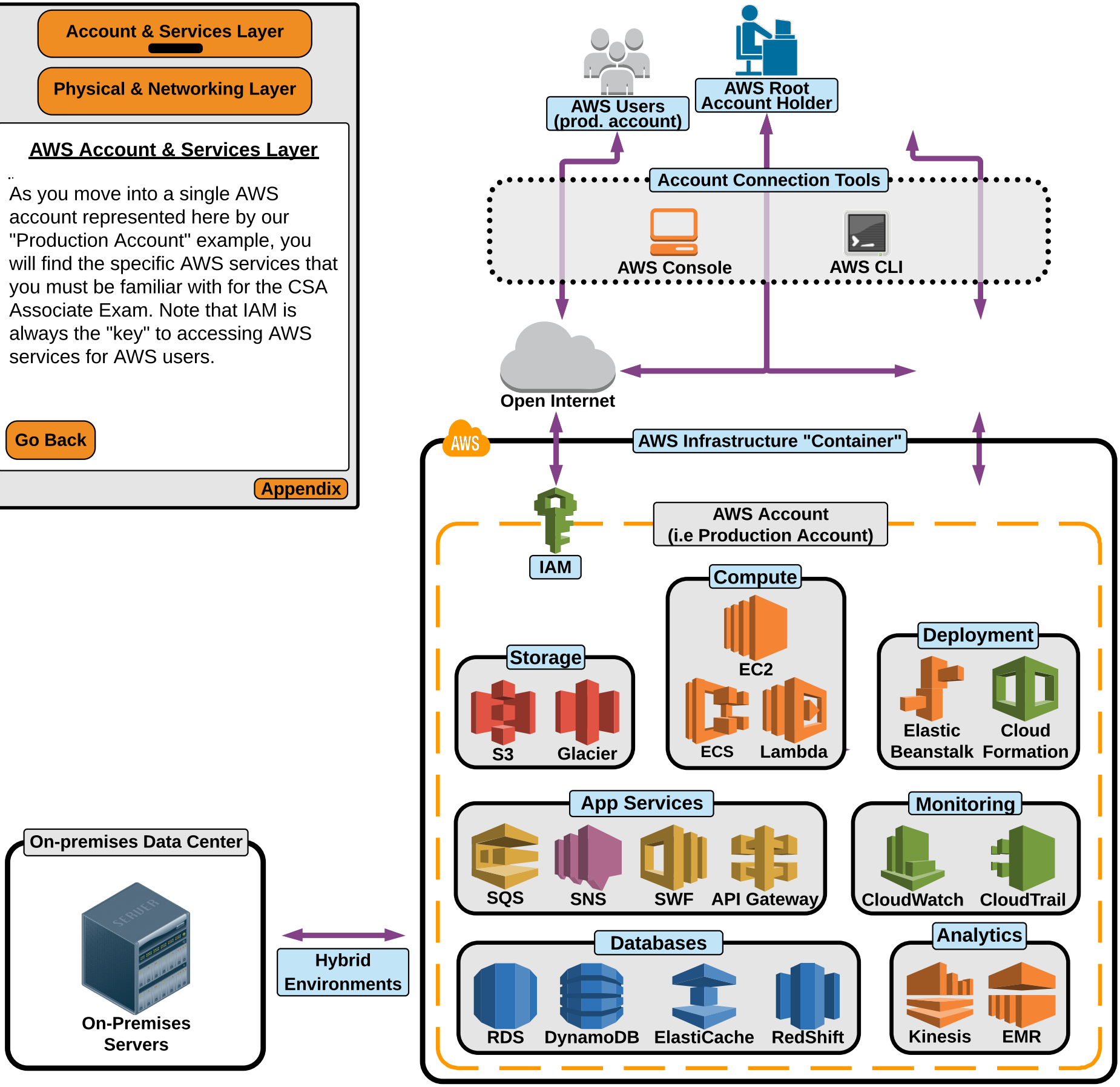
Physical & Networking Layer

AWS Account & Services Layer

As you move into a single AWS account represented here by our "Production Account" example, you will find the specific AWS services that you must be familiar with for the CSA Associate Exam. Note that IAM is always the "key" to accessing AWS services for AWS users.

Go Back

Appendix



Root User:

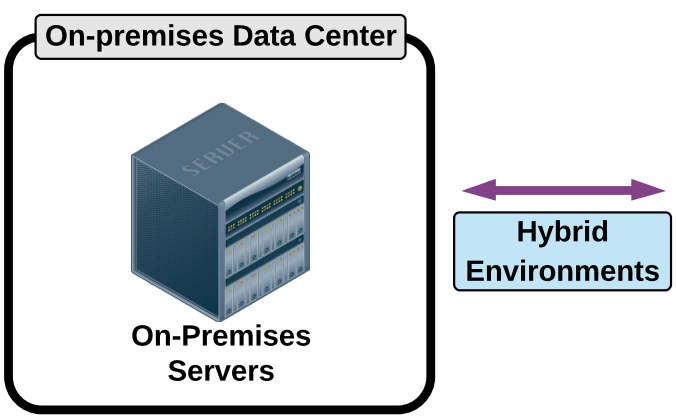
- The user created when you first create your AWS account is called the ***"root"*** user.
- It's credentials are the email address and password used when signing up for your AWS account.
- By default, the root user has ***FULL*** administrative rights and access to every part of the account.

Best Practices for Root User:

- You should not use the root user for daily work and AWS administration. You should create a second user that has admin rights and sign in as that user for daily work.

- You should always protect your root account with MFA.

Appendix

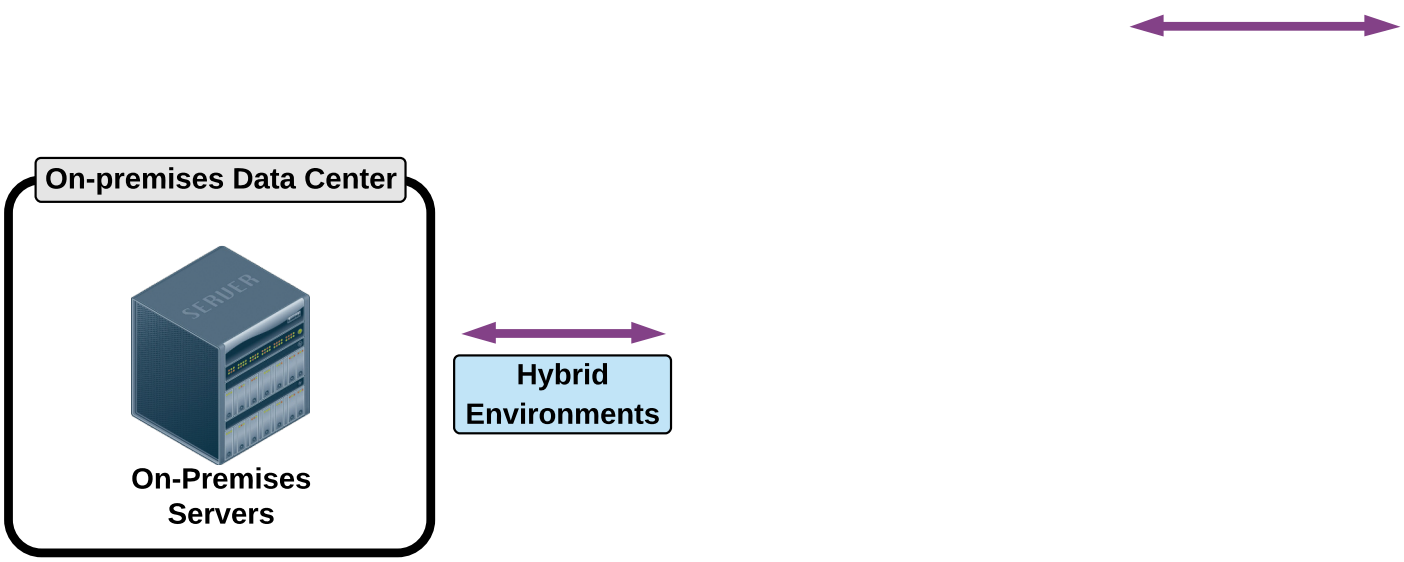


Cross Account Access:

- ***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, as seen in the diagram below).

- Cross account access is handled through the use of ***IAM roles***.
- Users in one account wil ASSUME a role that grants access to resources in another.

- ***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 sign into another in order to access resources that are in different AWS accounts.

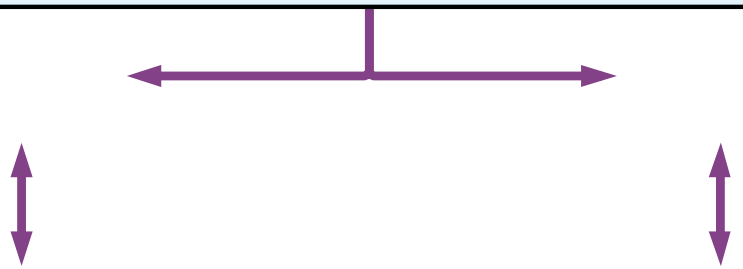


AWS Users:

X

- This represents an AWS users that you may create (in IAM), who will have varying degrees of access to the "production" AWS account in the example below.

Appendix



On-premises Data Center

On-Premises Servers

Hybrid Environments

Account Connection Tools:

X

Console

CLI

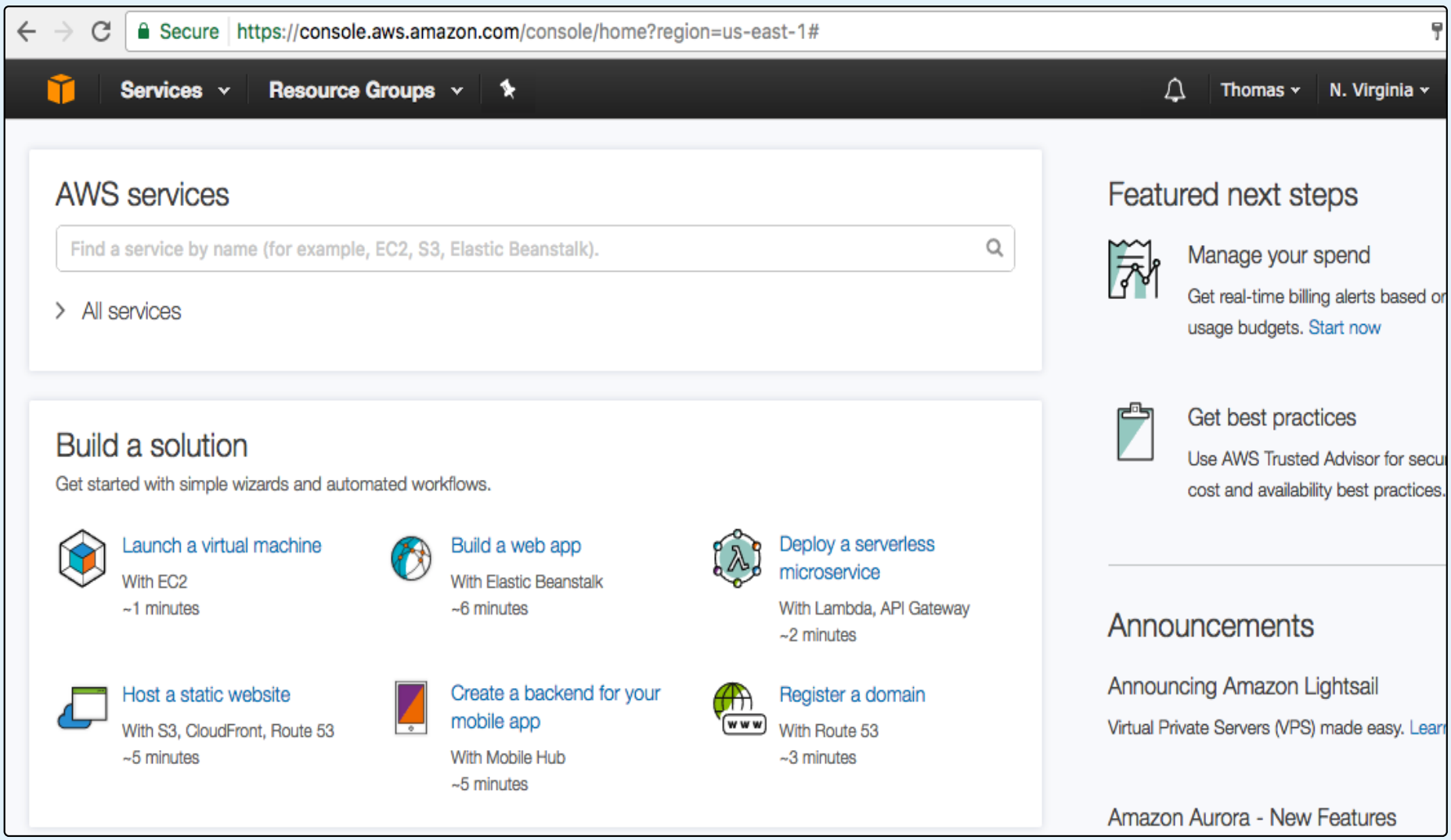
Account & Services Layer

Physical & Networking Layer

AWS Management Console:

- The AWS Management Console (generally referred to as the "console") is the primary means for which we will access and interact with AWS in this course.
- All actions done in the console are API Calls.

Example:



Account & Services Layer

Physical & Networking Layer

AWS Command Line Interface (CLI):

- The AWS Command Line Interface (generally referred to as the "CLI") is a text based interface for accessing and administering AWS resources.
- All commands executed using the CLI are API calls - and require API Key configuration.

Example:

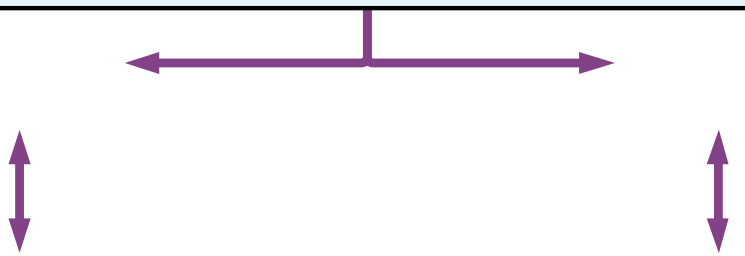
```
MacBook-Pro:downloads thaslett$ ssh -i "cli.pem" ec2-user@ec2-54-9
1-117-71.compute-1.amazonaws.com
Last login: Sat Feb  4 20:47:08 2017 from 172.58.40.128

       __|  __|_  )
       _|  (     /   Amazon Linux AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-ami/2016.09-release-notes/
[ec2-user@ip-172-31-31-150 ~]$
```

AWS Infrastructure Container:

X

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

Appendix



On-premises Data Center

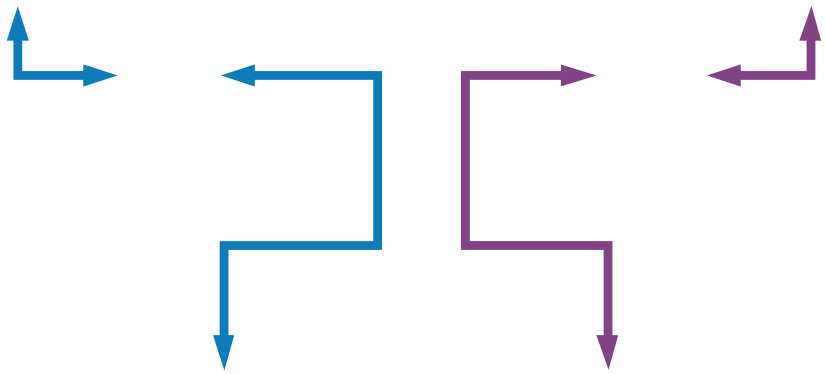
On-Premises Servers

Hybrid Environments

Account & Services Layer

Physical & Networking Layer

Appendix



On-premises Data Center

On-Premises Servers

Hybrid Environments

Account & Services Layer

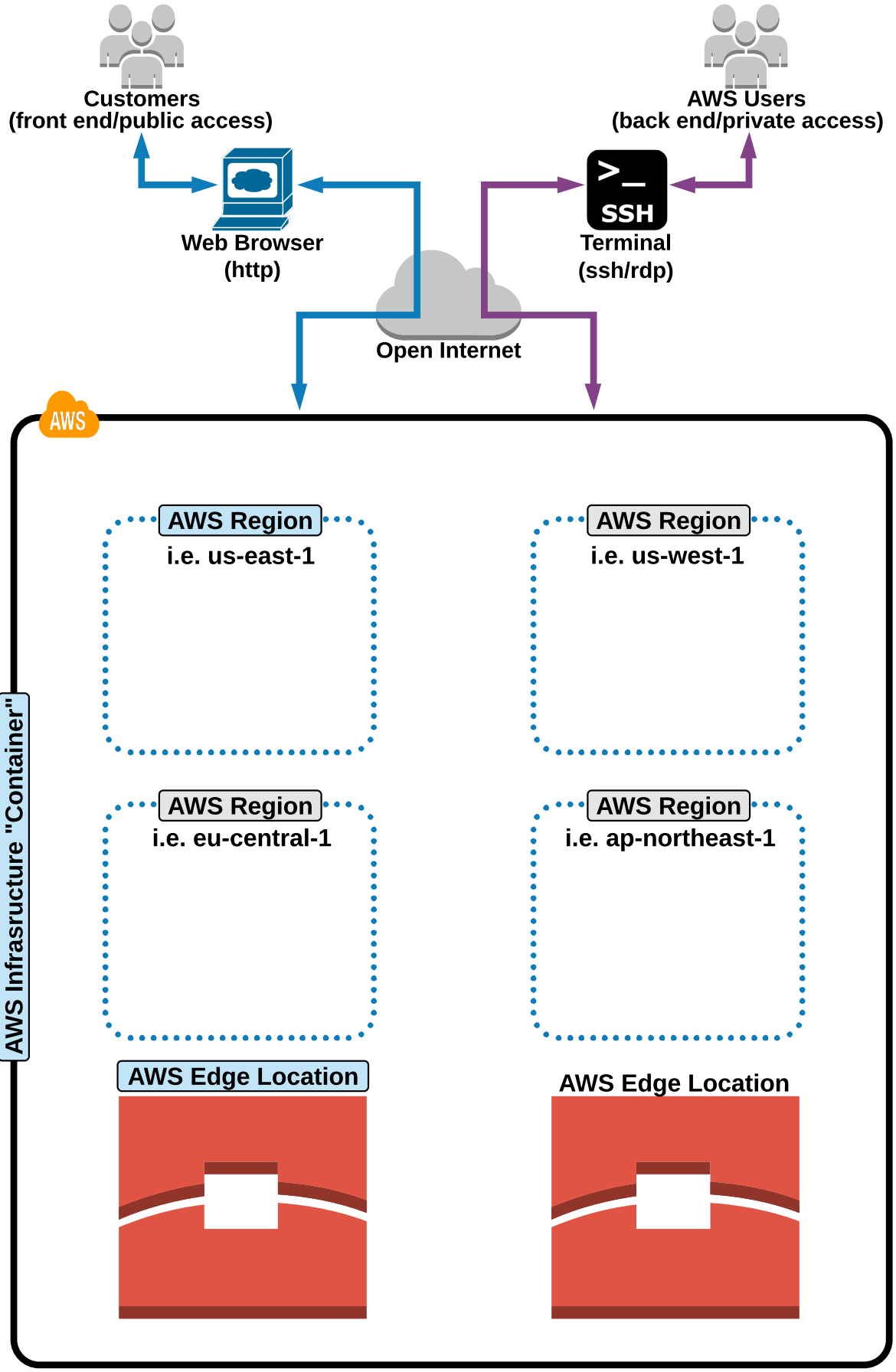
Physical & Networking Layer

AWS Physical & Networking Layer

The Physical & Networking Layer represents the global infrastructure of AWS in terms of where resources are physically located around the world and how data flows through the AWS network.

This layer is all about how AWS is organized, and how internal and external communication with AWS works.

Appendix



Account & Services Layer

Physical & Networking Layer

AWS Physical & Networking Layer

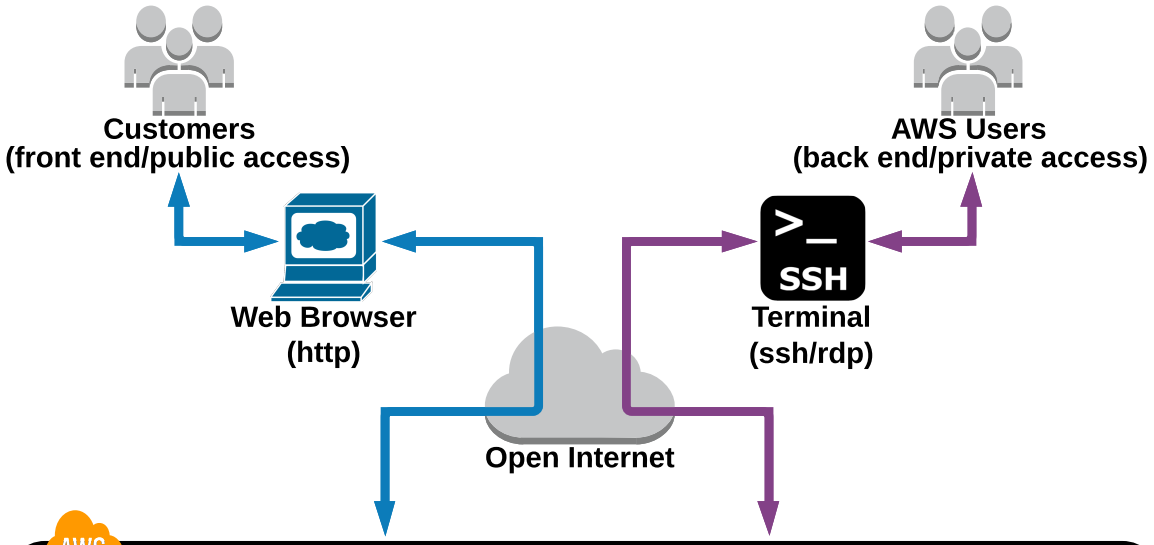
(regions)

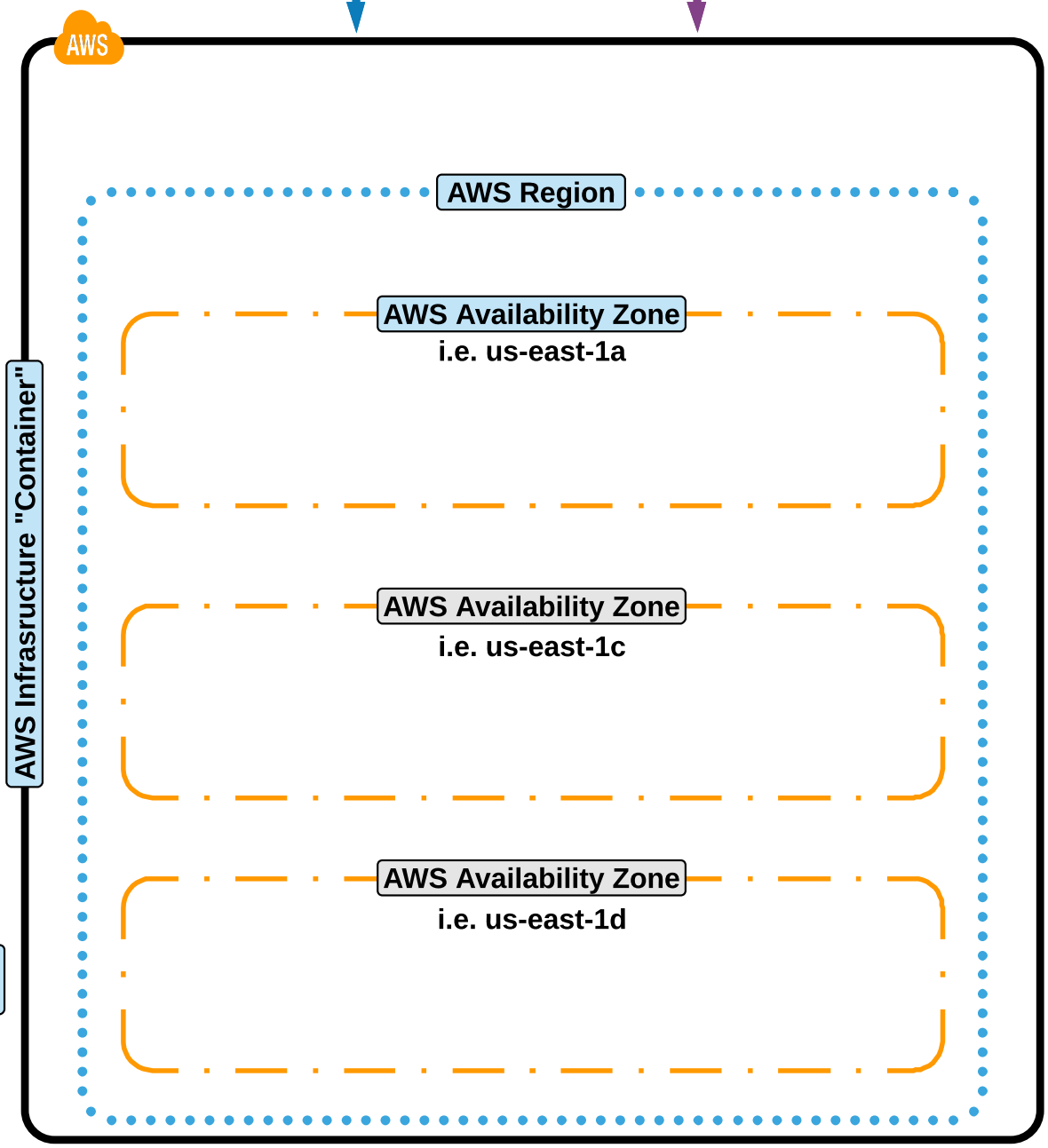
You have now moved into an AWS ***Region***:

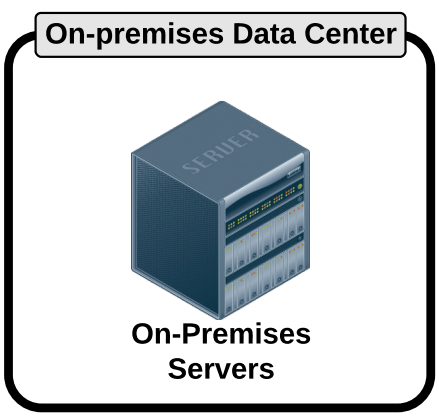
Each ***Region*** consist of multiple ***Availability Zones***. Availability zones provide the foundation for ***High Availability and Fault Tolerance***.

Go Back

Appendix







Hybrid Environments

Account & Services Layer

Physical & Networking Layer

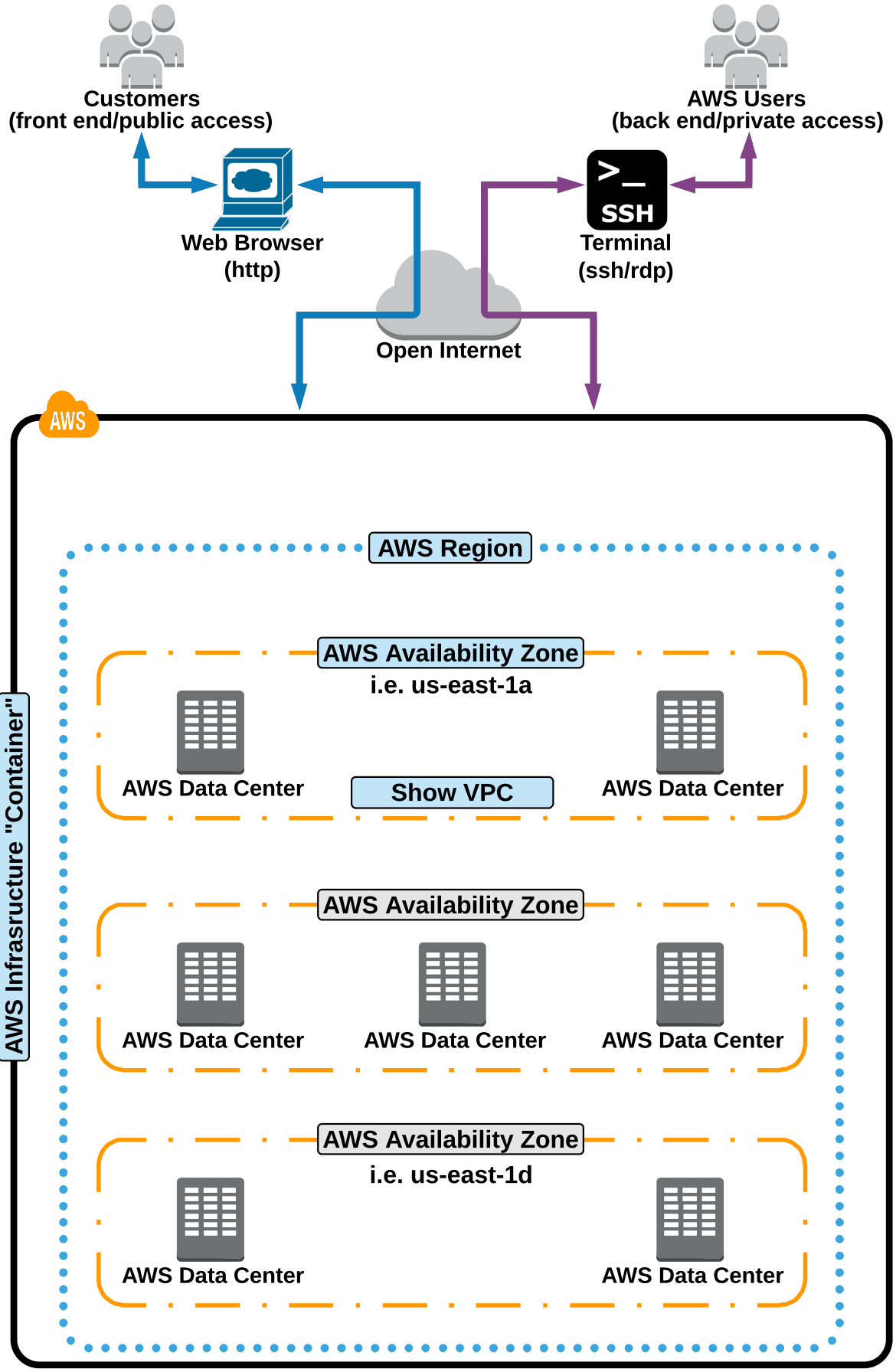
AWS Physical & Networking Layer

(Availability Zones)

Physically located in each ***Availability Zone*** is where you will find AWS ***Data Centers***. These Data Centers house the servers that run most AWS services and provide the hardware for ***VPC Networking***.

Go Back

Appendix

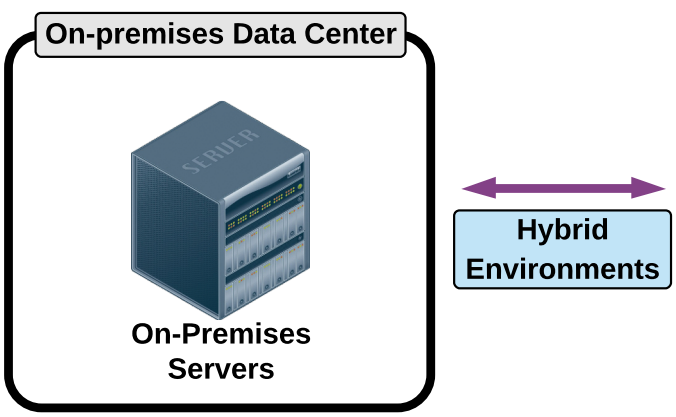


Edge Location:

- An ***Edge Location*** is an AWS datacenter which does not contain AWS services.
- Instead, it is used to deliver content to parts of the world.

- An example would be ***CloudFront*** which is a CDN:
  - Cached items such as a PDF file can be cached on an edge location which reduces the amount of "space/time/latency" required for a request from that part of the world.

Appendix



Account & Services Layer

Physical & Networking Layer

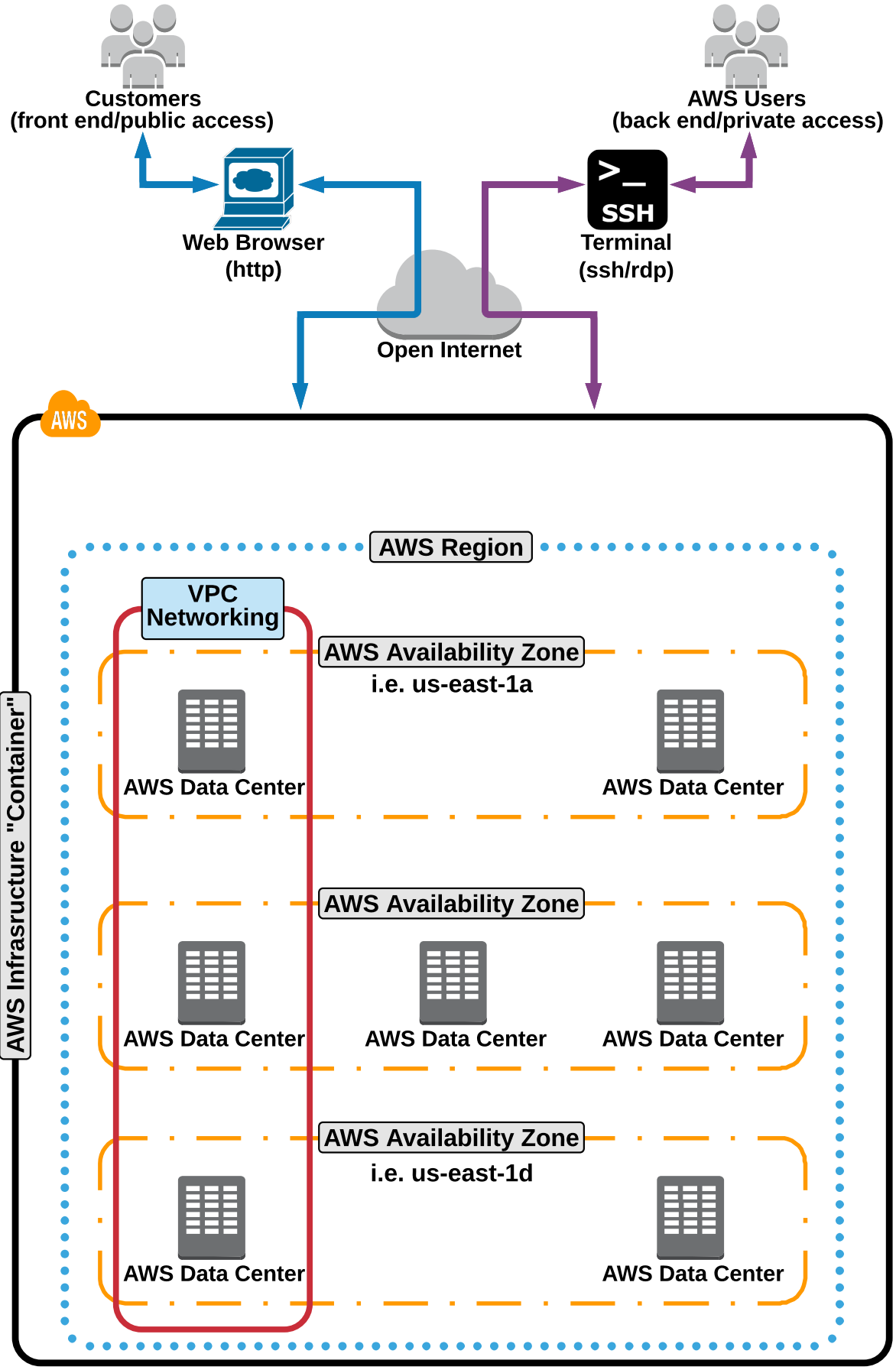
AWS Physical & Networking Layer

(AZs with VPC shown)

Demonstrated here, we can see how an AWS ***VPC*** spans ***multiple Availability Zones and Data Centers***. This allows for ***highly available and fault tolerant*** applications and architecture when using VPCs.

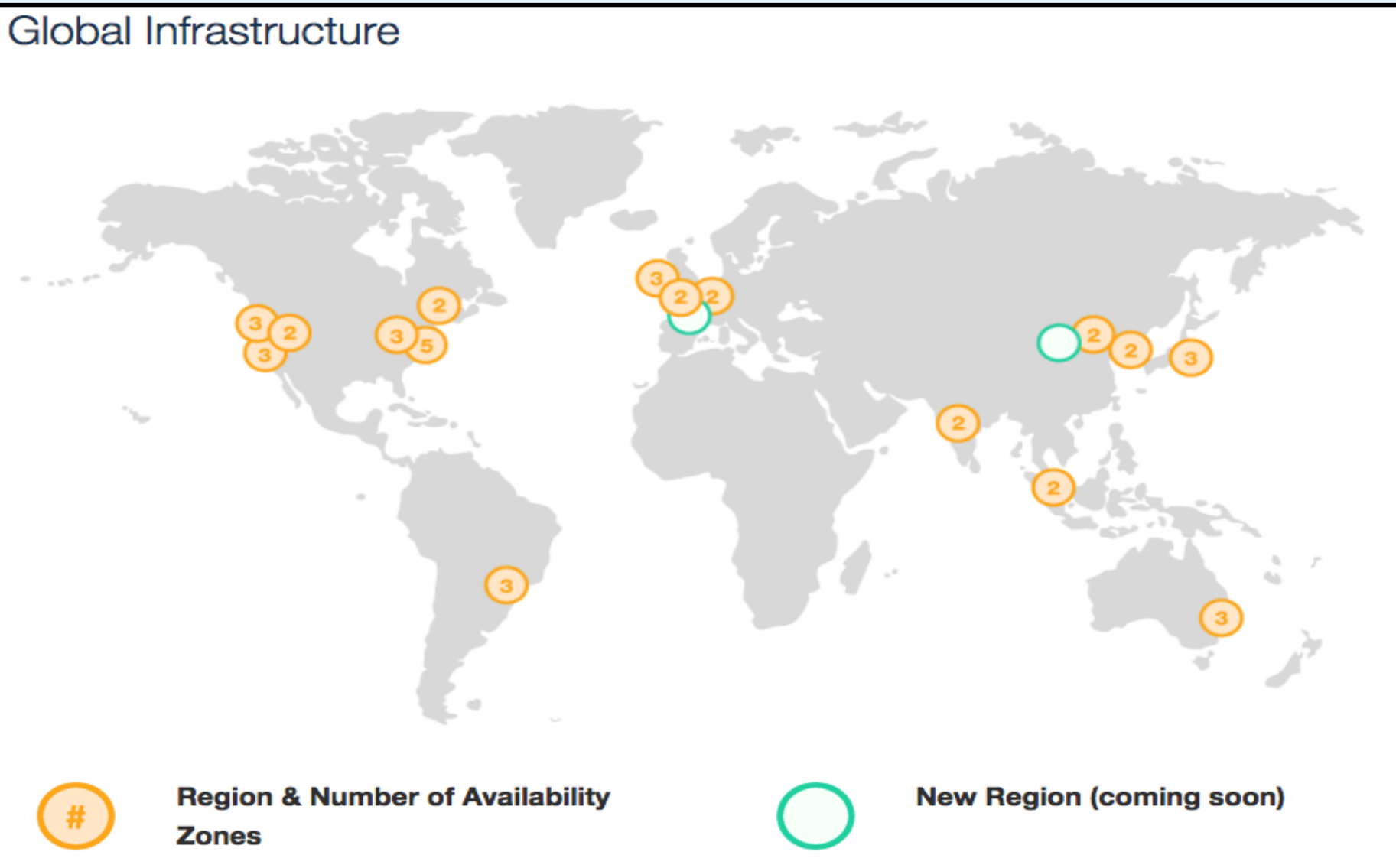
Go Back

Appendix



AWS Regions:

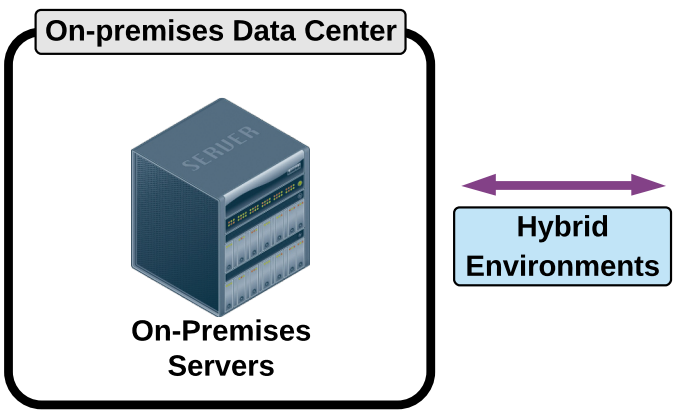
- AWS is made up of regions which are a grouping of independently separated data centers in a specific geographic regions known as “Availability zones”.
- Availability of regions allows the architect to design applications to conform to specific laws and regulations for specific parts of the world. When viewing a region in the console you will only view resources in one region at a time but they will be across all AZs within that region.
- Some AWS services work “globally” and not within a specific region. For example users created in IAM will work across regions



Availability Zones:

- Availability zones work together in a region to make up a collection of your AWS resources.
- Properly designed applications will utilize multiple availability zones for fault tolerance and failover.
- AZ's (as they are known) have direct low latency connections between each AZ in a region but each AZ is isolated from other AZ's to ensure fault tolerance.

Appendix

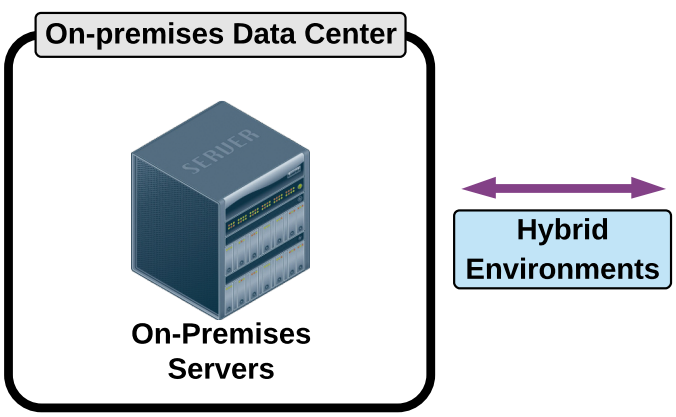


AWS Infrastructure Container:

X

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

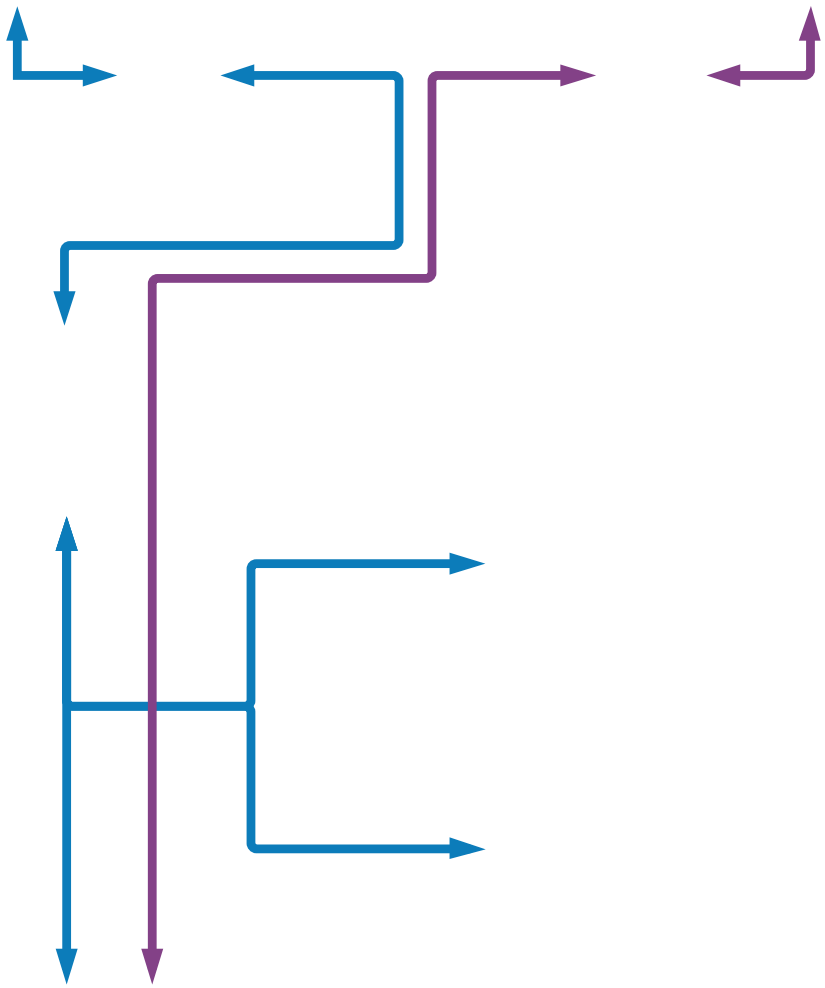
Appendix



Account & Services Layer

Physical & Networking Layer

Appendix



On-premises Data Center

On-Premises Servers

Hybrid Environments

Account & Services Layer

Physical & Networking Layer

AWS Physical & Networking Layer

(Highly Available & Fault Tolerant)

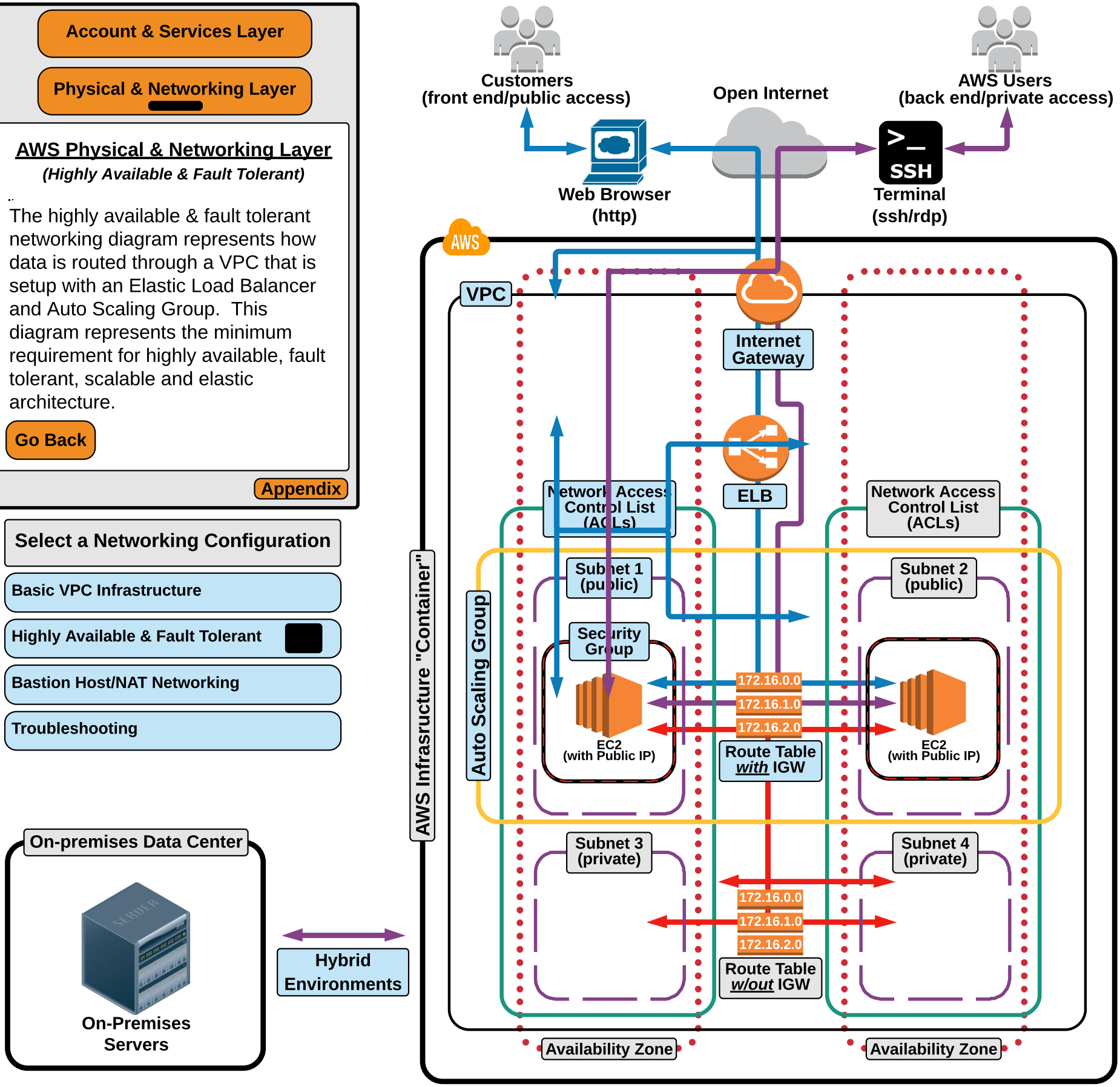
The highly available & fault tolerant networking diagram represents how data is routed through a VPC that is setup with an Elastic Load Balancer and Auto Scaling Group. This diagram represents the minimum requirement for highly available, fault tolerant, scalable and elastic architecture.

Go Back

Appendix

Select a Networking Configuration

- Basic VPC Infrastructure
- Highly Available & Fault Tolerant
- Bastion Host/NAT Networking
- Troubleshooting



Account & Services Layer

Physical & Networking Layer

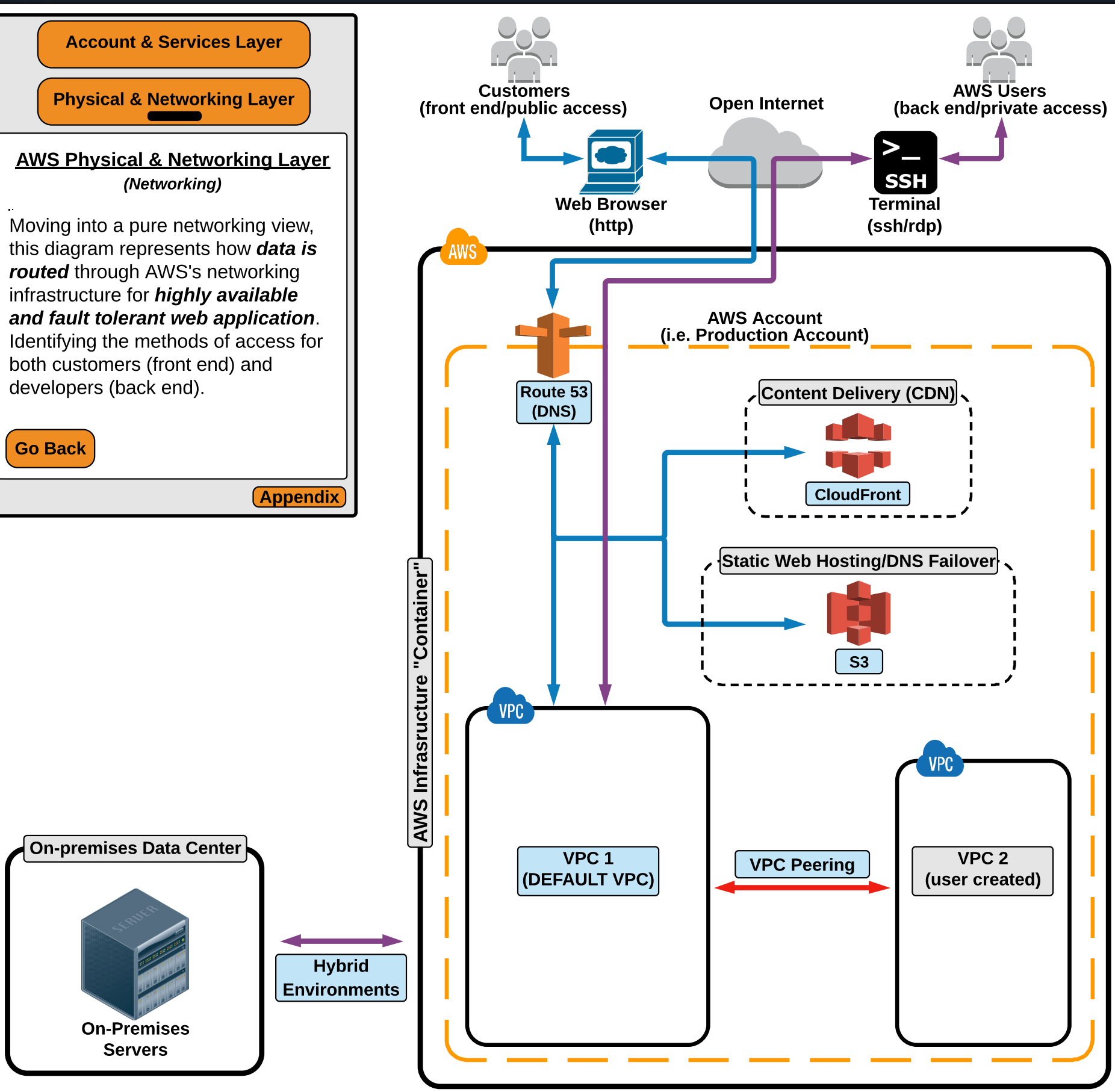
AWS Physical & Networking Layer

(Networking)

Moving into a pure networking view, this diagram represents how ***data is routed*** through AWS's networking infrastructure for ***highly available and fault tolerant web application***. Identifying the methods of access for both customers (front end) and developers (back end).

Go Back

Appendix



Account & Services Layer

Physical & Networking Layer

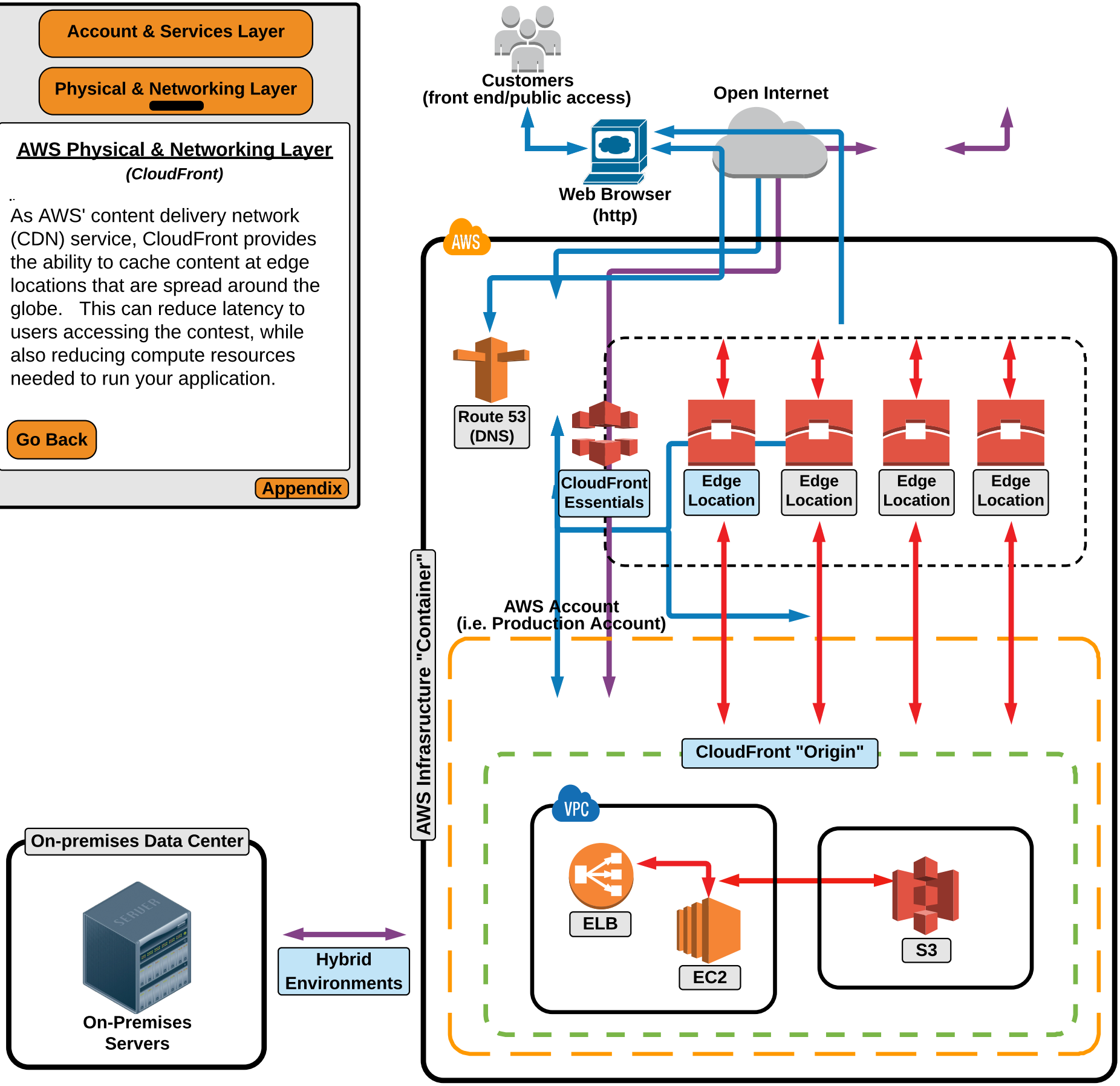
AWS Physical & Networking Layer

(CloudFront)

As AWS' content delivery network (CDN) service, CloudFront provides the ability to cache content at edge locations that are spread around the globe. This can reduce latency to users accessing the contest, while also reducing compute resources needed to run your application.

Go Back

Appendix



Account & Services Layer

Physical & Networking Layer

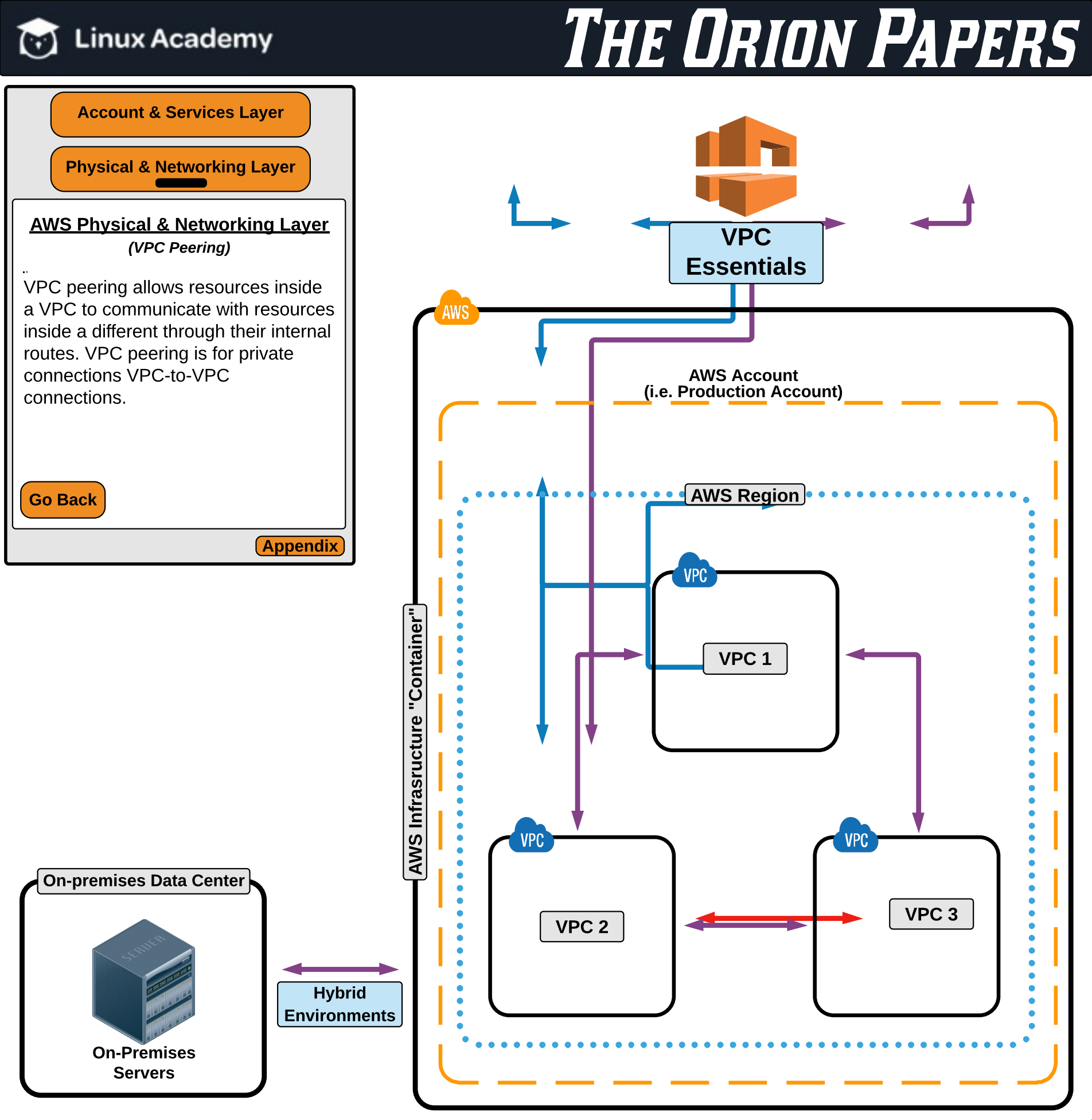
AWS Physical & Networking Layer

(VPC Peering)

VPC peering allows resources inside a VPC to communicate with resources inside a different through their internal routes. VPC peering is for private connections VPC-to-VPC connections.

Go Back

Appendix



The Orion Papers

Account & Services Layer

Physical & Networking Layer

AWS Physical & Networking Layer

(Basic VPC Networking)

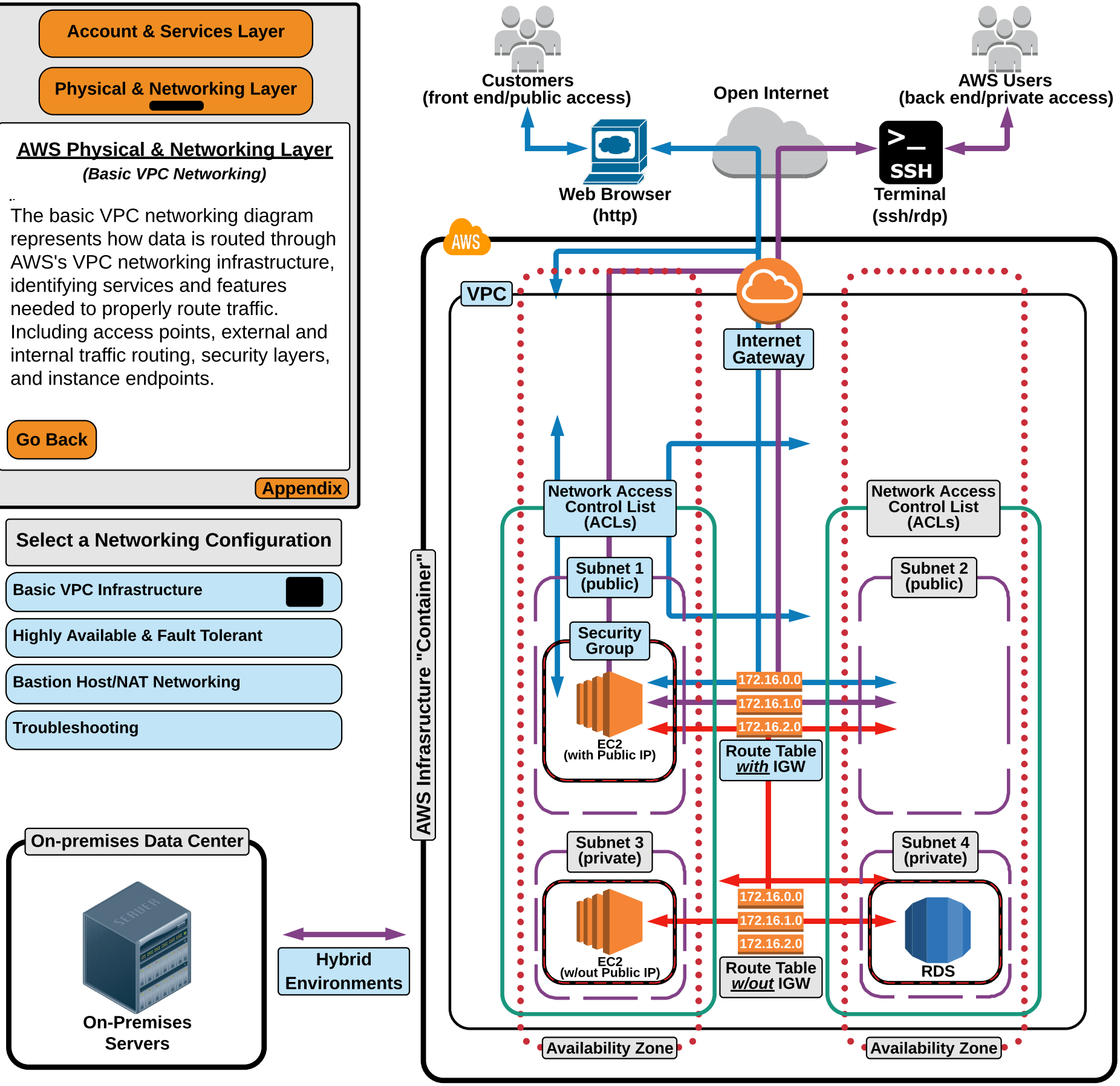
The basic VPC networking diagram represents how data is routed through AWS's VPC networking infrastructure, identifying services and features needed to properly route traffic. Including access points, external and internal traffic routing, security layers, and instance endpoints.

Go Back

Appendix

Select a Networking Configuration

- Basic VPC Infrastructure
- Highly Available & Fault Tolerant
- Bastion Host/NAT Networking
- Troubleshooting



Account & Services Layer

Physical & Networking Layer

AWS Physical & Networking Layer

(Basic VPC Networking)

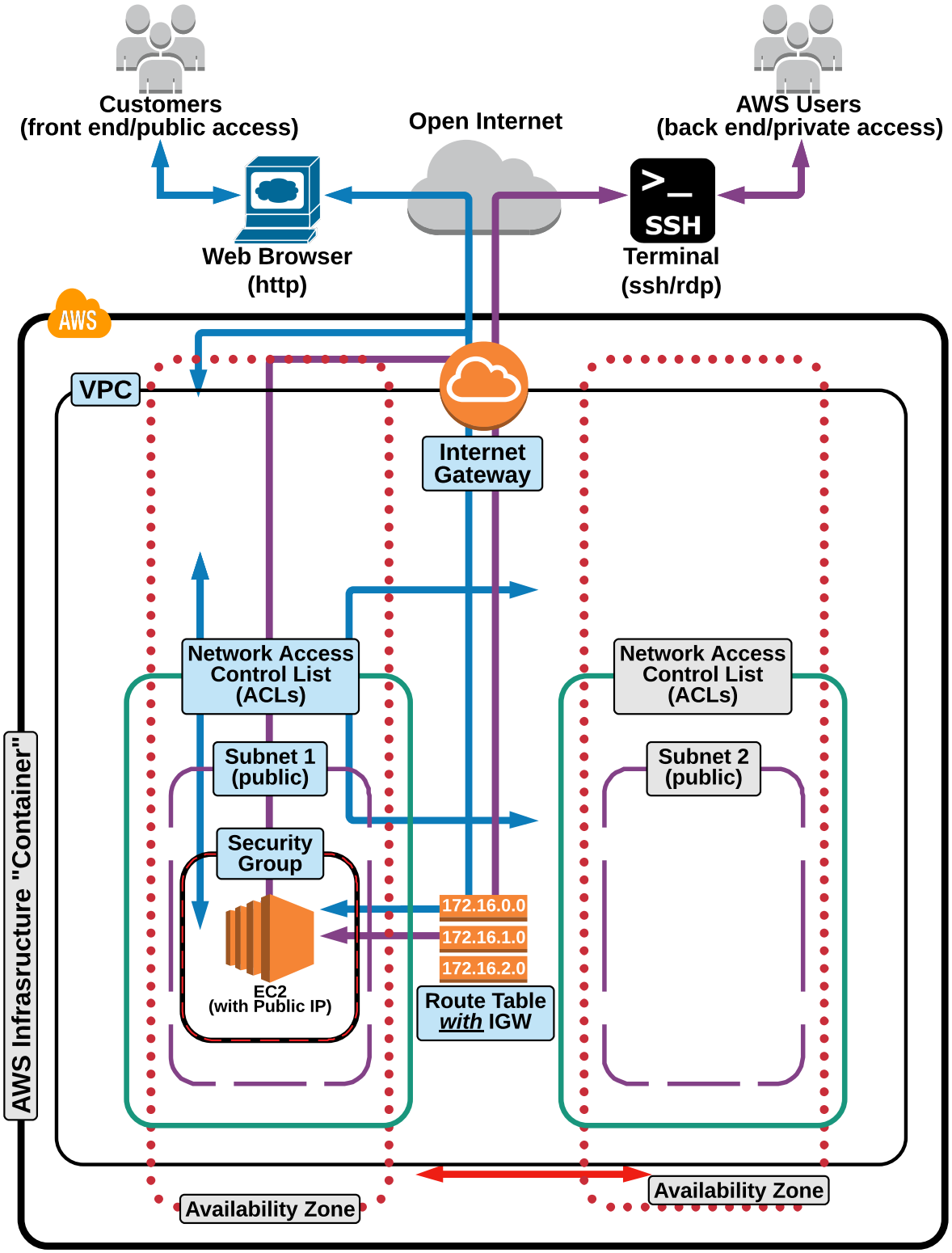
The basic VPC networking diagram represents how data is routed through AWS's VPC networking infrastructure, identifying services and features needed to properly route traffic. Including access points, external and internal traffic routing, security layers, and instance endpoints.

Go Back

Appendix

Select a Networking Configuration

- Basic VPC Infrastructure
- Highly Available & Fault Tolerant
- Bastion Host/NAT Networking
- Troubleshooting



On-premises Data Center

On-Premises Servers

Hybrid Environments

Text

Account & Services Layer

Physical & Networking Layer

AWS Physical & Networking Layer

(Bastion Host & NAT Networking)

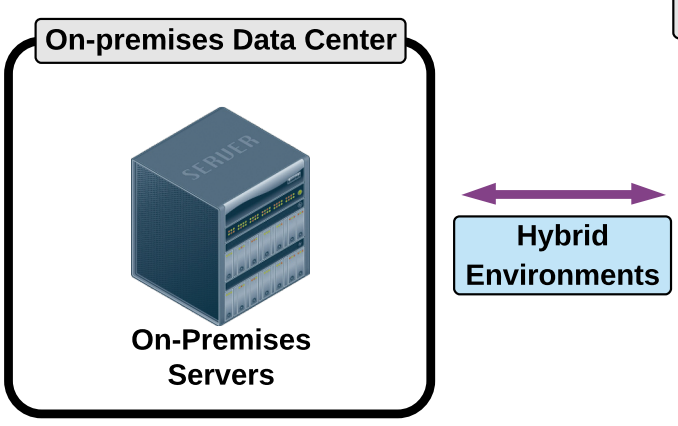
The Bastion Host & NAT networking diagram represents how data is routed through a VPC that is setup for increased security. By using a bastion host, NAT gateway, and ELB - you can protect EC2 instances by placing them in private subnets, while still being able to access and serve traffic to them.

Go Back

Appendix

Select a Networking Configuration

- Basic VPC Infrastructure
- Highly Available & Fault Tolerant
- Bastion Host/NAT Networking
- Troubleshooting



Account & Services Layer

Physical & Networking Layer [redacted]

AWS Physical & Networking Layer

(Troubleshooting)

Here you will find many common EC2/VPC Networking connectivity and other trouble shooting issue. Along with their suggested solutions.

Go Back

Appendix

Select a Networking Configuration

- Basic VPC Infrastructure
- Highly Available & Fault Tolerant
- Bastion Host/NAT Networking
- Troubleshooting [redacted]

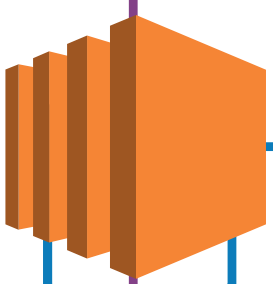
On-premises Data Center

On-Premises Servers

Hybrid Environments

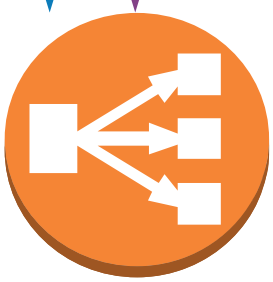
AWS

EC2/VPC Network Troubleshooting

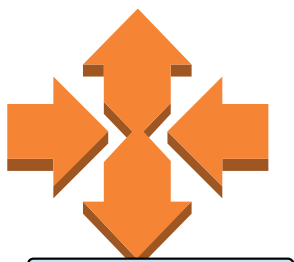


EC2 Troubleshooting

VPC Troubleshooting



ELB Troubleshooting



Auto Scaling Troubleshooting

AWS VPC Networking:

X

ACL Essentials

ACL Rules

AWS VPC Networking:

Route 53 Essentials | Hosted Zones | Record Sets

AWS VPC Networking:

X

VPC Essentials | VPC Benefits | Default VPC | VPC Limits

AWS VPC Networking:

X

CloudFront Essentials

Performance Considerations

AWS VPC Networking:

X

ELB Essentials | Classic ELB | Application ELB

AWS VPC Networking:

Subnets:

"When you create a VPC, it spans all of the Availability Zones in the region. After creating a VPC, ***you can add one or more subnets in each Availability Zone***. Each subnet ***must reside entirely*** within one Availability Zone and ***cannot span zones***." -Amazon Web Services

- Subnets MUST be associated with a route table.
- A ***PUBLIC*** subnet ***HAS*** a route to the Internet.
  - It is associated with a route table that has an IGW attached.
- A ***PRIVATE*** subnet ***does NOT have*** a route to the Internet.
  - It is associated with a route table that does NOT have an IGW attached.

- Instances launched into a ***private subnet*** can't communicate with the internet.
  - This creates a higher level of security, but it creates a limitation of an instance not being able to download software and/or updates.
  - This issue is solved by routing traffic through a NAT instance.

- By default all subnets traffic is allowed to each other available subnet within via the ***local*** target in the route table.
- A subnet is located in one specific availability zone, and does not span AZs.

NOTE: The "default" VPC already has subnets created and associated with a route table.

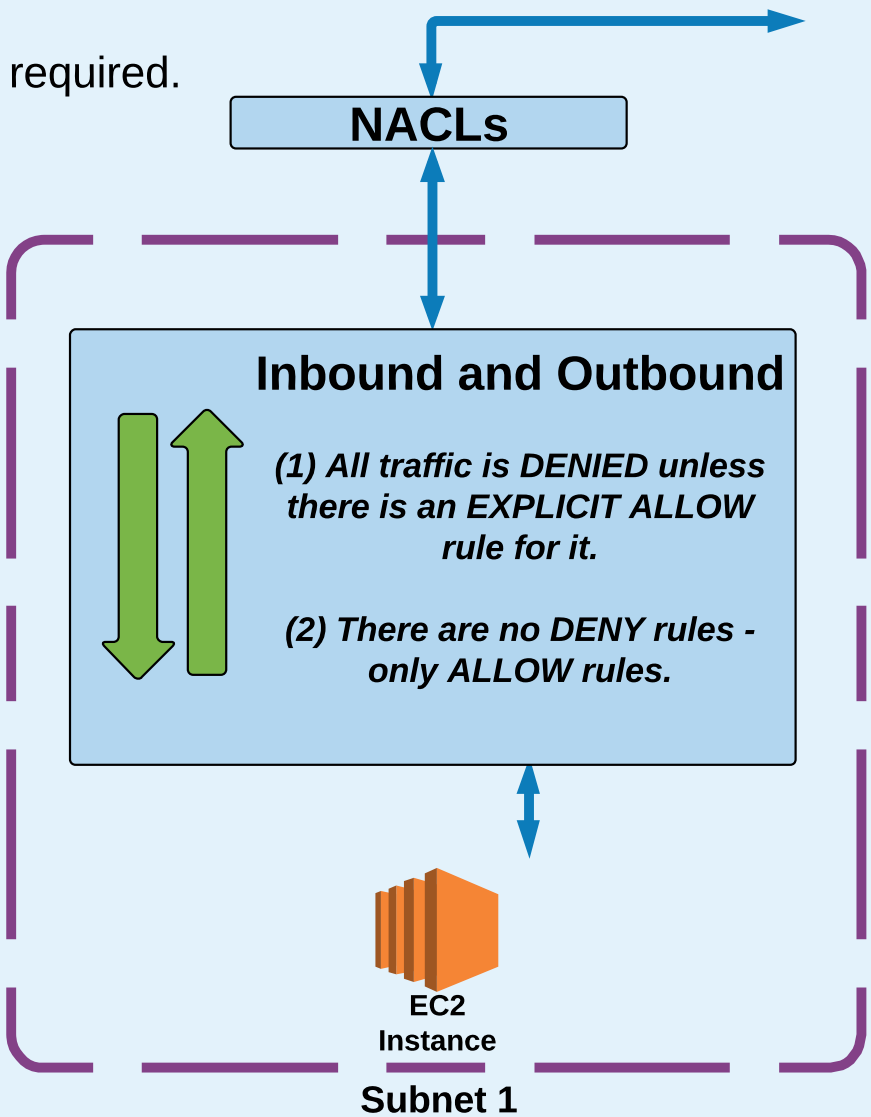
AWS VPC Networking:

Security Groups:

- Security groups are very similar to NACLs in that they ***allow/deny traffic***.
- However, security groups are security for the ***instance level*** (as opposed to the subnet level with ALCs).
- In addition, the way ***allow/deny "rules" work are different from ACLs***:
  - Security groups support only allow rules.
  - They are ***stateful***: so return traffic requests are allowed regardless of rules.
  - All rules are evaluated before deciding to allow traffic.

NOTE: Best practice is to allow ONLY traffic that is required.

Type	Protocol	Port Range	Source
SSH (22)	TCP (6)	22	0.0.0.0/0
HTTP (80)	TCP (6)	80	0.0.0.0/0



AWS VPC Networking:

Elastic Load Balancer (ELB):

- ***Load balancing*** (as a concept) is a common method used for distributing incoming traffic among servers.
- An ***Elastic Load Balancer*** is an EC2 service that automates the process of distributing incoming traffic (evenly) to all the instances that are assocaited with the ELB.
- An elastic load balancer can load balance traffic to multiple EC2 instances located across multiple availability zones.
  - This allows for highly availability and fault tolerant architecture.
- Elastic load balancing should be paired with ***Auto Scaling*** to enhance high availability and fault tolerance, AND allow for automated scalability and elasticity.
- An ELB has it's own DNS record set that allows for direct access from the open internet access.

Other important ELB facts:

- When used within a VPC, and ELB can act as an ***internal*** load balancer and load balance to internal EC2 instances on private subnets (as often done with multi-tier applications).
- ELBs will automatically stop serving traffic to an instance that becomes unhealthy (via health checks).
- An ELB can help reduce compute power on an EC2 instance by allowing for an SSL certificate to be applied directly to the elastic load balancer.

AWS VPC Networking:

Route Tables:

"A route table contains a ***set of rules***, called ***routes***, that are used to ***determine where network traffic is directed***." - Amazon web Services

- A route table's rules are comprised of two main components:
  - ***Destination***: The CIDR block range of the *target* (where the data is routed to).
  - ***Target***: A name identifier of where the data is being routed to.

- By default, all subnets traffic is allowed to each other available subnet within your VPC which is called the local route.
- You cannot modify the local route

- Unlike an IGW, you can have multiple "active" route tables in a VPC
- You cannot delete a route table if it has ***"dependancies"*** (associated subnets)

Best practice is to leave the default route table and create a new route table when new routes are needed for specific subnets.

NOTE: The "default" VPC already has a ***"main"*** route table.

Destination	Target
172.31.0.0/16	local
0.0.0.0/0	igw-95d589f2

AWS VPC Networking:

X

Auto Scaling:

- ***Auto Scaling*** is a service (and method) provided by AWS that automates the process of increasing or decreasing the number of provisioned on-demand instances available for your application.
- Auto scaling will increase or decrease the amount of instances based on chosen ***Cloudwatch*** metrics.

- For example: If your application's demand increases un-expectantly, auto scaling can automatically scale up (add instance) to meet the demand and terminate instances when the demand decreases.
  - This is known as "***elasticity***" in the AWS environment.

Auto Scaling has two main components:

- ***Launch Configuration:***
  - The EC2 "template" used when the auto scaling group needs to provision an additional instance (i.e. AMI, instance type, user-data, storage, security groups, etc)

- ***Auto Scaling Group:***
  - All the rules and settings that govern if/when an EC2 instance is automatically provisioned or terminated.
    - Number of MIN & MAX allows instances
    - VPC & AZs to launch instances into
    - If provisioned instances should receive traffic from a ELB
    - Scaling policies (cloudwatch metrics thresholds that trigger scaling)
    - SNS notifications (to keep you informed when scaling occurs)

NOTE: ***For architecture to be considered highly available and fault tolerant - it MUST have and ELB serving traffic to and ASG with a MIN of two instances located in separate availability zones.***

AWS Networking:

X

S3 for DNS Failover:

- By using a failover routing policy in a Route 53 DNS record set, an S3 bucket can be used as a failover endpoint.
- This can provide an extremely reliable backup solution if your primary endpoint fails.
- And even though S3 should only be used for static web hosting, it gives you the opportunity to provide your users with some type of information until the primary endpoint is working again.
- An S3 bucket can also be used as a primary endpoint, if you just want to host a simple static site.

Note: For a DNS record to use an S3 bucket as an endpoint, the bucket name MUST be the same as the domain name.

AWS VPC Networking:

X

Bastion Host:

- A ***Bastion Host*** is an EC2 instance that lives in a public subnet, and is used as a "gateway" for traffic that is destined for instances that live in private subnets.
- This means that we can use a bastion host as a "portal" to access EC2 instances that are located in a private subnet.
- A bastion host is considered the "***critical strong point***" of the network - as all traffic must pass through it first.
- A bastion host should have increased and extremely tight security (usually with extra 3rd party security and monitoring software installed).
- A bastion host can be used as an access point to "ssh" into an internal network (to access private resources) without a VPN (virtual private network).

"A system identified by the firewall administrator as a critical strong point in the network's security. Generally, bastion hosts will have some degree of extra attention paid to their security, may undergo regular audits, and may have modified software" – Marcus J. Ranum .

AWS VPC Networking:

X

VPC Peering Essentials:

VPC Peering enables the ability to create a direct network route between one VPC and another. This allows the sharing of resources between two subnets as if it was on the same network. Basically, at a high level it creates a link between the two.
VPC Peering can occur between other AWS accounts and other VPCs within the same region

§VPC Peering connections cannot occur between two regions

Scenarios
Peering two VPCs – Company runs multiple AWS accounts and you need to link all the resources as if they were all under one private network (assuming resources in the same region)
Peering To A VPC – Multiple VPCs can connect to a central VPC but cannot communicate with each other, only communication can occur between the peered VPC and the primary. This use case could be if a third party was sharing a resource that the customers needed to connect to. (File sharing, Customer Access, Active Directory)

AWS VPC Networking:

X

Internet Gateway:

- Is a VPC component that ***allows communication between instances in your VPC and the Internet***.
- Is a horizontally scaled, ***redundant and highly available.***
- It imposes no availability risks or bandwidth constraints on your network traffic.
- Provides NAT translation for instances that have a public IP addresses assigned (public IP to private IP).

NOTE: Your "default" VPC already has an IGW ***attached***.

Internet Gateway rules and details you need to know:

- Only 1 IGW can be attached to a VPC at a time.
- An IGW cannot be detached from a VPC while there are active AWS resources in the VPC (such as an EC2 instance or RDS Database)
- An IGW must be attached to a VPC if the resources inside the VPC need to connect to resources via the open internet.

"To enable access to or from the internet for instances in a VPC subnet, you must attach an Internet gateway to your VPC, ensure that your subnet's route table points to the Internet Gateway, ensure that instances in your subnet have a public IP address or Elastic IP address, and ensure that your network access control and security group rules allow the relevant traffic to and from your instance" – AWS

AWS VPC Networking:

NAT Gateway:

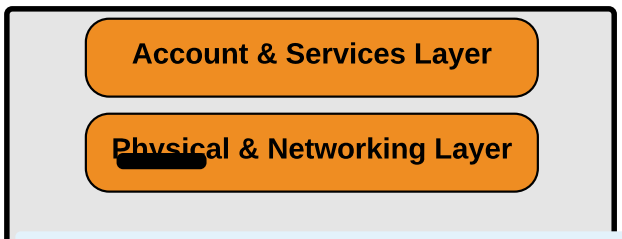
- A ***NAT Gateway*** is designed to provide EC2 instances that live in a private subnet with a route to the internet (so they can download software packages and updates).
- A NAT Gateway will prevent any hosts located outside of the VPC from initiating a connection with instances that are associated with it.
- A NAT Gateway will only allow incoming traffic through if a request for it originated from an instance in a private subnet.
- A NAT Gateway is needed because instances launched into private subnets can't communicate with the open internet.
- Placing instances in a private subnet creates a higher level of security, but also creates the limitation of the instances not being able to download software and software updates.

A NAT Gateway MUST:

- Be created in a public subnet.
- Be part of the private subnets route table.

NAT Instance:

- A NAT Instance is identical to a NAT gateway in its purpose.
- However, it is executed differently by configuring an actual EC2 instance to do the same job.
- A NAT Instance is starting to become more of a legacy feature in AWS.
- However, questions about them may still appear on the exam.



Virtual Private Cloud (VPC) Essentials:

"Amazon Virtual Private Cloud (Amazon VPC) enables you to launch Amazon Web Services (AWS) resources into a virtual network that you've defined. This virtual network closely resembles a traditional network that you'd operate in your own datacenter, with the benefits of using the scalable infrastructure of AWS" – Amazon Web Services

A VPC is designed to resemble:

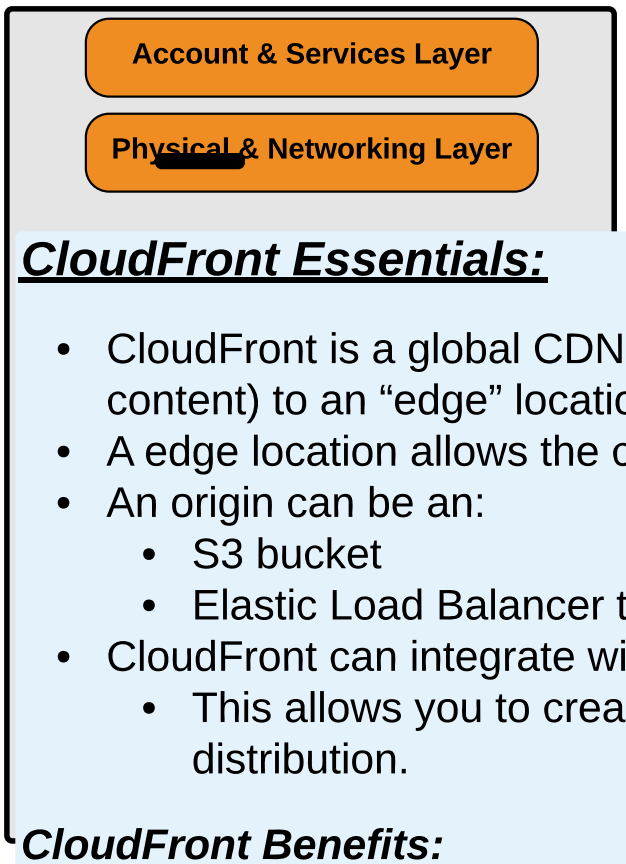
- Private on-premise data centers
- Private corporate network

Private network features available in AWS VPCs:

- Private and Public subnets
- Scalable architecture
- Ability to extend corporate/on-premise network to the cloud as if it was part of your network (VPN)

Important VPC Facts:

- A VPC is housed within a chosen AWS region.
- A VPC spans multiple availability zones within a region.
  - This allows you to provision redundant resource in separate availability zones while having them accessible on the same network (foundation of high availability and fault tolerant architecture).
- AWS provides a DNS server for your VPC so each instance has a hostname. However, you can run your own DNS servers by changing the DHCP option set configuration within the VPC.



CloudFront Essentials:

- CloudFront is a global CDN which delivers content from an “origin” location (the source of the content) to an “edge” location (AWS CDN data center).
- A edge location allows the caching of static objects from the origin location.
- An origin can be an:
  - S3 bucket
  - Elastic Load Balancer that distributes requests among origin EC2 instances.
- CloudFront can integrate with Route 53 for “alternate” CNAMEs.
  - This allows you to create a URL such as http://cdn.mydomain.com that works with your distribution.

CloudFront Benefits:

- Users experience lower latency and content load time.
- Reduces load on your applications resources (origin services) - thus reducing cost.

Updating Cached Files:

- Caching is done based off the object name.
- 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 off the object name.
- "Invalidations" have a cost, so if you have to invalidate a large CloudFront distribution then perhaps you should just create a new distribution and move DNS names.
- Cached objects can also be set with a specific expiration time/date, or set to not cache at all.

Signed URLs:

- Signed URLs allow access to “private content” by creating a temporary, one-time-use URL based off of the number of seconds you want it to be accessible.
- Signed with a X.509 certificate.

Account & Services Layer

Physical & Networking Layer

Elastic Load Balancer (ELB) Essentials:

- ***Load balancing*** (as a concept) is a common method used for distributing incoming traffic among servers.
- An ***Elastic Load Balancer*** is an EC2 service that automates the process of distributing incoming traffic (evenly) to all the instances that are assocaited with the ELB.
- An elastic load balancer can load balance traffic to multiple EC2 instances located across multiple availability zones.
  - This allows for highly availability and fault tolerant architecture.
- Elastic load balancing should be paired with ***Auto Scaling*** to enhance high availability and fault tolerance, AND allow for automated scalability and elasticity.
- An ELB has it's own DNS record set that allows for direct access from the open internet access.

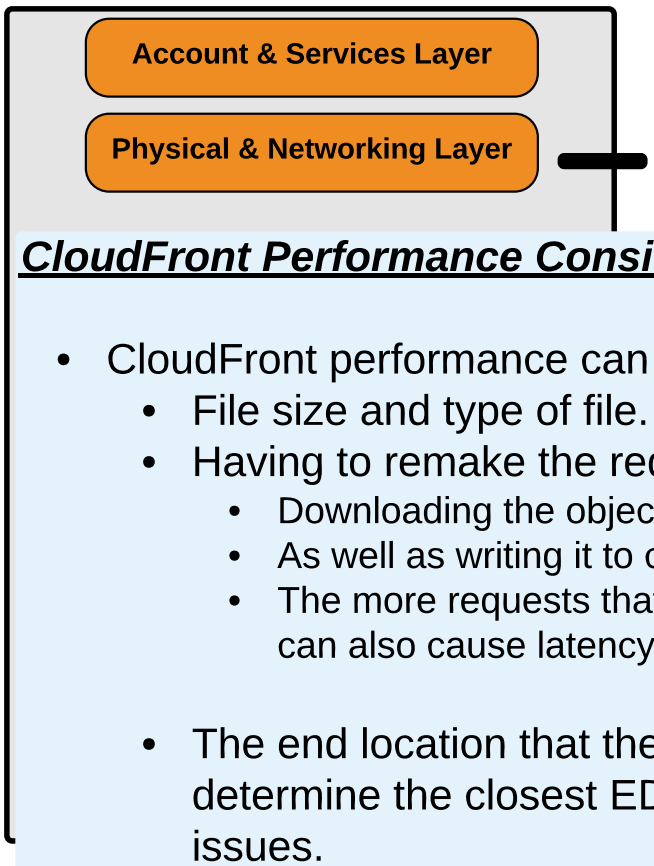
Other important ELB facts:

- When used within a VPC, and ELB can act as an ***internal*** load balancer and load balance to internal EC2 instances on private subnets (as often done with multi-tier applications).
- ELBs will automatically stop serving traffic to an instance that becomes unhealthy (via health checks).
- An ELB can help reduce compute power on an EC2 instance by allowing for an SSL certificate to be applied directly to the elastic load balancer.

On-premises Data Center

On-Premises Servers

Hybrid Environments



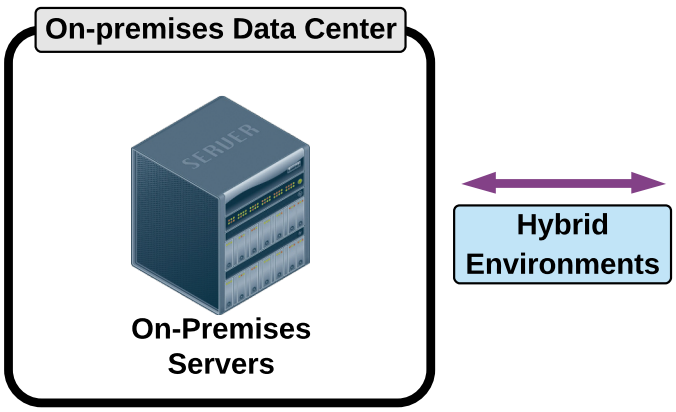
CloudFront Performance Considerations:

- 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.

  - The end location that the user's request goes to is dependent upon a "DNS check" to determine the closest EDGE location. So slow DNS issues can cause performance issues.

  - Query strings (request to the origin to serve a specific object) reduce cache “hits”:
    - http://cdn.linuxacademy.com/?querythis=querythat
    - 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 increases performance (less frequent request to the source).

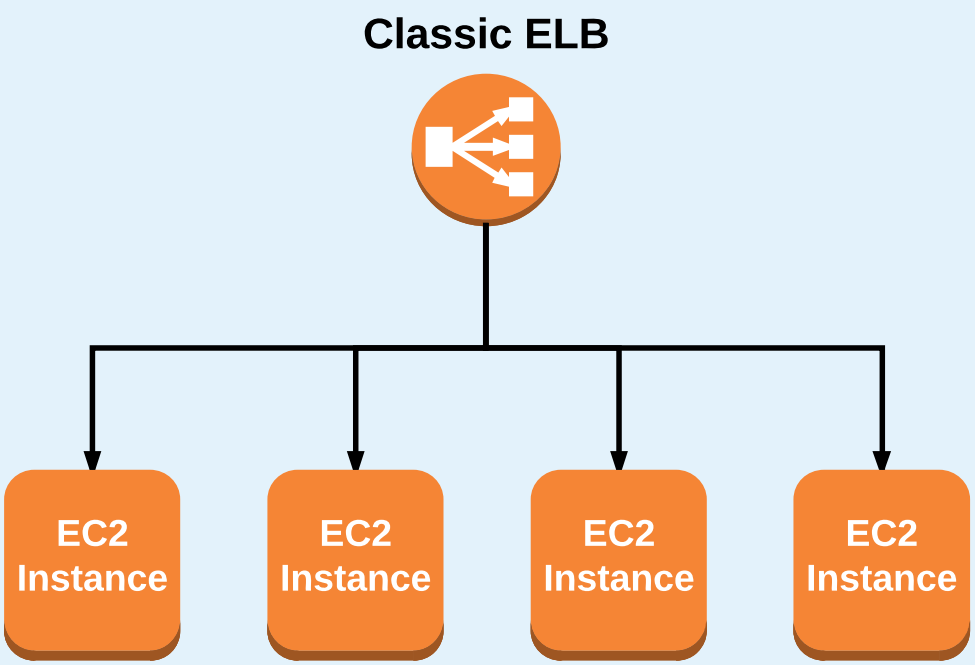


Account & Services Layer

Physical & Networking Layer

Classic Elastic Load Balancer:

- A "classic" elastic load balancer is designed for ***simple*** balancing of traffic to multiple EC2 instances.
- There are no granular routing "rules" - all instances get routed to evenly, and no special routing request can be made based on specific content request from the user.
- Classic load balancing is best used when all instances (that are being served traffic) contain the same data.



(all instances have the same content)

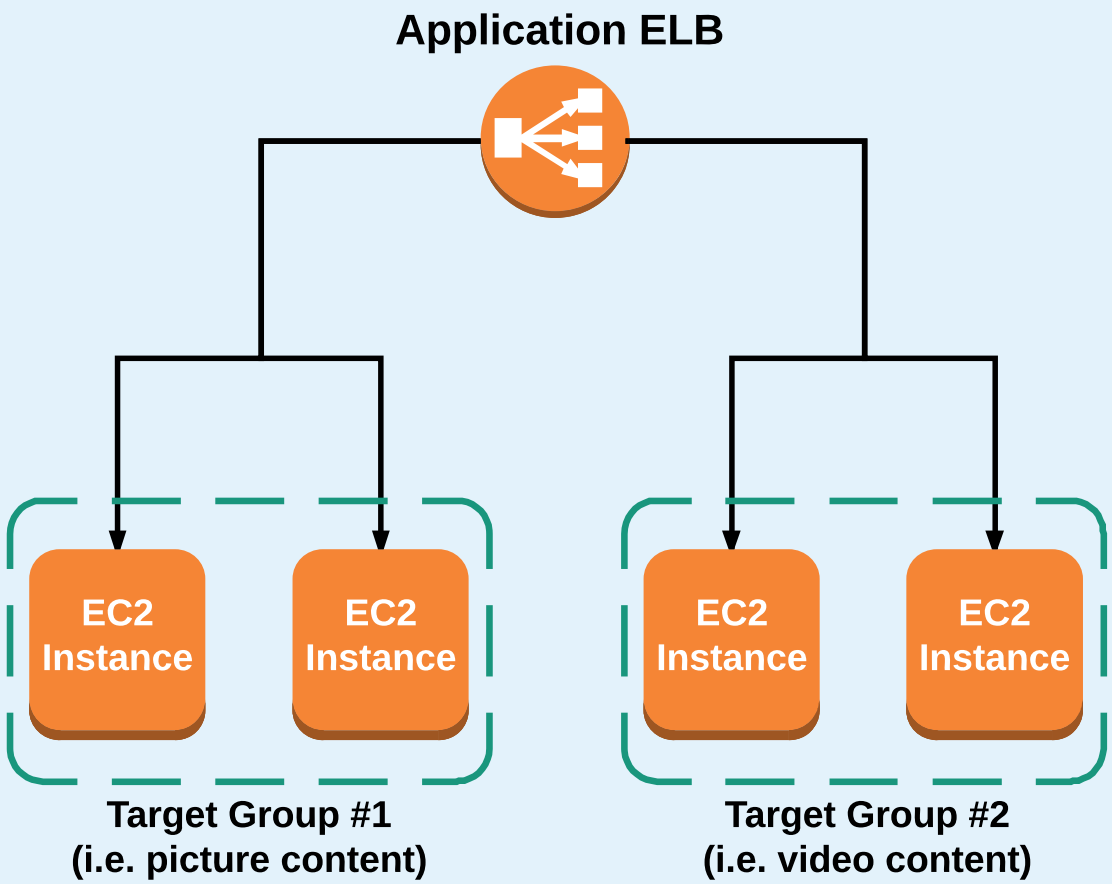
On-Premises Servers

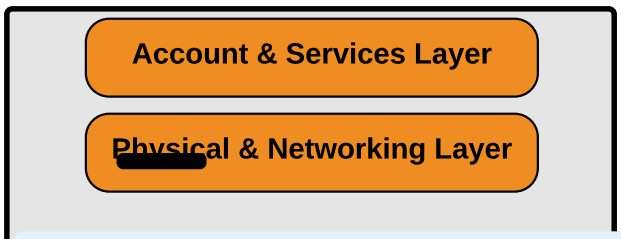
Account & Services Layer

Physical & Networking Layer

Application Elastic Load Balancer:

- An "Application" elastic load balancer is designed for ***complex*** balancing of traffic to multiple EC2 instances using ***Content-based "rules"***.
- Content-based rules (setup 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 ULR 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"***.
- An Application ELB also supports ECS Containers, HTTPS, HTTP/2, WebSockets, Access Logs, Sticky Sessions, and AWS WAF (Web Application Firewall).





Network Access Control List (ACLs) Essentials:

- ACLs Operate at the network/subnet level.
- They support ***allow*** AND ***deny*** rules for traffic traveling into or out of a subnet.
- They are ***stateless***: so return traffic must be allowed through an ***outbound rule***.

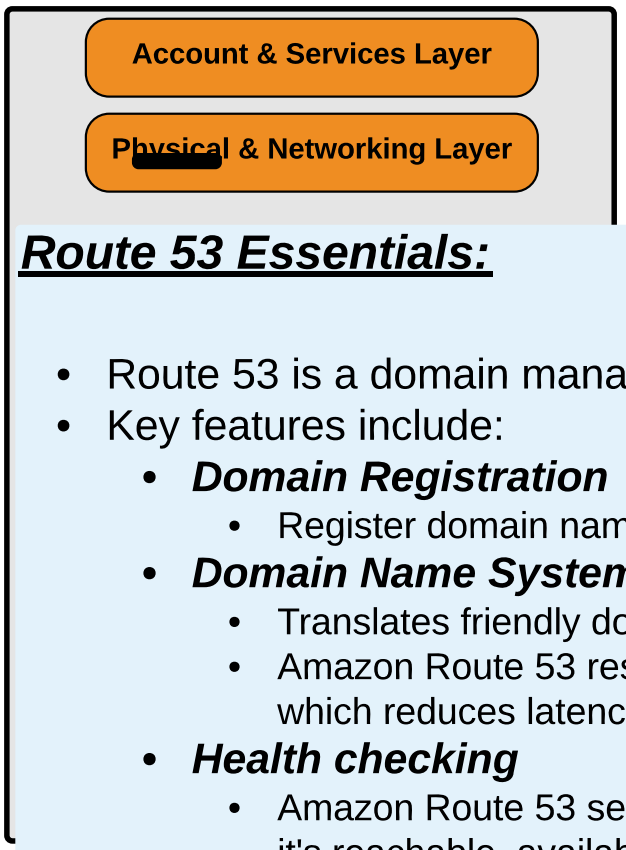
- They process rules in number order when deciding whether to allow traffic.
- Rules are evaluated in order, starting with the lowest rule number - for example:
  - If traffic is denied at a lower rule number and allowed at a higher rule number, the allow rule will be ignored and the traffic will be denied.
- The last rule in every ACL is a "catch all" deny rule.
  - This means that unless a protocol/port is explicitly allowed, it will be denied.

- A network access control list (NACL) is an ***optional layer of security*** for your VPC that acts as a ***firewall*** for controlling traffic in and out of one or more ***subnets***.

- ***Best practice*** to increment numbers by 10 so if you have to place in a rule in a certain order it does not create an issue

NOTE: Your "default" VPC already has a NACL and it is associated with the default subnets.

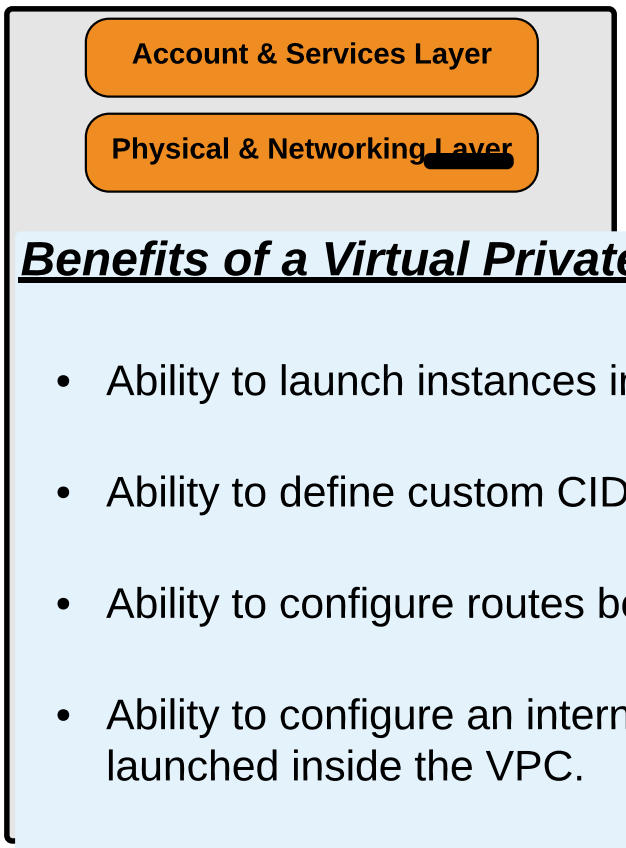
Rule #	Type	Protocol	Port Range	Source	Allow / Deny
80	SSH (22)	TCP (6)	22	0.0.0.0/0	ALLOW
90	HTTP (80)	TCP (6)	80	0.0.0.0/0	ALLOW
*	ALL Traffic	ALL	ALL	0.0.0.0/0	DENY



Route 53 Essentials:

- Route 53 is a domain management service (DNS hosting solution) provided by AWS.
- Key features include:
  - ***Domain Registration***
    - Register domain names, such as orionpapers.com
  - ***Domain Name System (DNS) service***
    - Translates friendly domains names like www.orionpapers.com into IP addresses like 192.0.2.1.
    - Amazon Route 53 responds to DNS queries using a global network of authoritative DNS servers, which reduces latency.
  - ***Health checking***
    - Amazon Route 53 sends automated requests over the Internet to your application to verify that it's reachable, available, and functional.

- Route 53 can manage external DNS for domain routing - routing request for www.orionpapers.com to the proper AWS resources such as a CloudFront distribution, ELB, EC2 instance, or RDS server.
- Route 53 is commonly used with an ELB to direct traffic from the domain to the ELB (and thus have traffic evenly distributed among servers running your application).
- Route 53 can also be used to manage internal DNS for custom internal hostnames within a VPC as long as the VPC is configured for it.
- Latency, GEO, basic, and failover routing policies allow for region-to-region fault tolerant architecture design.
- You can easily configure for failover to S3 (if website bucket hosting is enabled) or CloudFront.



Benefits of a Virtual Private Cloud (VPC):

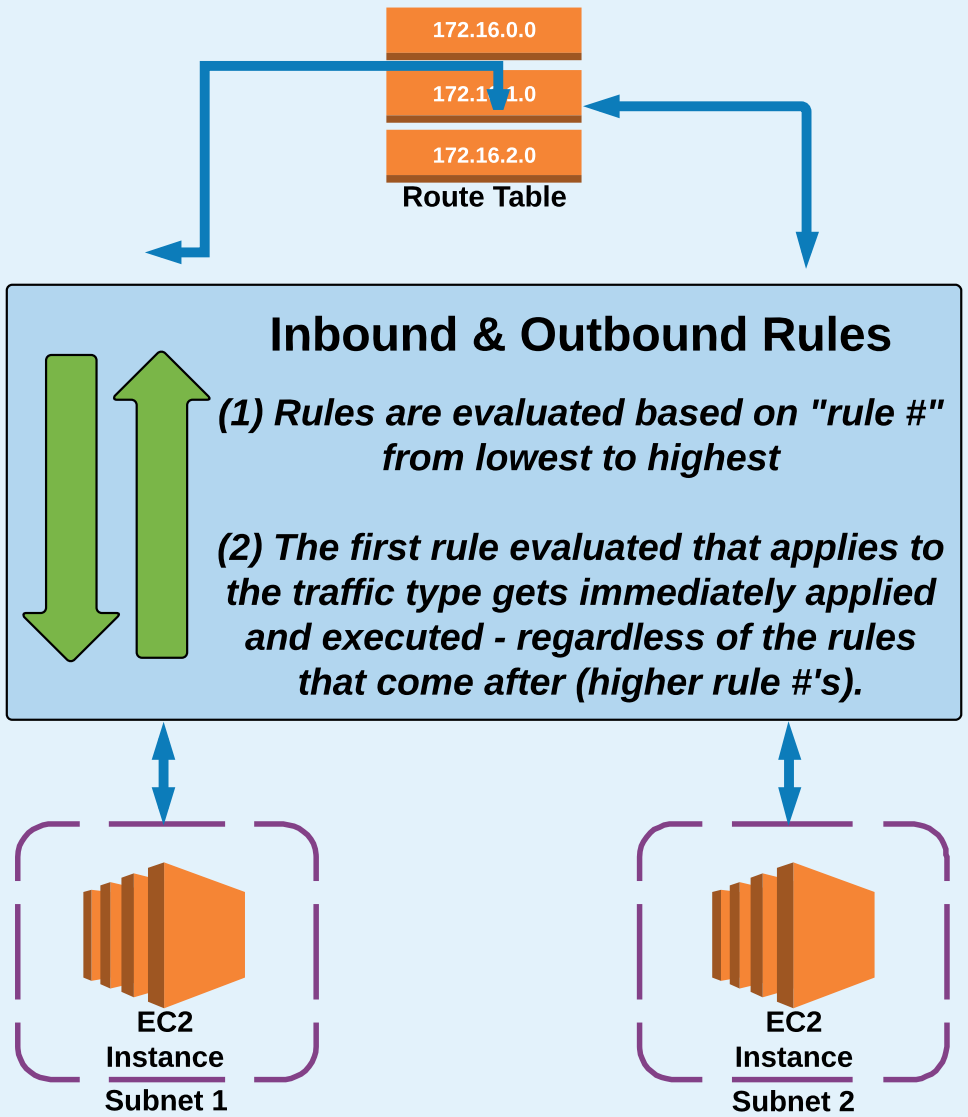
- Ability to launch instances into a subnet.
- Ability to define custom CIDR (IP address range) inside each subnet.
- Ability to configure routes between subnets via route tables.
- Ability to configure an internet gateway to provide a route to the internet for resources launched inside the VPC.
- Ability to create a layered network of resources.
- Ability to extend your on-premise network into the cloud with VPN/VPG and an IPsec VPN tunnel.
- Layered Security:
  - Instance level Security Groups (firewall on the instance level)
  - Subnet level network ACLs (firewall on the subnet level)

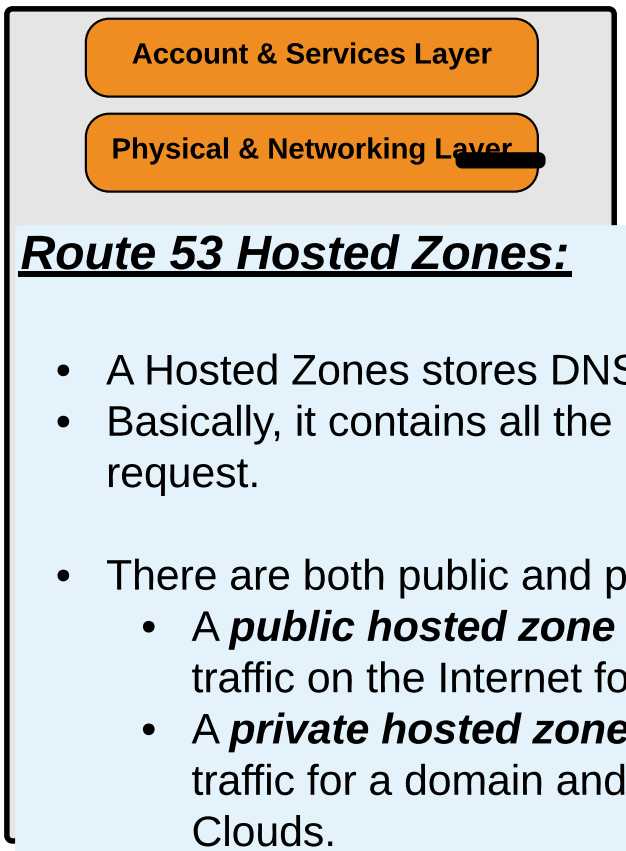
Account & Services Layer

Physical & Networking Layer

Network Access Control List (ACLs) Rules:

- Rules are evaluated from **lowest to highest based on "rule #"**.
- The first rule found that applies to the traffic type is immediately applied, regardless of any rules that come after it (have a higher "rule #").
- A subnet can only be associated with ONE NACL as a time.
- An NACL allows or denies traffic from entering a subnet. Once inside the subnet, other AWS resources (i.e. EC2 instances) may have an additional layer of security (security groups).



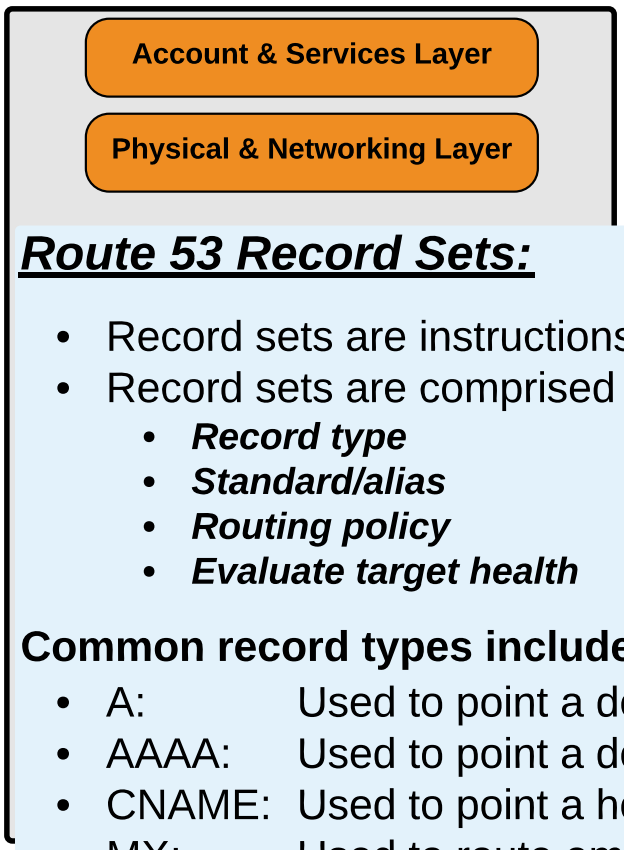


Route 53 Hosted Zones:

- A Hosted Zones stores DNS records for your domain.
- Basically, it contains all the rules (record sets) that tells Route 53 what do to with DNS request.

- There are both public and private hosted zones:
  - A ***public hosted zone*** is a container that holds information about how you want to route traffic on the Internet for a domain, such as linuxacademy.com, and its subdomains.
  - A ***private hosted zone*** is a container that holds information about how you want to route traffic for a domain and its subdomains within one or more Amazon Virtual Private Clouds.

- After you create a hosted zone for your domain, such as orionpapers.com, you create resource record sets to tell the Domain Name System (DNS) how you want traffic to be routed for that domain.
- Hosted zones come pre-populated with NS (name server) and SOA (start of authority) record sets.



Route 53 Record Sets:

- Record sets are instructions that actually match domain names to IP addresses.
- Record sets are comprised of various options, including:
  - ***Record type***
  - ***Standard/alias***
  - ***Routing policy***
  - ***Evaluate target health***

Common record types include:

- A: Used to point a domain to an IPv4 IP address.
- AAAA: Used to point a domain to an IPv6 IP address.
- CNAME: Used to point a host/name to another host/name.
- MX: Used to route email (mail exchange).

Alias Record Sets:

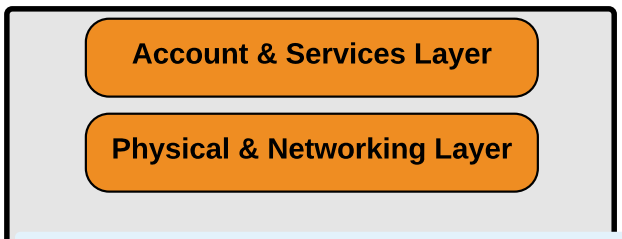
- Instead of an IP address (standard record sets), an ***alias*** record set contains a pointer to an AWS specific resource, such as:
  - An elastic load balancer
  - CloudFront distribution
  - Elastic Beanstalk environment
  - Amazon S3 bucket that is configured as a static website

Routing Policy:

- Simple: Route all traffic to one endpoint
- Weighted: Route traffic to multiple endpoints (manual load balancing)
- Latency: Route traffic to an endpoint based on the users latency to various endpoints
- Failover: Route traffic to a "secondary" endpoint if the "primary" is unavailable.
- Geolocation: Route traffic to an endpoint based on the geographical location of the user.

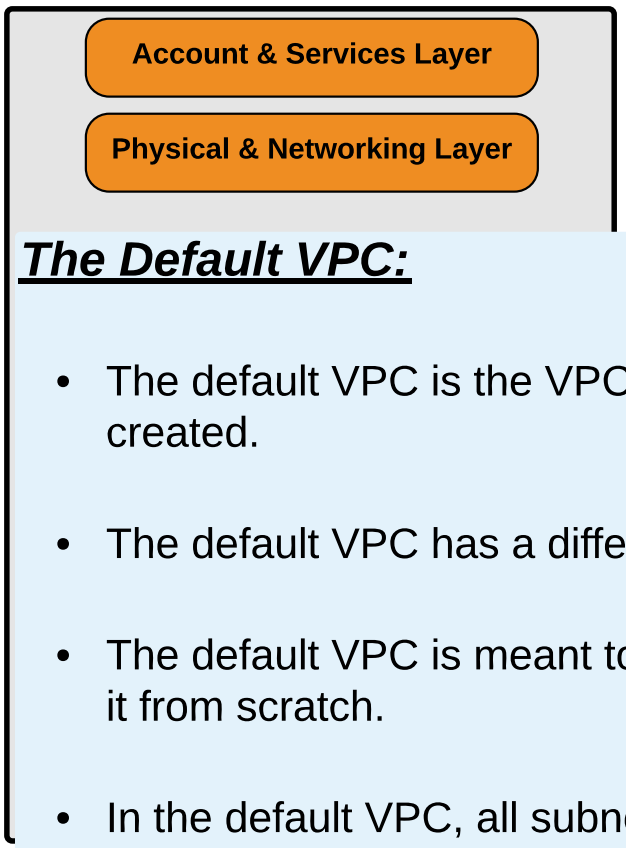
Evaluate Health Check:

- Can monitor the health of your application and trigger an action.



VPC Limits:

- 5 VPCs per region (more available upon request)
- 5 internet gateways per region (this is equal to your VPC limit because you can only have one internet gateway attached to a VPC at a time)
- 50 customer gateways per region
- 50 VPN connections per region
- 200 route tables per region / 50 entries per route table
- 5 elastic IP addresses
- 500 security groups per VPC
- 50 rules per security group
- 5 security groups per network interface (security groups although generally referred to as being on the on the instance level are technically on the VPC level)



The Default VPC:

- The default VPC is the VPC that comes preconfigured in your AWS account when it is first created.
- The default VPC has a different setup than a non-default VPCs.
- The default VPC is meant to allow the user easy access to a VPC without having to configure it from scratch.
- In the default VPC, all subnets have a route to the internet via route table and an attached IGW.
- Each instance launched in the default VPC (by default) has a private and public IP address (defined on the subnet settings).

AWS Networking:

X

CloundFront:

-

AWS Networking:

EC2 Troubleshooting:

Connectivity issues to an EC2 instance

- Correct ports on the security group are may not be open.

Cannot attach and EBS volume to an EC2 instance

- EBS volumes must live in the same availability zone as the EC2 instance they are to be attached to.
- You can create a snapshot from the volume and launch the volume in the correct availability zone.

Cannot launch additional instances

- You have probably reached the EC2 limit and need to contact AWS to increase limit.

Unable to download package updates

- The EC2 instance may not have a public/Elastic IP address, and/or does not belong to a public subnet.

Applications seeming to slow down on T2 micro instances

- T2 micro instances utilize CPU credits (for "burstable" processing), so chances are your application is using too much processing power and needs a larger instance or different instance type.

AMI unavailable in other regions

- AMI's are only available in the regions that they are created.
- An AMI can be copied to another region but will receive a new AMI id.

"Capacity error" when attempting to launch an instance in a placement group

- Start and stop all the instances in the placement group (AWS tries to locate them as close as possible).

AWS Networking:

VPC Troubleshooting:

New EC2 instances are not automatically being assigned a public IP address

- Modify the Auto-Assign Public IP setting on the subnet.

NAT Gateway is configured but instances inside a private subnet still cannot download packages

- Need to add 0.0.0.0/0 route to the NAT gateway on the route table for private subnets.

Traffic is not making it to the instances even though security group rules are correct

- Check the Network Access Control Lists to ensure the proper ports from the proper sources are open. (also check you IGW and route table settings)

Error when attempting to attach multiple internet gateways to a VPC

- Only one internet gateway can be attached to a VPC at any given time.

Error when attempting to attach multiple Virtual Private gateways to a VPC

- Only one Virtual Private Gateway can be attached to a VPC at any given time.

VPC Security group (for EC2 instances) does not have enough rules for the required application

- Assign the EC2 instance to multiple security groups.

Cannot SSH/communicate with resources inside of a private subnet

- Either you have not setup a VPN, or you have not connected to an EC2 instance (Bastion host) within the VPC to launch a connection from.

Successful site-to-site VPN connection but unable to access extended resources

- Need to add on-premise routes to the Virtual Private Gateway route table.

Failure to create a VPC peering connection between two VPC's in different regions

- Peering connections can only be created between two VPC's in the same region.

AWS Networking:

Auto Scaling Troubleshooting:

An Auto Scaled instance continues to start and stop (or create/terminate) in short intervals

- The scale-up and scale-down thresholds may be too close to each other. Either raise the scale-up threshold or lower the scale-down threshold.

Auto Scaling does not occur even though scaling policies are configured correctly

- The “max” number of instances set in the auto scaling group may have been reached.

Please note: that there are more troubleshooting considerations that are outside the scope of the AWS CSA. However, they are covered in the AWS SysOps Certification and training course at the Linux Academy.

AWS Networking:

ELB Troubleshooting:

Load balancing is not occurring between instances in multiple availability zones

- Make sure "Enable Cross-Zone load balancing" has been selected.

Instances are healthy but are not registering as healthy with the ELB

- Check configuration for the "health check" to make sure you have selected the proper ping protocol, ping port, and ping path.

The ELB is configured to listen on port 80, but traffic is not making it to the instances that belong to the ELB

- You may have mistaken the "Listeners" for the security group. Listeners are not the same as the security group rules, port 80 still needs to be open on the security group that the ELB is using.

Access logs on web servers show IP address of the ELB not the source traffic

- Enable Access Logs to Amazon S3 (found under "attributes")

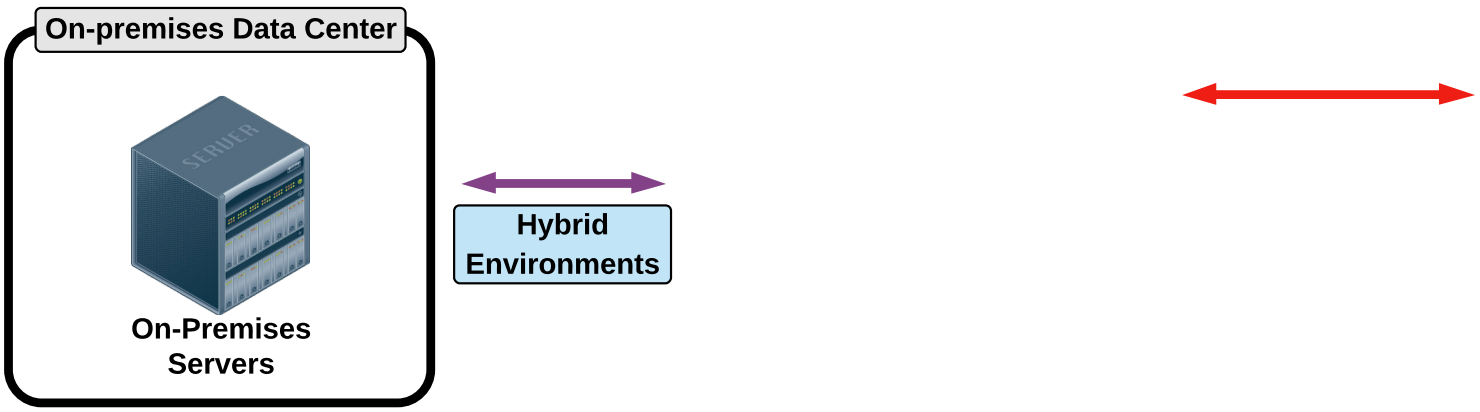
Unable to add instances from a specific subnet to the ELB

- Most likely the subnet that the instance lives in has not been added to the ELBs configuration.

AWS Networking:

VPC Peering Essentials:

- VPC peering is used to extend your ***private network*** from one VPC, or one subnet, or specifically one instance, to another VPC.
- This is for sharing ***internal*** resources, via private IP addresses.
- VPC peering can only occur between two VPCs that are in the same region.
- You cannot configure VPC peering between VPCs in two different regions.
- You can however configure VPC peering between two VPCs in different accounts - but only if they are in the same region.
- To peer VPCs, they must have separate (non-overlapping) CIDR block ranges.
- Transitive connections are not allowed.
- You can configure the peering to connect the entire VPC, or just specific subnets.



AWS Networking:

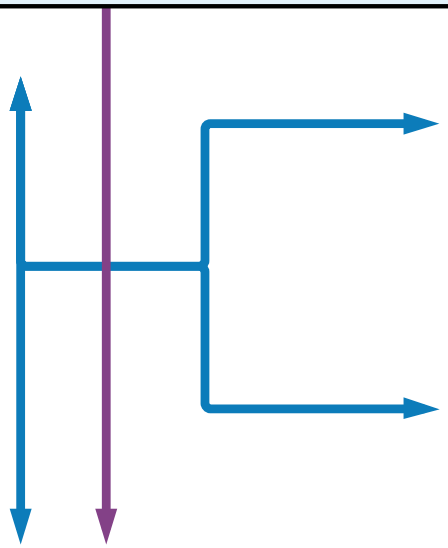
X

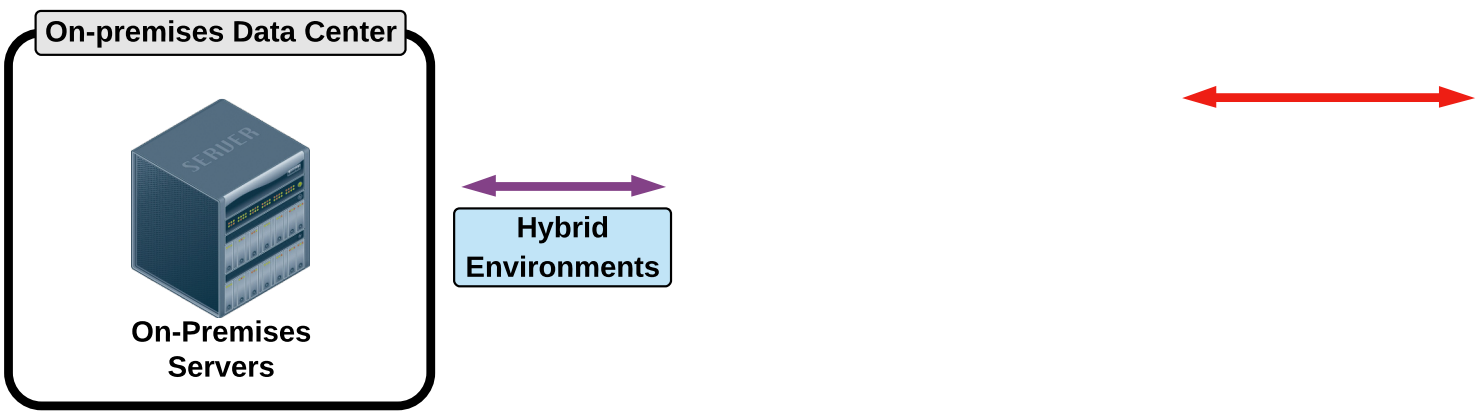
Edge Location:

- An ***Edge Location*** is an AWS datacenter which does not contain AWS services.
- Instead, it is used to deliver content to parts of the world.

- An example would be ***CloudFront,*** which is a CDN:
  - Cached items such as a PDF file can be cached on an edge location which reduces the amount of "space/time/latency" required for a request from that part of the world.

Appendix





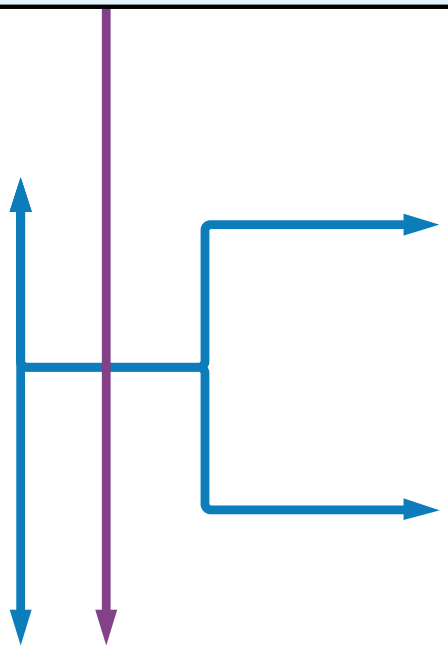
AWS Networking:

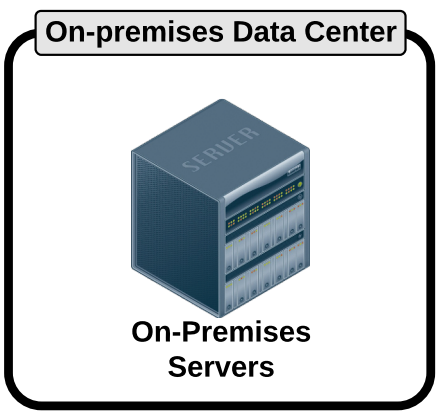
X

CloudFront Origin:

- An “origin” location is the source of the content (static objects).
- An origin can be an:
  - S3 bucket
  - Elastic Load Balancer that distributes requests among origin EC2 instances.

Appendix





Account & Services Layer

Physical & Networking Layer

Appendix

On-premises Data Center

On-Premises Servers

On-Premises Servers

Account & Services Layer

Physical & Networking Layer

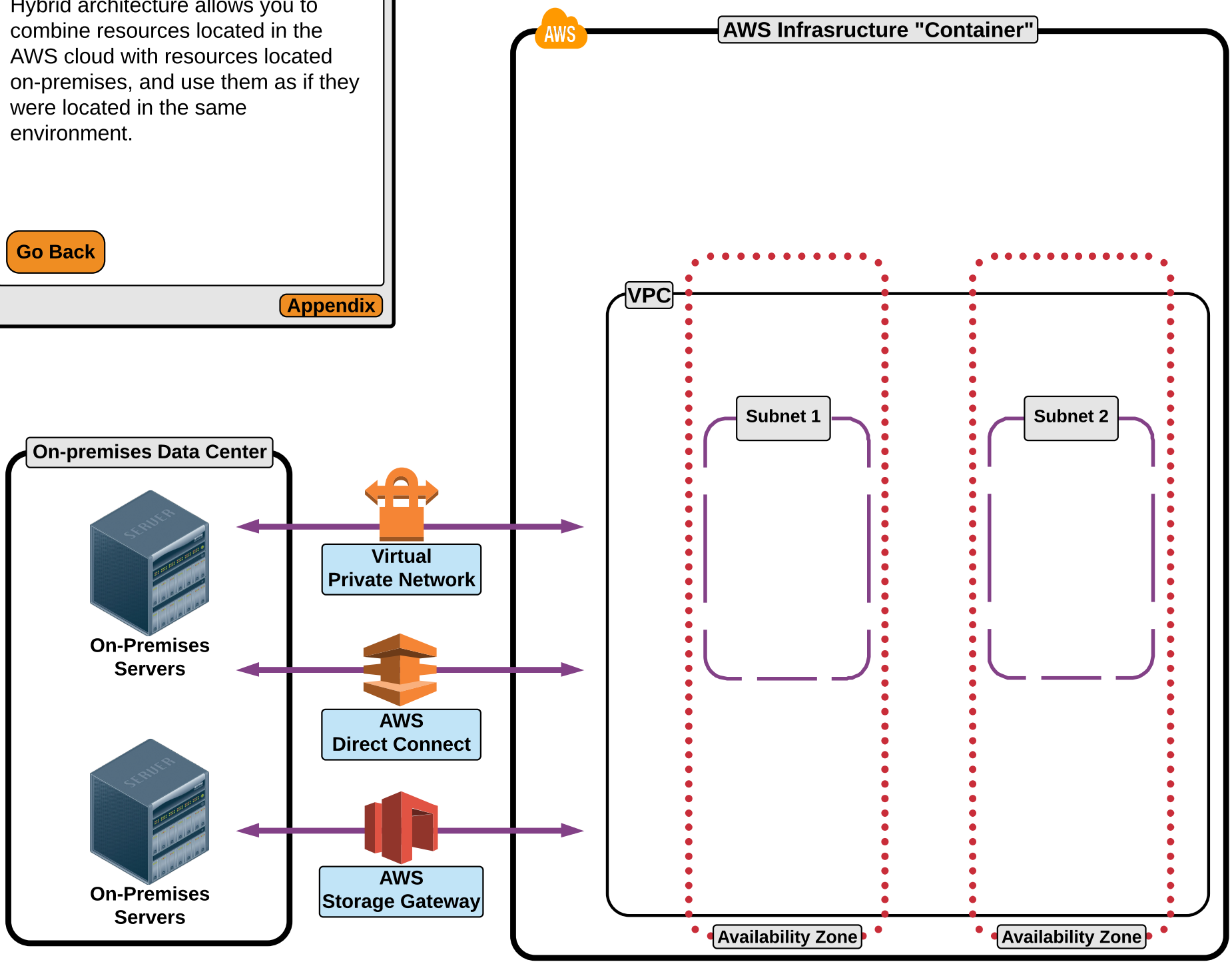
AWS Physical & Networking Layer

(Hybrid Environments)

Hybrid architecture allows you to combine resources located in the AWS cloud with resources located on-premises, and use them as if they were located in the same environment.

Go Back

Appendix



Account & Services Layer

Physical & Networking Layer

AWS Physical & Networking Layer

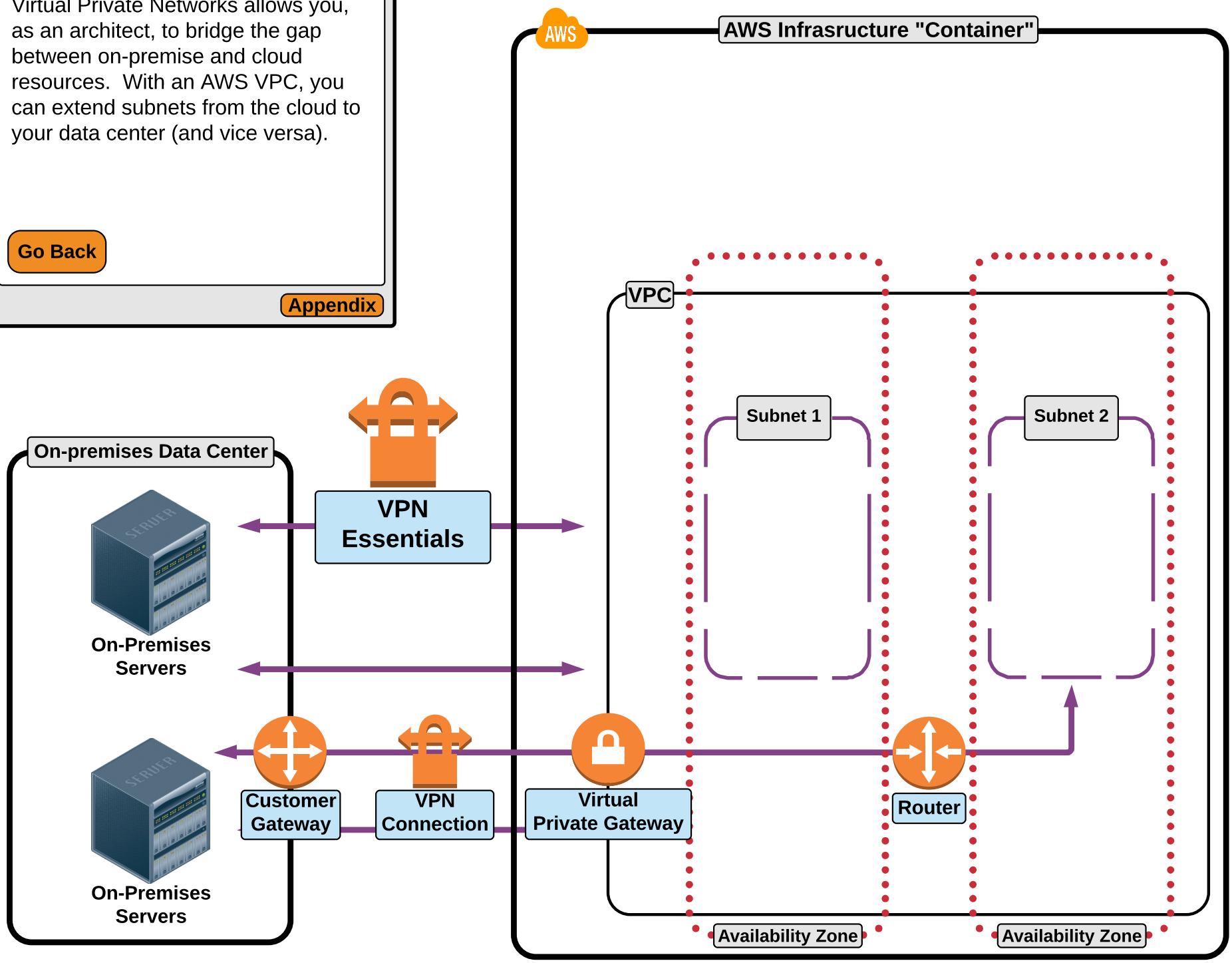
(Virtual Private Networks)

Virtual Private Networks allows you, as an architect, to bridge the gap between on-premise and cloud resources. With an AWS VPC, you can extend subnets from the cloud to your data center (and vice versa).

Go Back

Appendix

Site-to-site Example



Account & Services Layer

Physical & Networking Layer

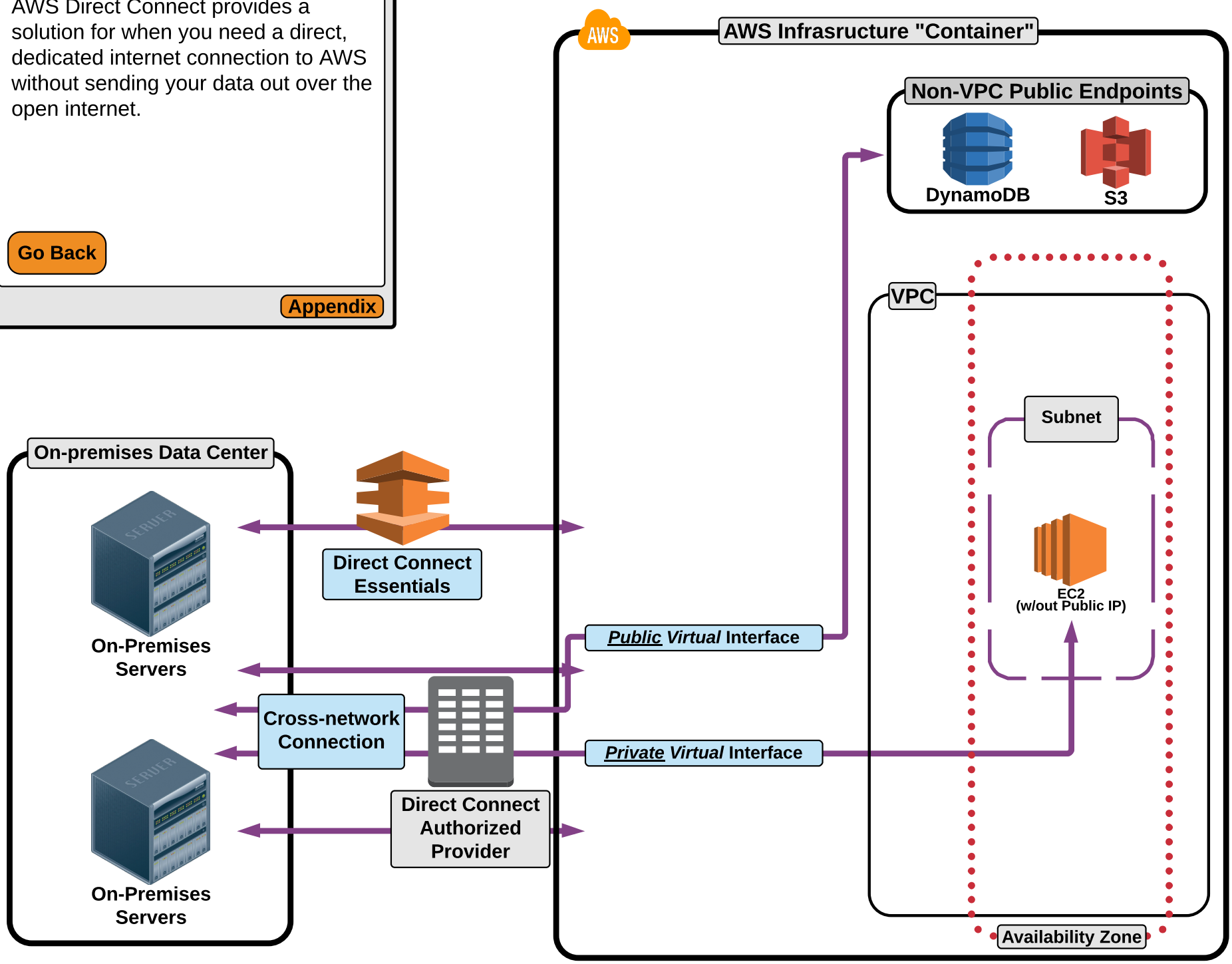
AWS Physical & Networking Layer

(AWS Direct Connect)

AWS Direct Connect provides a solution for when you need a direct, dedicated internet connection to AWS without sending your data out over the open internet.

Go Back

Appendix



Account & Services Layer

Physical & Networking Layer

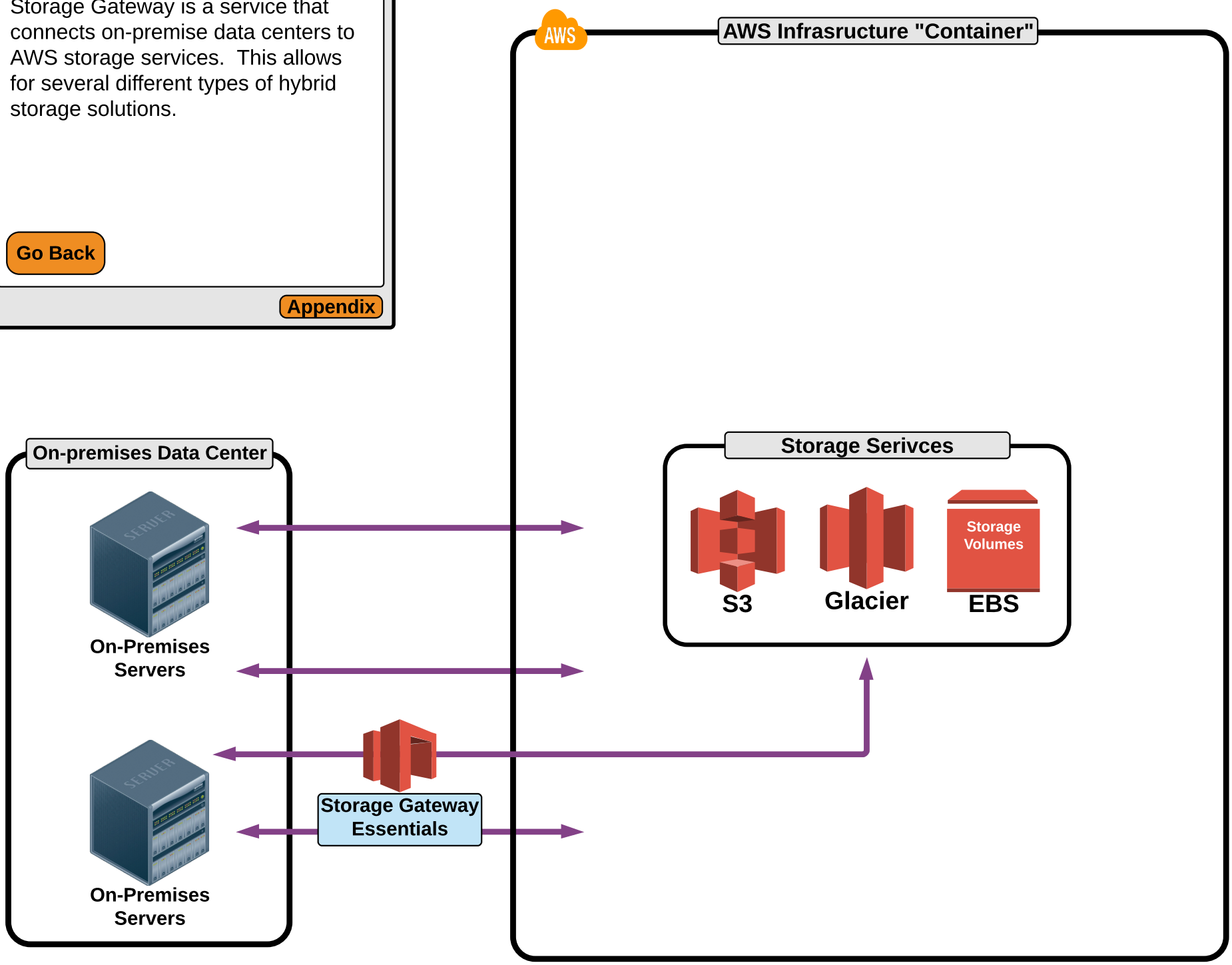
AWS Physical & Networking Layer

(AWS Storage Gateway)

Storage Gateway is a service that connects on-premise data centers to AWS storage services. This allows for several different types of hybrid storage solutions.

Go Back

Appendix



Virtual Private Network (VPN):

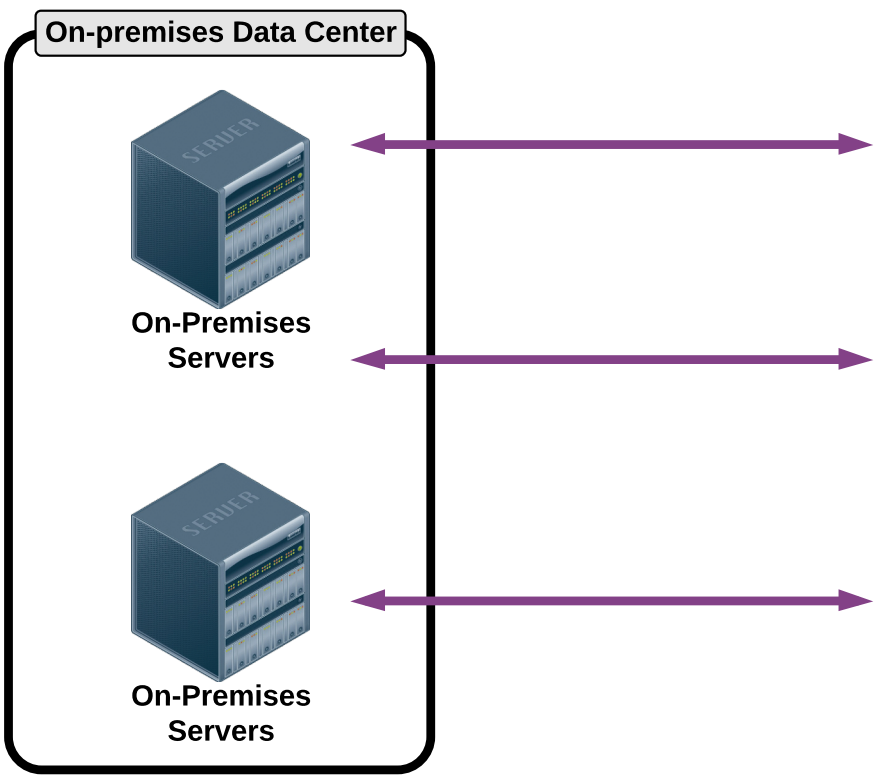
X

Customer Gateways:

- A ***customer gateway*** is a physical device or software application at the on-premise location that acts as the “connector” to the VPN connection.
- In your AWS account, the ***customer gateway component*** is where you configure the public IP (internet routable static IP) address of the physical device or software application at the on-premise location.

Note: Both a VPG and a Customer Gateway are required to establish a VPN connection.

Appendix



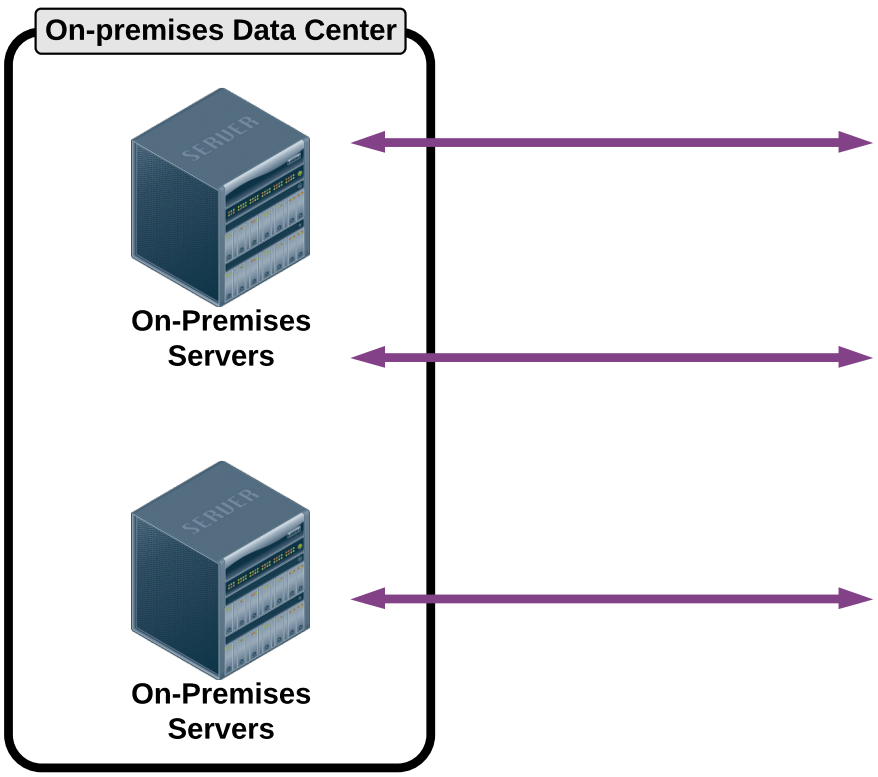
Virtual Private Network (VPN):

X

VPN Connection:

- The VPN connection is the actual link between the virtual private gateway and the cusotmer gateway.
- This connection is setup and managed in AWS.
- Each connection uses two IPsec tunnels for redundancy.

Appendix



Virtual Private Network (VPN):

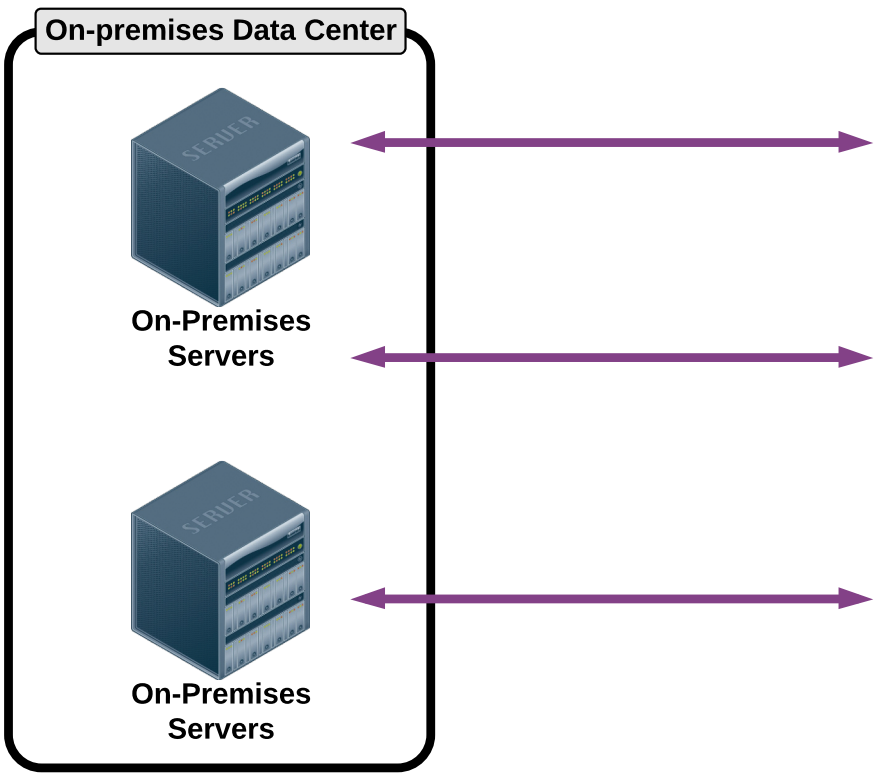
X

Virtual Private Gateway (VPG):

- A ***virtual private gateway*** acts as the “connector” on the VPC (AWS) side of the VPN connection.
- The VPG is connected to the VPC.

Note: Both a VPG and a Customer Gateway are required to establish a VPN connection.

Appendix



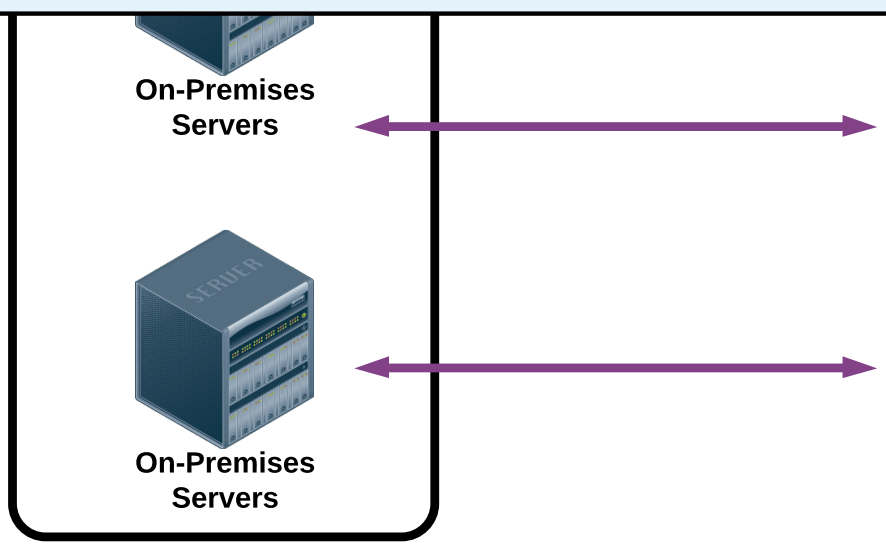
Virtual Private Network (VPN):

VPN Essentials:

- A virtual private network enables the ability to extend a subnet from one geographic location to another geographic location on two separate networks.
- Extending the subnets allows the network at location "A" to communicate internally with all resources at location "B".
- This is essentially “extending” the on-premise network to the cloud, or the cloud to the on-premise network.

- ***For AWS***, this allows us to communicate with all resources (like an EC2 instance) internally without the need for public IP addresses and an internet gateway.
- It also provides an additional level of security by ensuring that traffic sent using the VPN is encrypted.

- The VPN connection has two parallel routes (IPsec tunnels), which is for redundancy.
- Only one Virtual Private Gateway can be attached to a VPC (just like only one IGW can be attached to a VPC).
- A VPC can have both a VPG and an IGW attached at the same time.



Virtual Private Network (VPN):

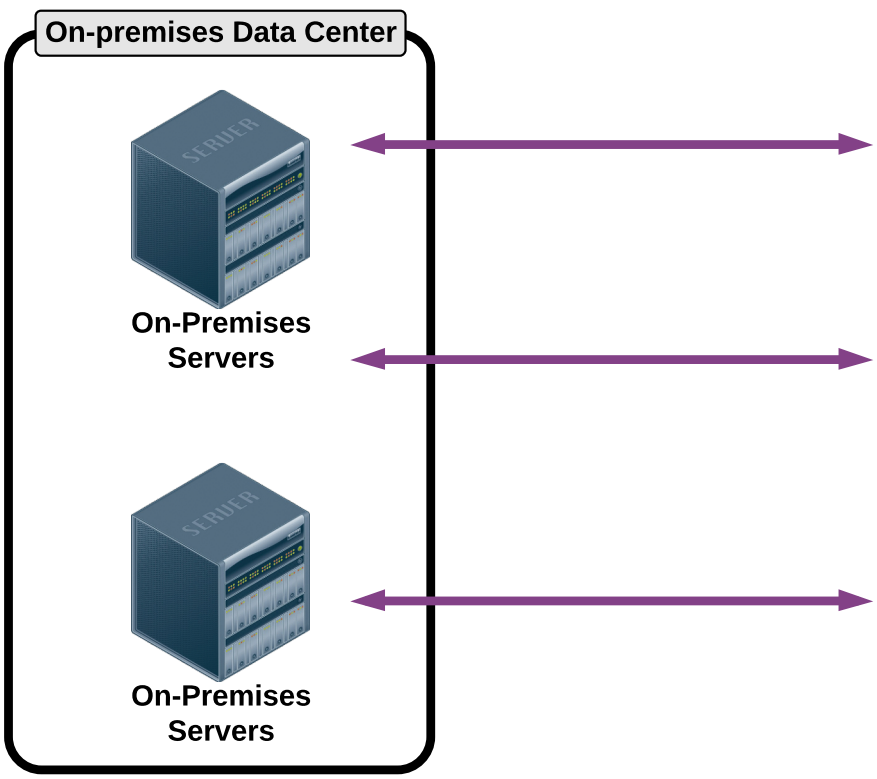
X

Router:

- AWS has dispensed with the concept of having users physically setup and manage a "router".
- However, it is important understand that route tables are actually part of a "router" assigned to your VPC.

- When setting up a VPN, the ***route table*** (for the subnet you wish to extend) must include routes for the on-premise network that are used by the VPN, and point them to the Virtual Private Gateway.

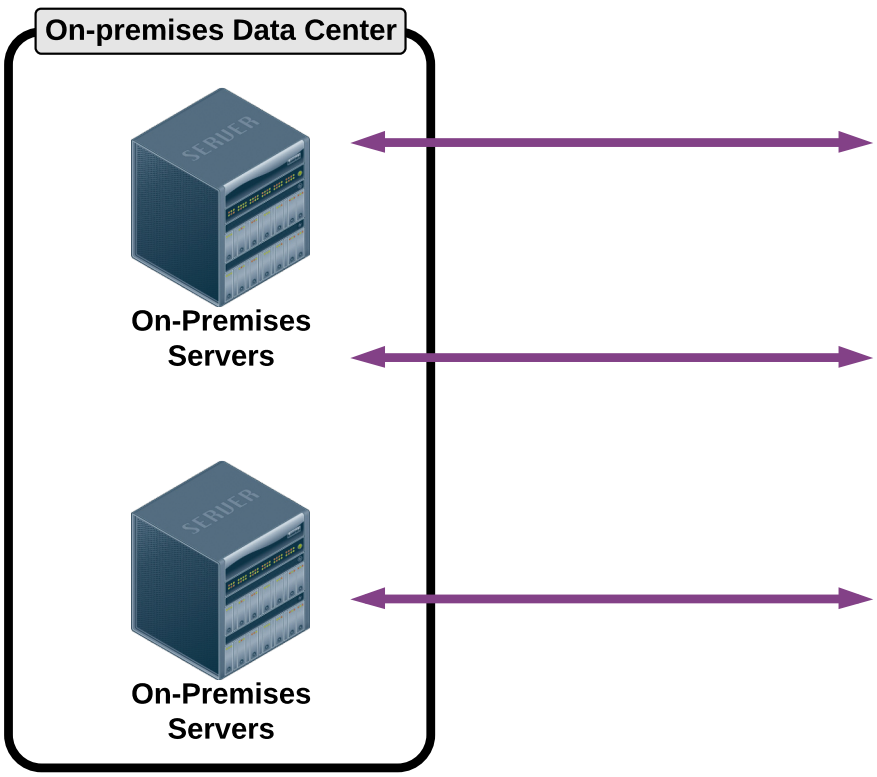
Appendix



AWS Direct Connect:

Private Virtual Interface:

- A Private Virtual Interfaces allows you to interface with an AWS (VPC).
  - With automatic route discovery using BGP
  - Requires a public or private ASN number
- Can only communicate with internal IP addresses inside of EC2.
- Cannot access public IP addresses, as Direct Connect is NOT an internet provider.
- This is a dedicated private connection which works like a VPN.
- For best practice, use two Direct Connect connections for active-active or active-failover availability.
- You can also use VPN as a backup to direct connect connections.
- You can create multiple private virtual interfaces to multiple VPC's at the same time.



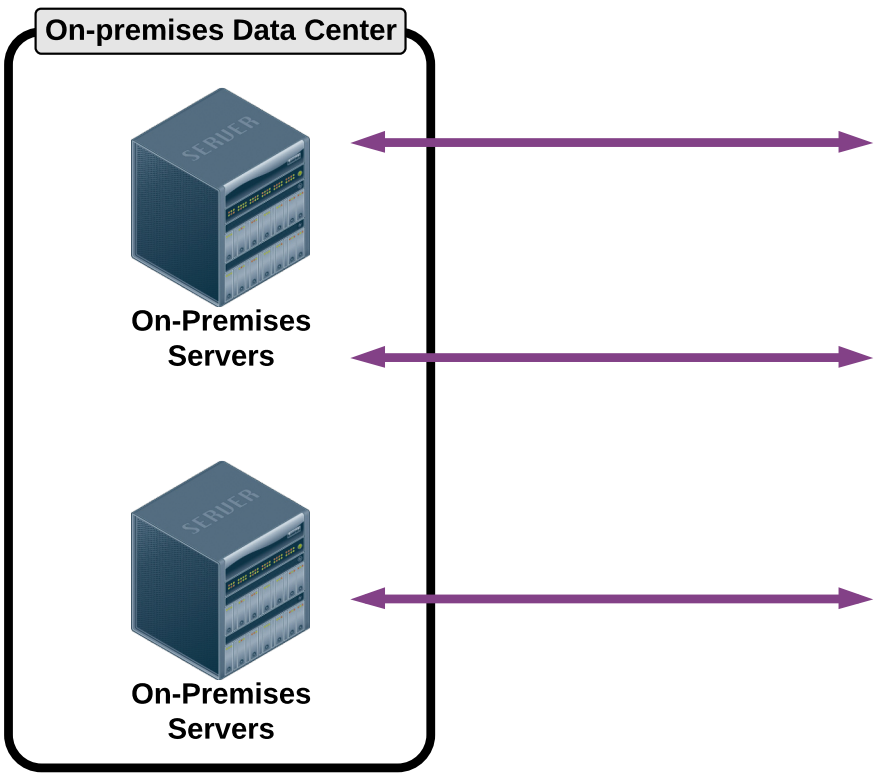
AWS Direct Connect:

X

Cross-network Connection (Cross Connect):

The ***physical connection between your network and the Direct Connect authorized partner***, which then handles the routes and connections to AWS networks.

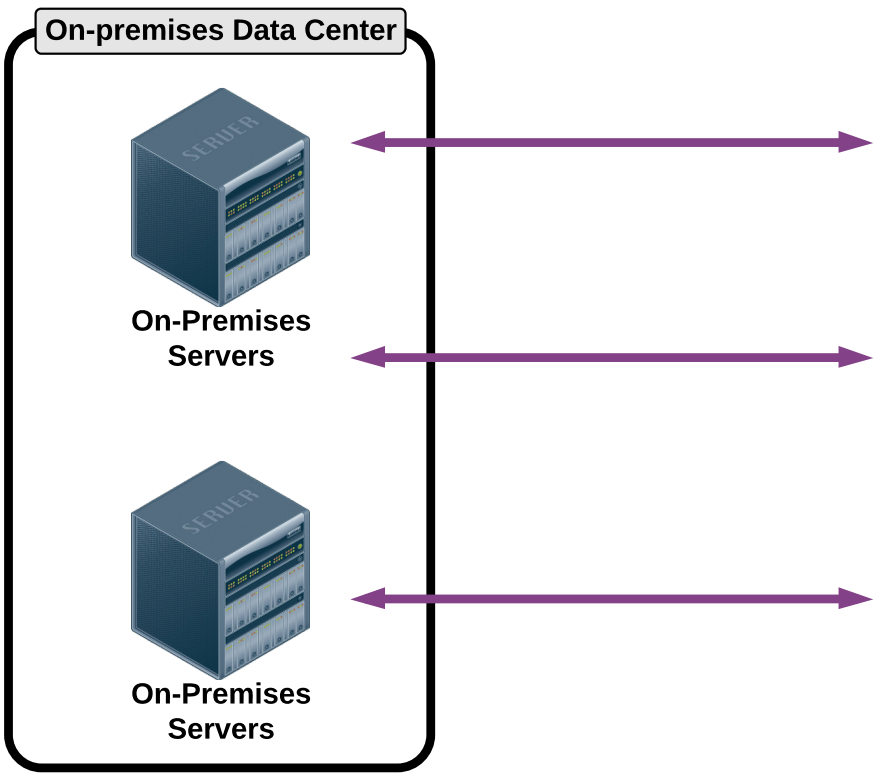
Appendix



AWS Direct Connect:

Public Virtual Interface:

- A Public Virtual Interface allows you use a Direct Connect conneciton to connect to ***public*** AWS endpoints:
  - Any AWS service (for exmample, DynamoDB and Amazon S3).
- Requires public CIDR block range.
- And even though we are accessing public endpoints, the connection maintains consistent traffic consistency as it is sent over your dedicated network.



AWS Direct Connect:

X

Direct Connect Essentials:

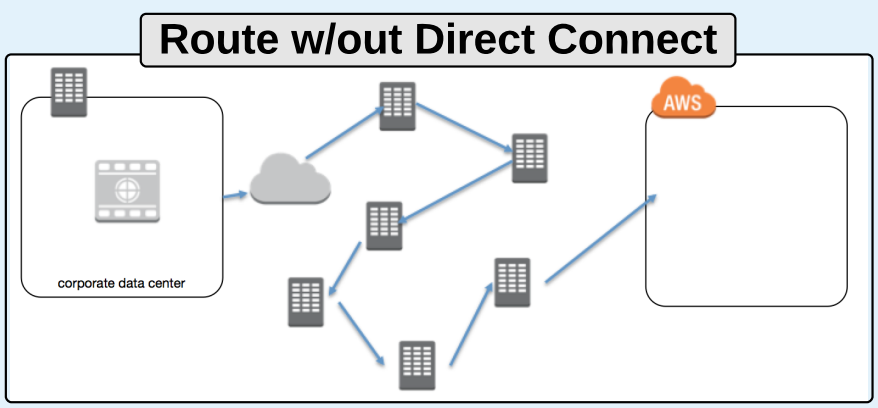
- AWS Direct Connect is a service that provides a dedicated network connection between your network and one of the AWS Direct Connect locations.
- This is done through an authorized Direct Connect Provider (i.e. Verizon or other ISPs)
- Does not require hosting any router/hardware at the Direct Connect Partner location, only requires a Direct Connect location and a participating backbone provider.
- An AWS Direct Connect location provides access to the AWS region it is associated with.
- It does not provide access to other AWS regions.

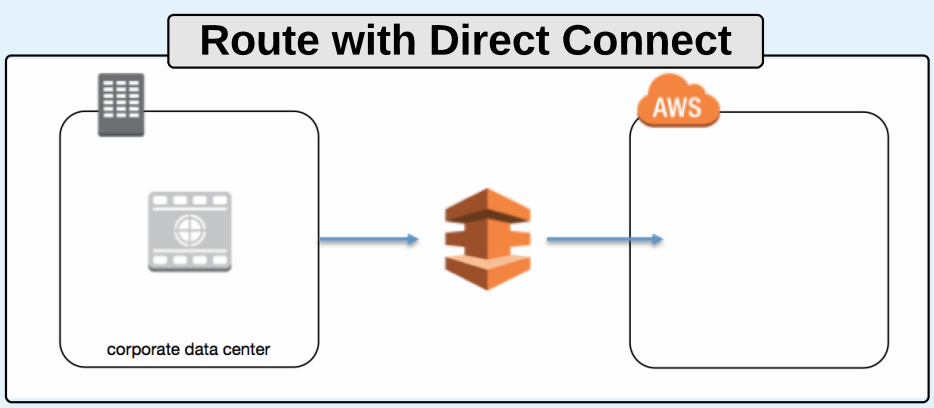
Direct Connect Benefits:

- Reduce network costs:
  - Reduce bandwidth commitment to corporate ISP over public internet.
  - Data transferred over direct connect is billed at a lower rate by Amazon (data in/out).

- Increase network consistency:
  - Dedicated private connections reduce latency (over sending the traffic via public routing).

- Dedicated private network connection to on-premise:
  - Connect the direct connect connection to a VGW in your VPC for a dedicated private connection from on-premise to VPC.
  - Use Multiple VIF (Virtual Interfaces) to connect to multiple VPCs.





AWS Storage Gateway:

Storage Gateway Essentials:

- Storage Gateway connects local data center software appliances to cloud based storage such as Amazon S3.
- It does this though the Storage Gateway virtual appliance, which connects directly to your local infrastructure as a file server, a local disk volume, or as a virtual tape library (VTL).
- It can maintain frequently accessed data on-premises (providing low-latency performance) while storing all other data in:
  - S3
  - EBS
  - Glacier
- Storage Gateway also integrates your data with:
  - AWS encryption
  - Identity management
  - Monitoring

Gateway-Cached Volumes

- Create storage volumes and mount them as iSCSI devices on the on-premise servers.
- The gateway will store the data written to this volume in Amazon S3 and will cache frequently access data on-premise in the storage device.

Gateway-Stored Volumes

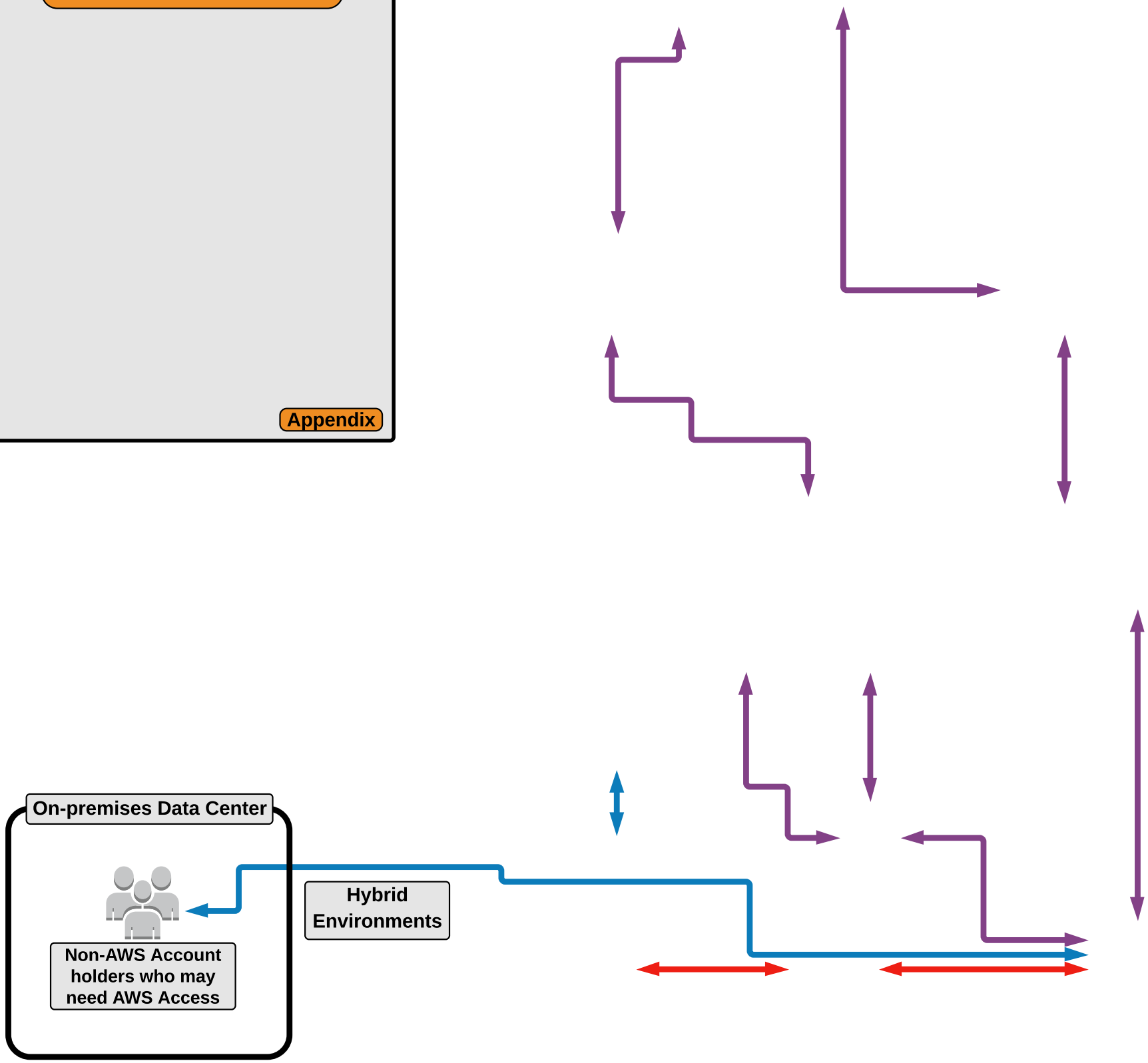
- Store all the data locally (on-premise) in storage volumes.
- Gateway will periodically take snapshots of the data as incremental backups and stores them on Amazon S3.

On-Premises Servers

Account & Services Layer

Physical & Networking Layer

Appendix



Account & Services Layer

Physical & Networking Layer

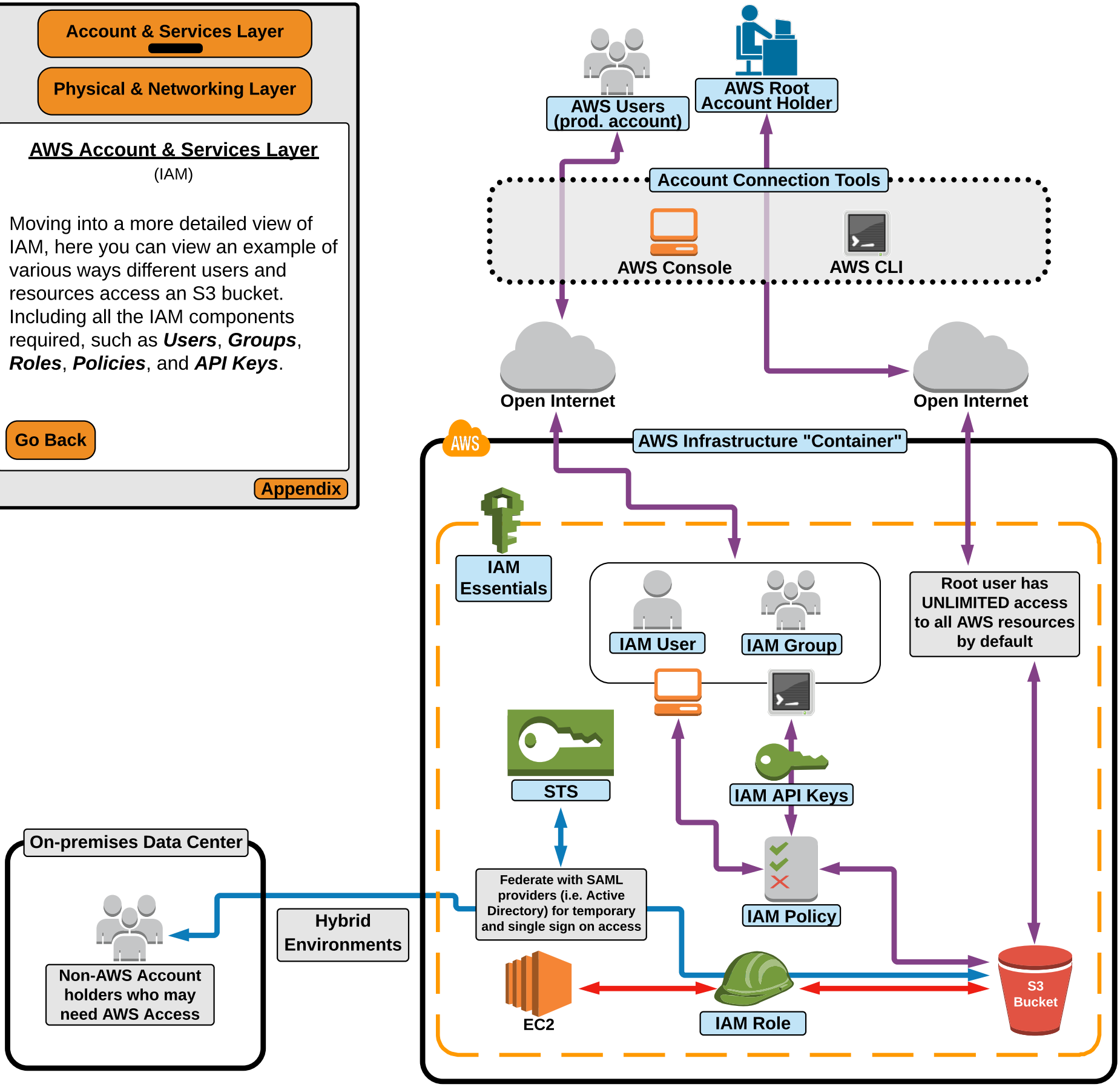
AWS Account & Services Layer

(IAM)

Moving into a more detailed view of IAM, here you can view an example of various ways different users and resources access an S3 bucket. Including all the IAM components required, such as ***Users***, ***Groups***, ***Roles***, ***Policies***, and ***API Keys***.

Go Back

Appendix



Root User:

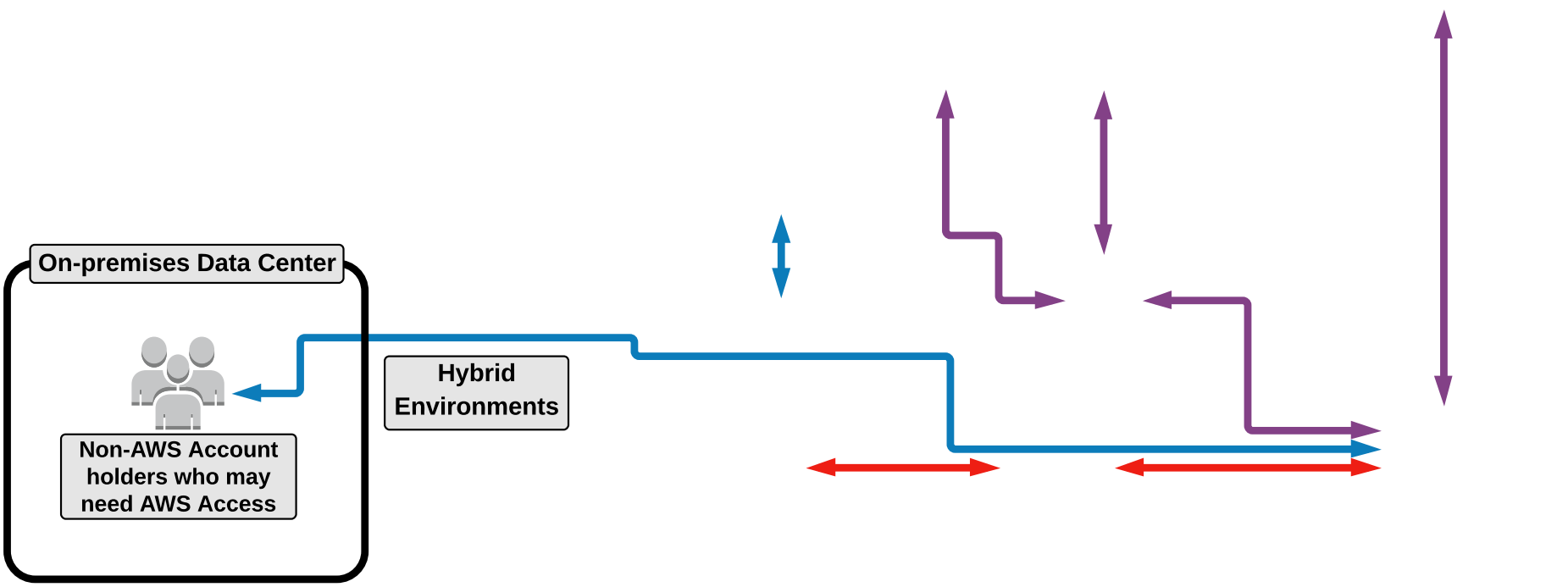
- The user created when you first create your AWS account is called the ***"root"*** user.
- It's credentials are the email address and password used when signing up for your AWS account.
- By default, the root user has ***FULL*** administrative rights and access to every part of the account.

Best Practices for Root User:

- You should not use the root user for daily work and AWS administration. You should create a second user that has admin rights and sign in as that user for daily work.

- You should always protect your root account with MFA.

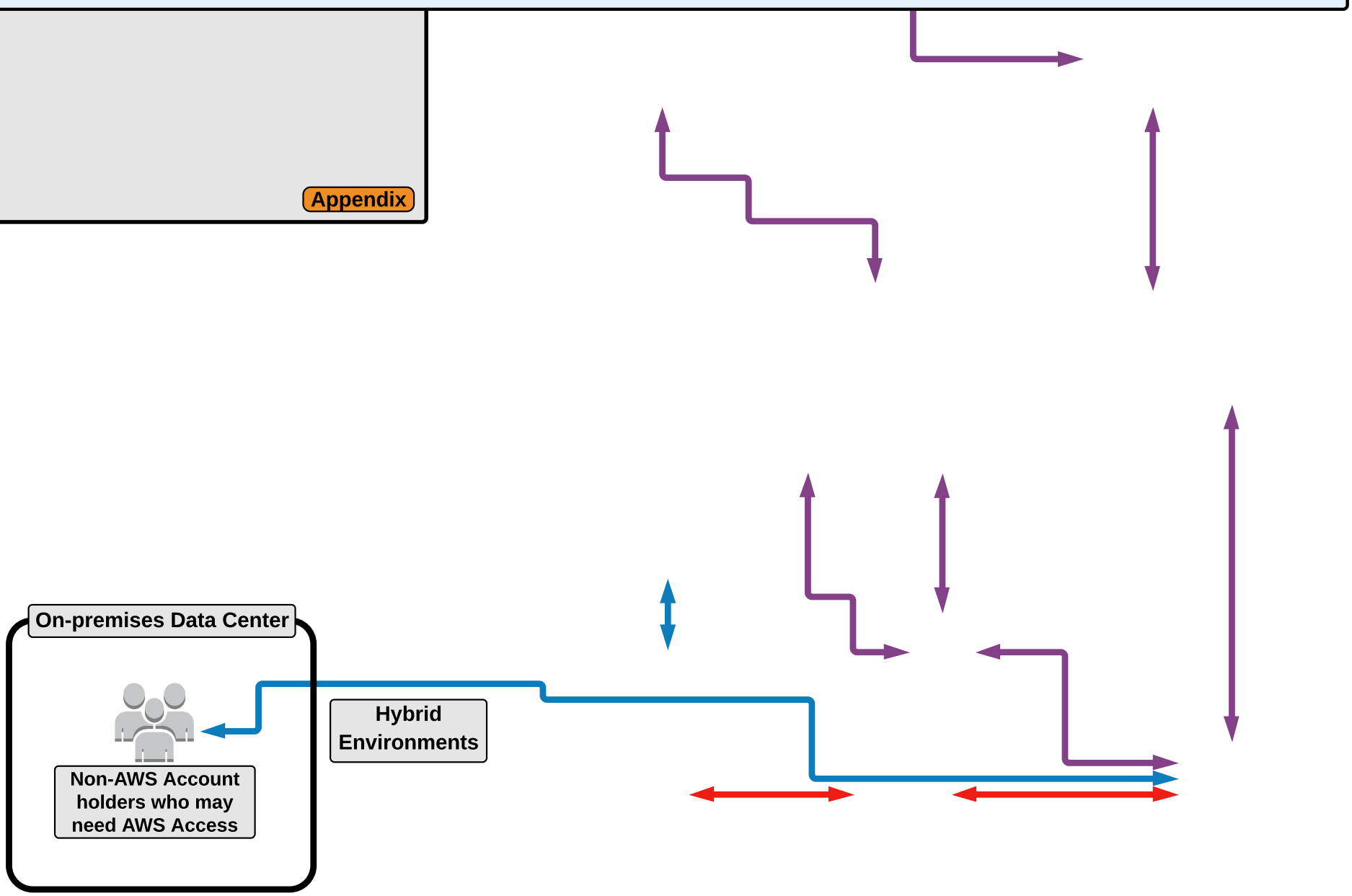
Appendix



AWS Users:

- This represents an AWS users that you may create (in IAM), who will have varying degrees of access to the "production" AWS account in the example below.

Appendix



Account Connection Tools:

X

Console

CLI

Identity & Access Management (IAM):

IAM Policies:

- A ***policy*** is a document that formally states one or more permissions.
- By default, an ***explicit deny*** always overrides and an ***explicit allow***.
  - This allows for the use of a "deny all" policy to quickly restrict ALL access that a user may have through multiple attached policies.

- 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 (i.e. user can only view what is in an S3 bucket).

- You can also create custom IAM permission policies using the ***policy generator*** or written 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).

IAM Policy Example (admin access)

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "*",
      "Resource": "*"
    }
  ]
}
```

Custom "deny all" Policy

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "*",
            "Resource": "*"
        }
    ]
}
```

Identity & Access Management (IAM):

X

IAM Essentials:

- **IAM** (**I**dentity & **A**ccess **M**anagement) is where you manage your AWS users, groups, and roles - and their access to AWS accounts and services:
  - IAM provides access and access permissions to AWS resources (such as EC2, S3, & DynamoDB)
  - IAM is global to all AWS regions, creating a user account will apply to all the regions

- The common use of **IAM** is to manage:
  - ***Users***
  - ***Groups***
  - ***Roles***
  - ***IAM Access Policies***
  - ***Api Keys***
  - ***Specify a password policy as well as manage MFA requirements on a per user basis***

- By default, any new IAM user you create in an AWS account is created with ***NO*** access to any AWS services. This is a ***non-explicit deny*** rule set on all new IAM users.

- For all users (besides the root user), permissions must be given that grant access to AWS services (this is done through IAM Policies).

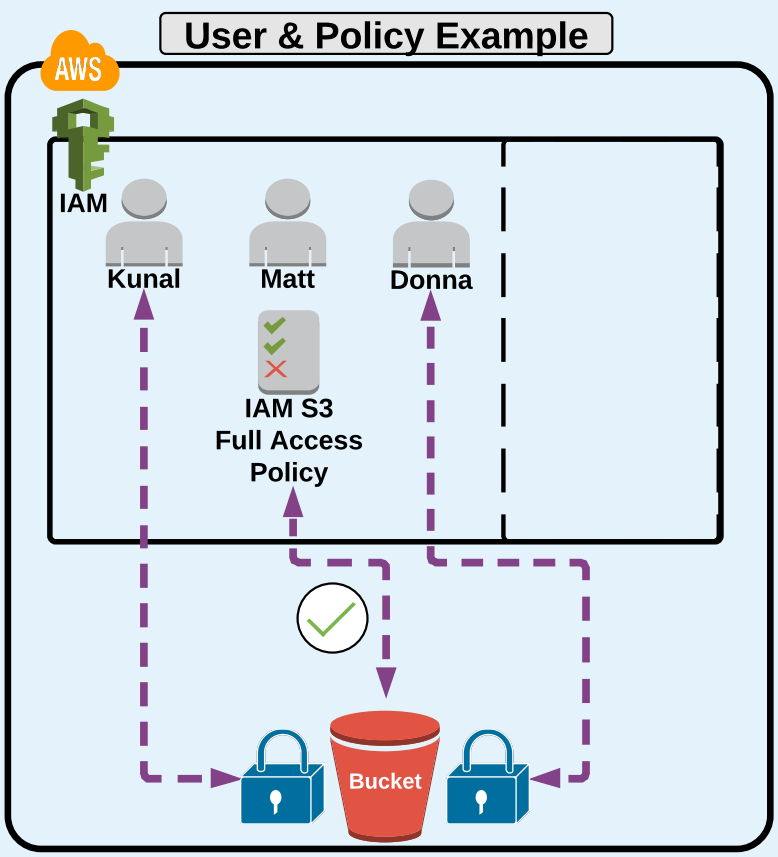
- When a new AWS root account is created, it is "***best practice***" to complete the tasks listed in IAM under "***Security Status***" - which include:
  - **Delete your root access keys**
  - **Activate MFA on your root account**
  - **Create individual IAM users**
  - **User groups to assign permissions**
  - **Apply an IAM password policy**

- ***Best practice*** is to login and do daily work as an IAM user - NOT as the root user.
- It is also best to always best to practice the "***Principal of Least Privilege***" when administering AWS accounts, users, groups, and roles.

Identity & Access Management (IAM):

IAM Users:

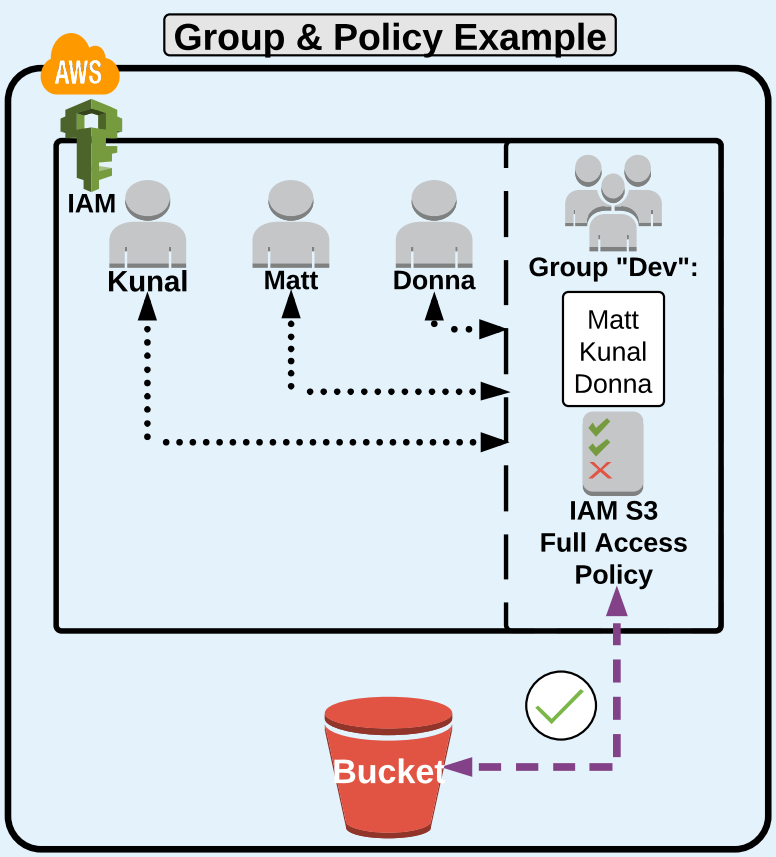
- When first created, by default an ***IAM User*** has a ***non-explicit "deny"*** for all AWS services - and does NOT have access to use them until a policy granting allow access has been applied to the user or to the group the user belongs to.
- IAM Users receive unique access credentials so you do not (and should not) share with others.
- User credentials should NEVER be stored or "passed" to an EC2 instance.
- Users can have group and regular user policies apply to them - meaning a user can have multiple IAM policies applied to them at any given time.
- By default, an ***explicit deny*** always overrides an ***explicit allow*** from attached IAM policies.
- MFA can be configured on a per user basis for login and resource access/actions.



Identity & Access Management (IAM):

IAM Groups:

- Allow you to assign IAM permission policies to more than one user at a time.
- This ability allows for easier access management to AWS resources.



Identity & Access Management (IAM):

IAM Roles:

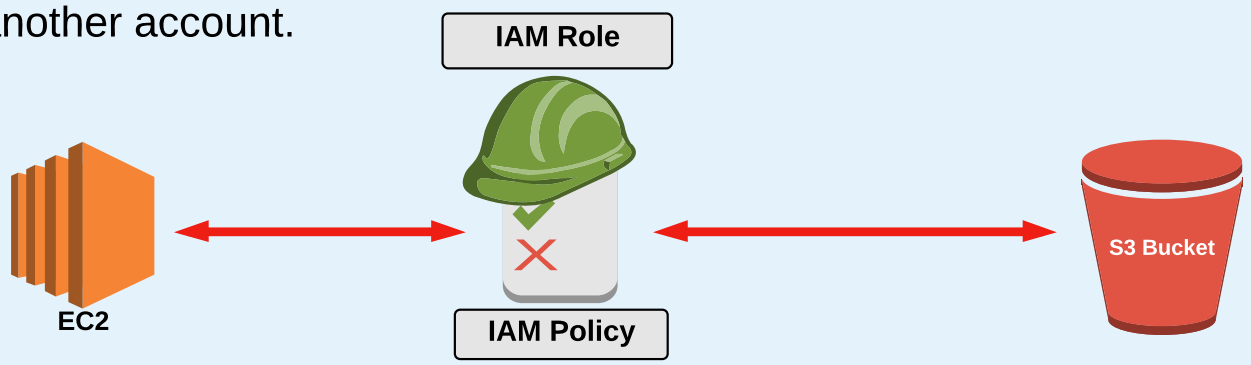
- A ***role*** is something that another entity can "assume", and in doing so acquires the specific permissions defined by the role.
- In the context of this course, "entities" that can assume a role include AWS resources (such as an EC2 instance) OR a non-AWS account holder who may need temporary access to an AWS resource (through a service like Active Directory).
- Roles must be used because policies cannot be directly attached to AWS services.

For example, if you are using an EC2 instance and it need to access an S3 bucket:

- Instance should assume a role from IAM with the proper required permissions. (S3 read-only)
- Instance can then perform actions based on the role it assumes. (read from S3)
- You “can” but should never pass or store credentials in or to an EC2 instance - so roles are used instead.
- Until recently, you could only assign a role to an EC2 instance during the EC2 instance's creation process. However, you can now assign/change a role that is assigned to an EC2 instance after the creation process via the CLI or the EC2 management console.
- An EC2 instance can only have ONE role attached at a time.

Other uses of roles:

- Other users can assume a “role” for temporary access to AWS accounts and resources through having something like Active Directory or single sign-on service (i.e Facebook, Google) assume an "Identity Provider Access" role.
- Create “cross account” access where a user from one account can assume a role with permissions in another account.



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, and can be active for a few minutes to several hours.
- Once expired, they can no longer be used to access your AWS resources.
- When requested through an STS API call, credentials are returned with three components:
  - ***Security Token***
  - ***An Access Key ID***
  - ***A Secret Access key***

STS Benefits:

- No distributing or embedding long-term AWS security credentials in an application.
- Grant access to AWS resources without having to create an IAM identity for them.
  - The basis for IAM roles and identity federation.
- Since the credentials are temporary, you don't have to rotate or revoke them.
  - You decide how long they are active for.

STS API Calls

AssumeRole: Cross-Account Delegation and Federation Through a Custom Identity Broker

AssumeRoleWithWebIdentity: Federation Through a Web-based Identity Provider

AssumeRoleWithSAML: Federation Through an Enterprise Identity Provider Compatible with SAML 2.0

GetFederationToke: Federation Through a Custom Identity Broker

GetSessionToken: Temporary Credentials for Users in Untrusted Environments

When to use STS:

- Identity Federation:
  - Enterprise identity federation (authenticate through your companies network)
    - STS Supports Security Assertion Markup Language (SAML), which allows for use of Microsoft Active Directory (of your own solutions).
  - Web identity federation (3rd party identity providers, i.e. Facebook, Google, Amazon)
- Roles for Cross-Account Access
  - Used for organizations that have more than one AWS account.
- Roles for Amazon EC2 (and other AWS services)
  - Grant access an to application running on an EC2 instance to access other AWS services without having to imbed credentials.

Identity & Access Management (IAM):

X

IAM API Keys:

- ***API Access Keys*** are required to make programmatic calls to AWS from the:
  - AWS Command Line Interface (CLI)
  - Tools for Windows PowerShell
  - AWS SDKs
  - Direct HTTP calls using the APIs for individual AWS services.

For Example:

- API credentials are user by developers, working from an on-premise network, for CLI access.

Important API Access Key Facts:

- API Keys are only available ONE time, when a new user is created OR when you reissue a new set of keys.
- AWS will NOT regenerate the same set of access keys.
- In order for API credentials to work, they must be assocaited with a USER.
- Roles does not have API credentials.
- In the AWS console you can only see the Access Key ID - never the sSecret Key ID
- If you require new API credentials - you must deactivate the current set and generate new one.
- NEVER create or store API keys on an EC2 instance.

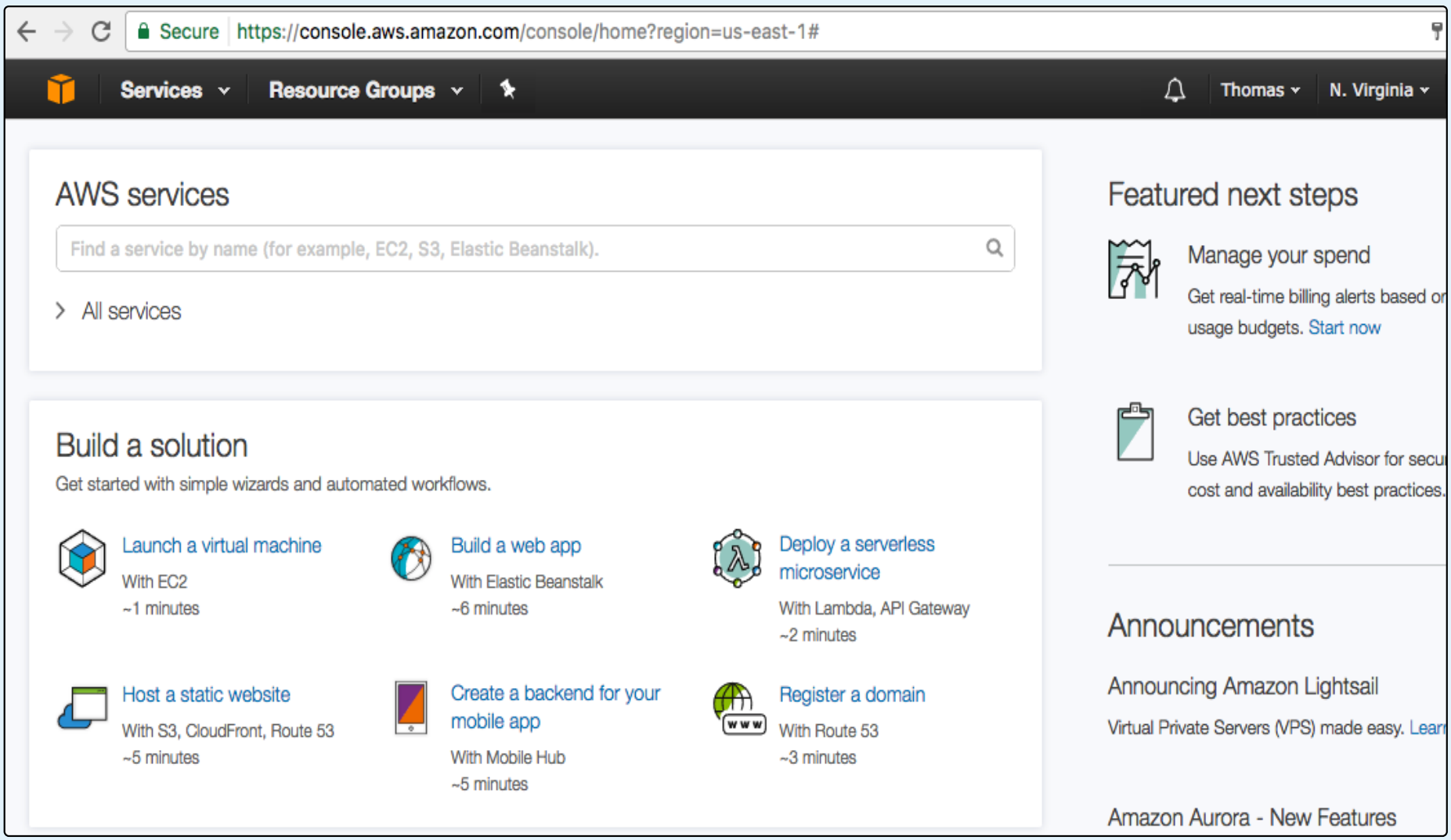
Account & Services Layer

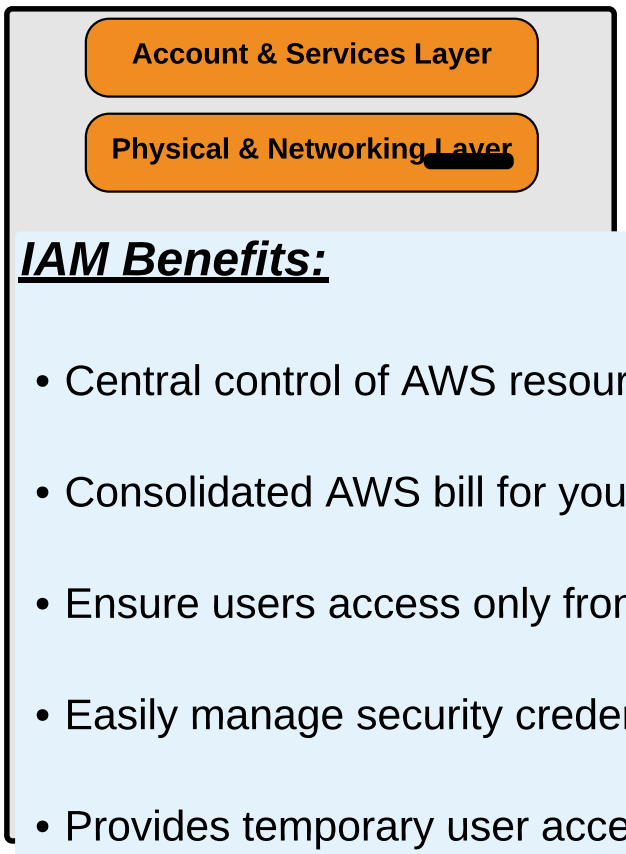
Physical & Networking Layer

AWS Management Console:

- The AWS Management Console (generally referred to as the "console") is the primary means for which we will access and interact with AWS in this course.
- All actions done in the console are API Calls

Example:





IAM Benefits:

- Central control of AWS resource access
- Consolidated AWS bill for your users
- Ensure users access only from specified networks
- Easily manage security credentials
- Provides temporary user access when needed
- Federate with SAML providers such as Active Directory for temporary and single sign on access
- Provide roles that other AWS resources can assume

Account & Services Layer

Physical & Networking Layer

AWS Command Line Interface (CLI):

- The AWS Command Line Interface (generally referred to as the "CLI") is a text based interface for accessing and administering AWS resources.
- All commands executing using the CLI are API Calls - and thus require API Key configuration.

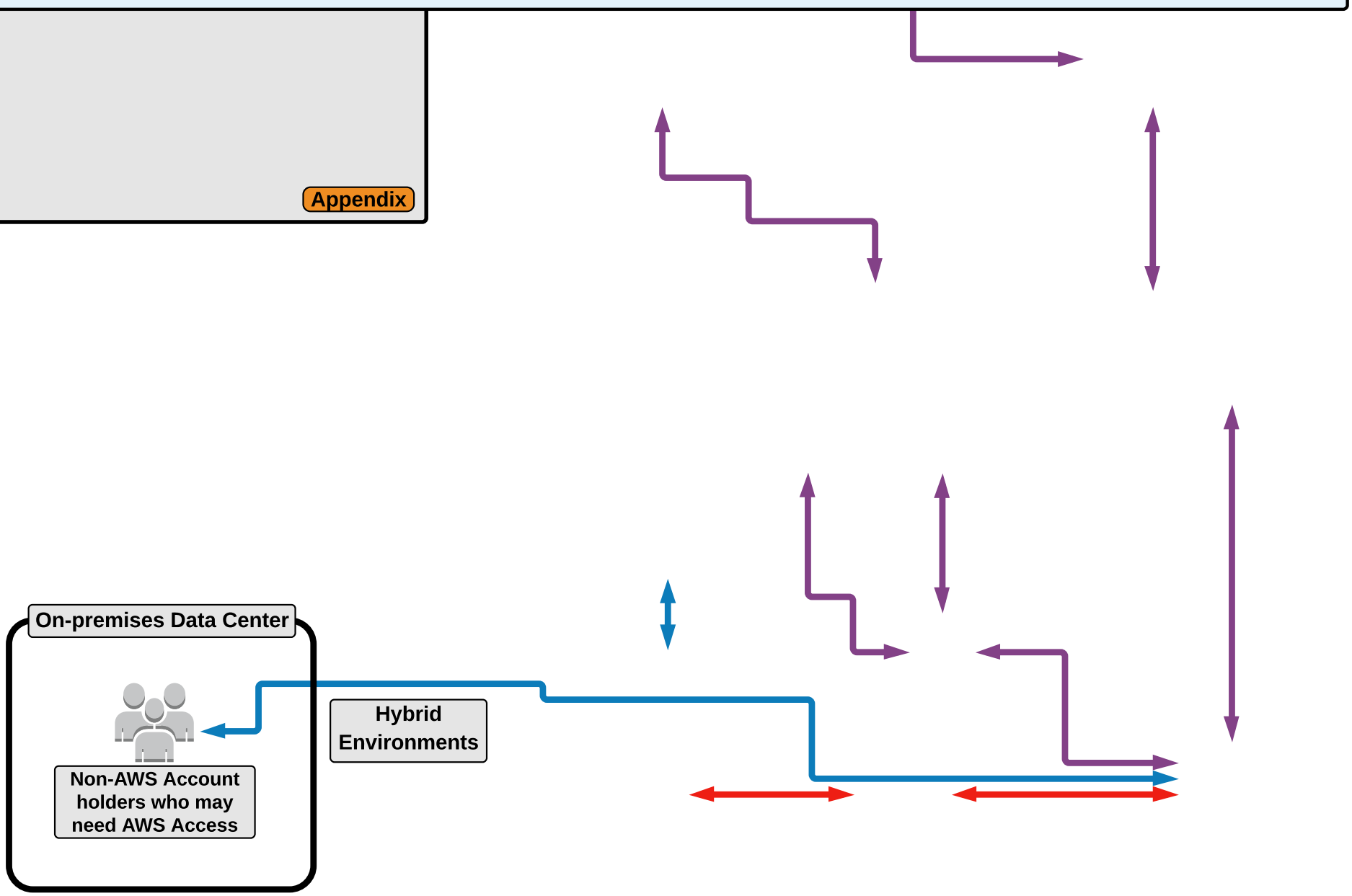
Example:

```
MacBook-Pro:downloads thaslett$ ssh -i "cli.pem" ec2-user@ec2-54-9
1-117-71.compute-1.amazonaws.com
Last login: Sat Feb  4 20:47:08 2017 from 172.58.40.128

       __|  __|_  )
       _|  (     /   Amazon Linux AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-ami/2016.09-release-notes/
[ec2-user@ip-172-31-31-150 ~]$
```

AWS Infrastructure Container:

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

Appendix



Account & Services Layer

Physical & Networking Layer

Appendix

On-premises Data Center

On-Premises Servers

Hybrid Environments

Account & Services Layer

Physical & Networking Layer

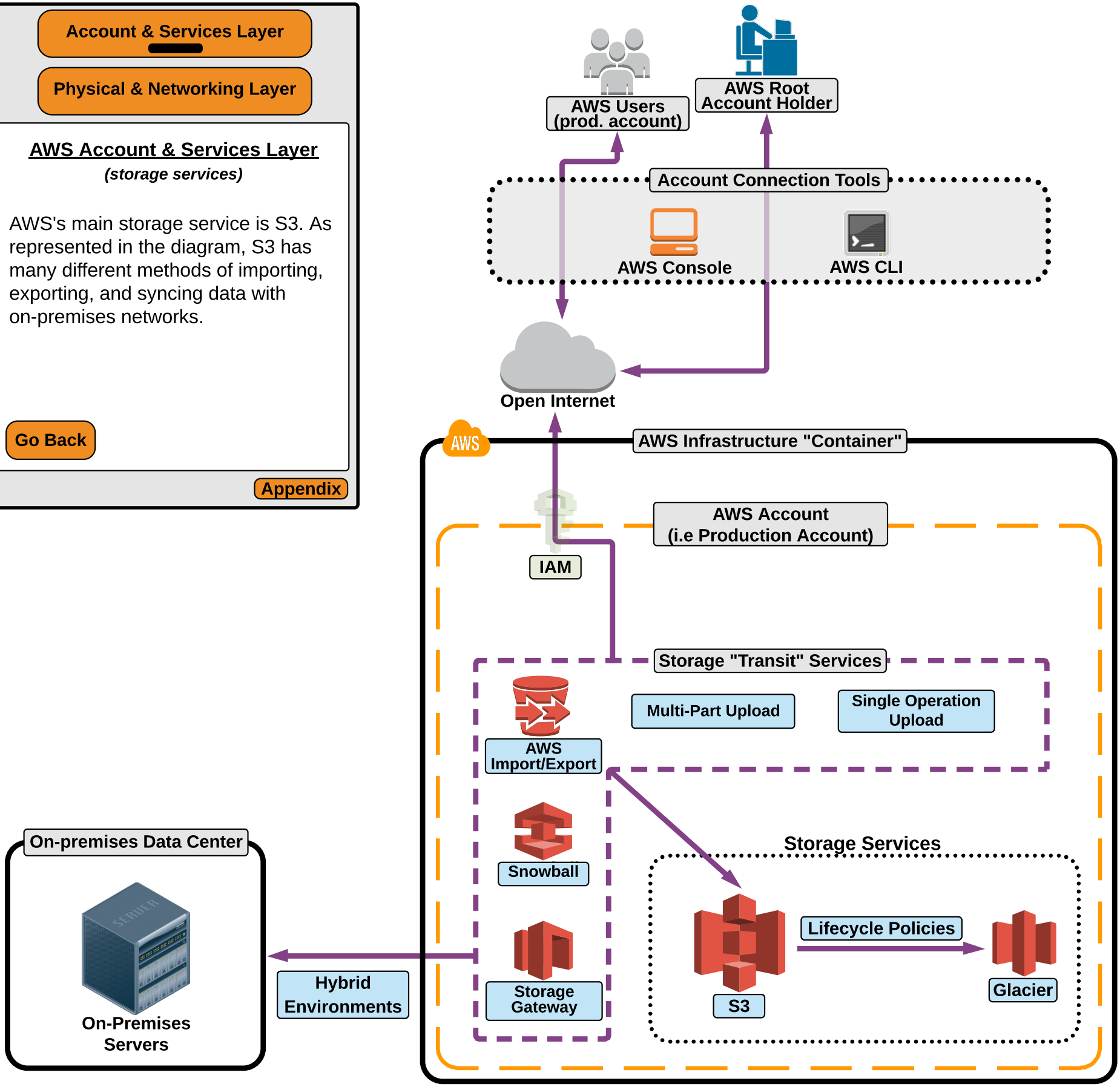
AWS Account & Services Layer

(storage services)

AWS's main storage service is S3. As represented in the diagram, S3 has many different methods of importing, exporting, and syncing data with on-premises networks.

Go Back

Appendix



Account & Services Layer

Physical & Networking Layer

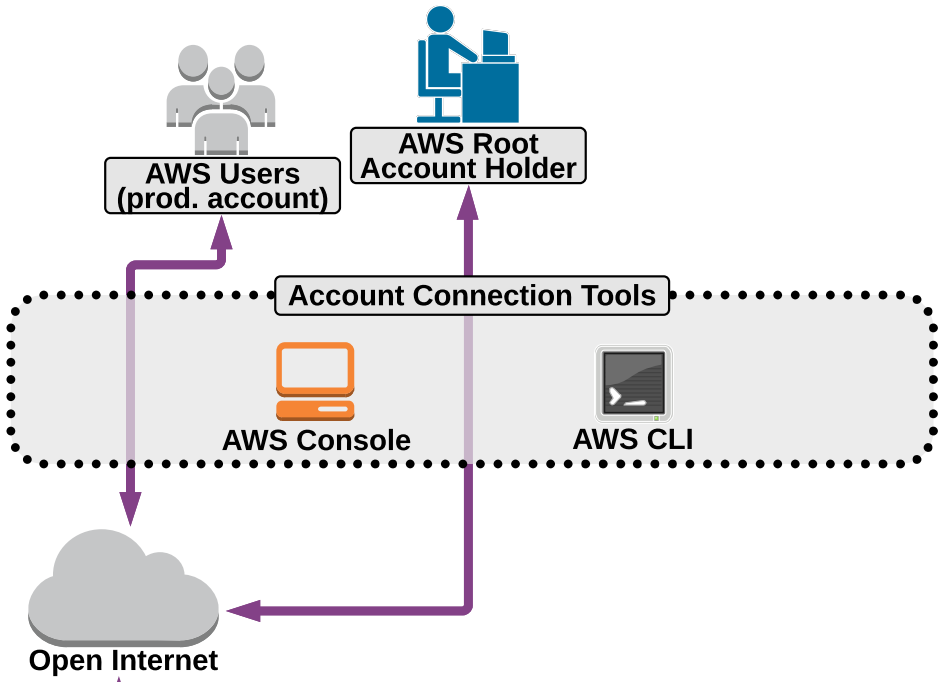
AWS Account & Services Layer

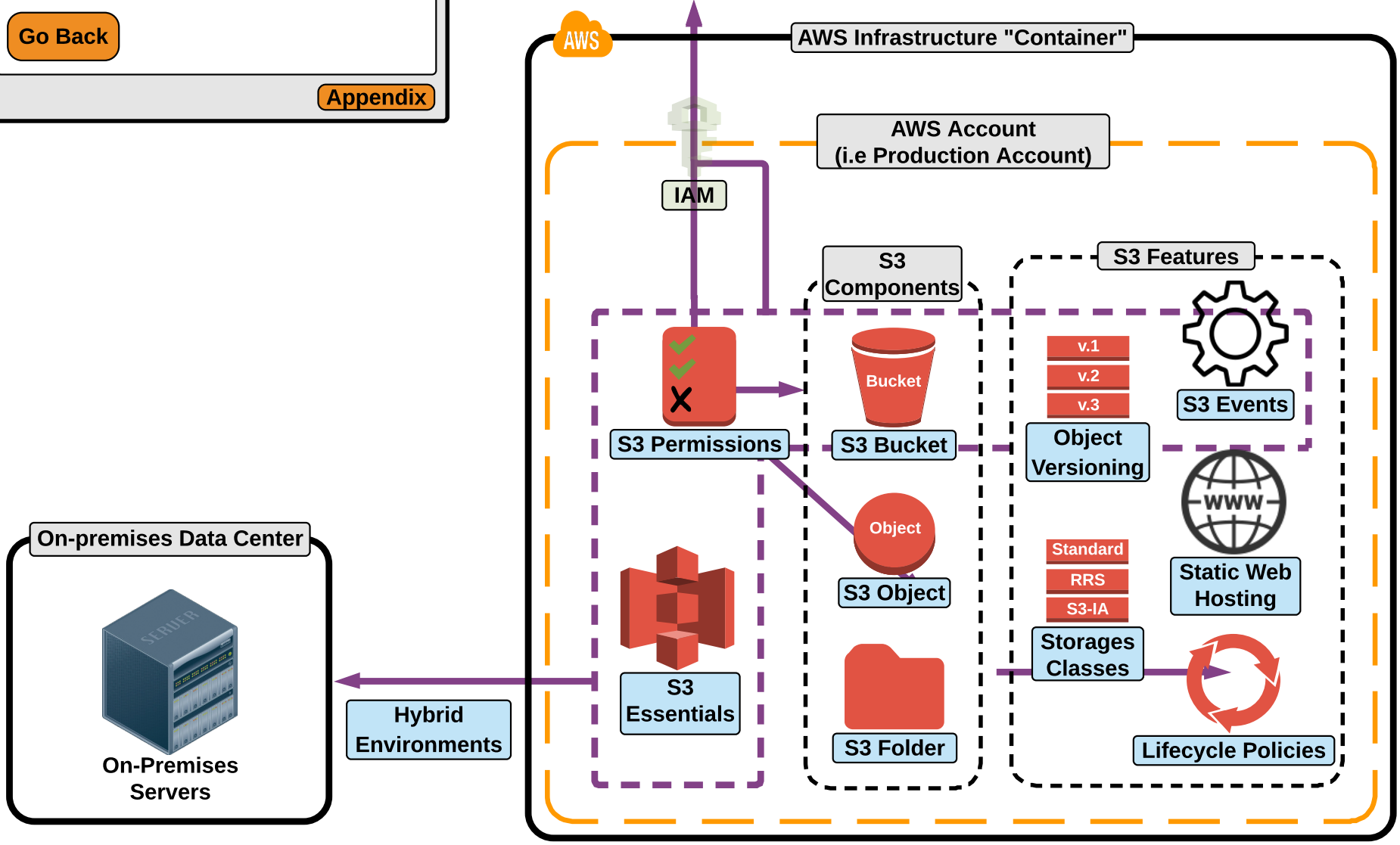
(Simple Storage Service)

Moving into S3, we can view its main components (***buckets***, ***objects***, and ***folders***) - as well as it's primary features that help us design highly available, fault tolerant, secure and const effective applications and architecture.

Go Back

Appendix





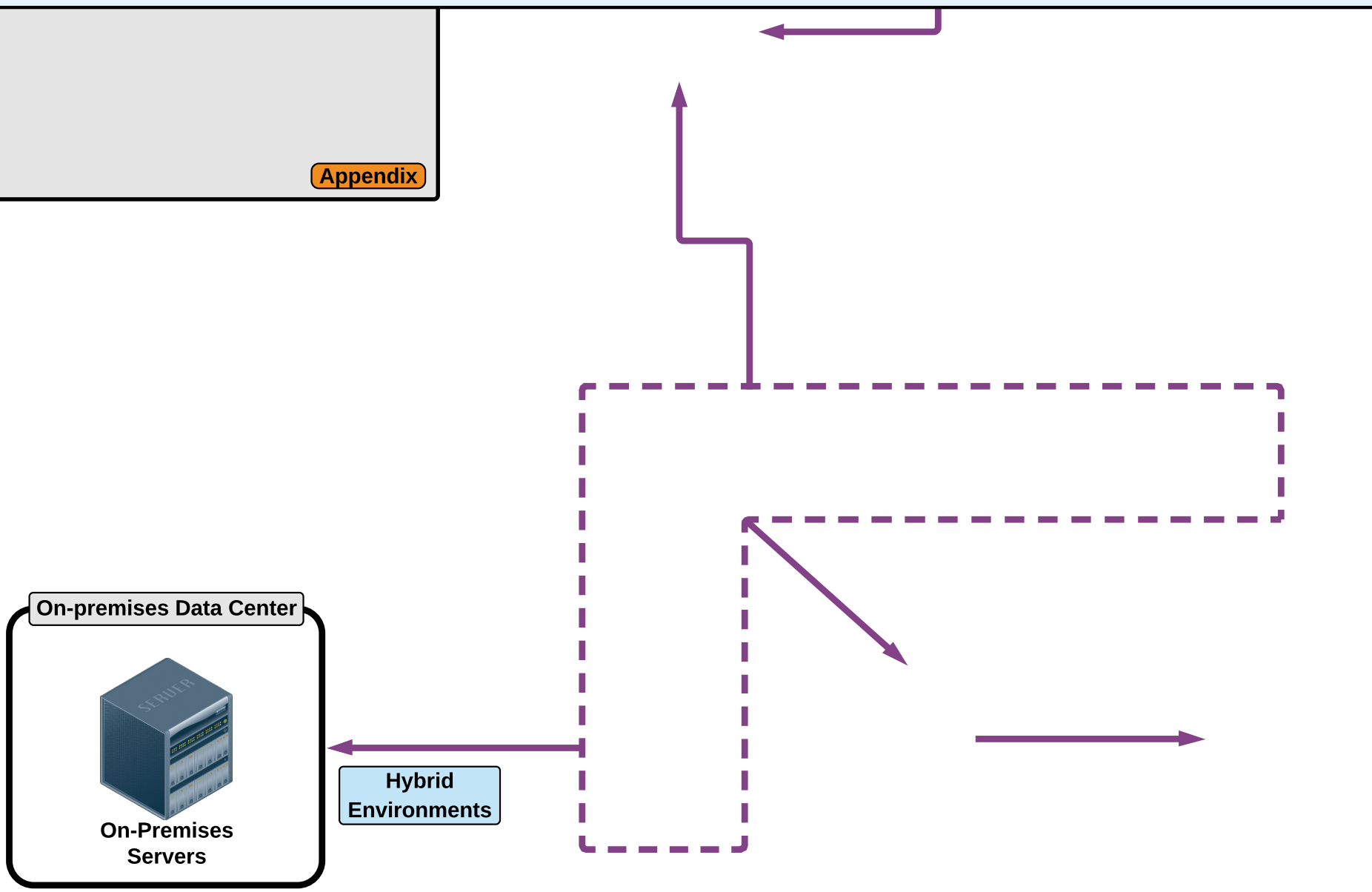
Simple Storage Service (S3):

X

Snowball:

- Snowball is a petabyte-scale data transport solution.
- Snowball uses an AWS provided secure transfer appliance.
- Quickly move large amounts of data into and out of the AWS cloud.

Appendix



Simple Storage Service (S3):

X

Storage Gateway:

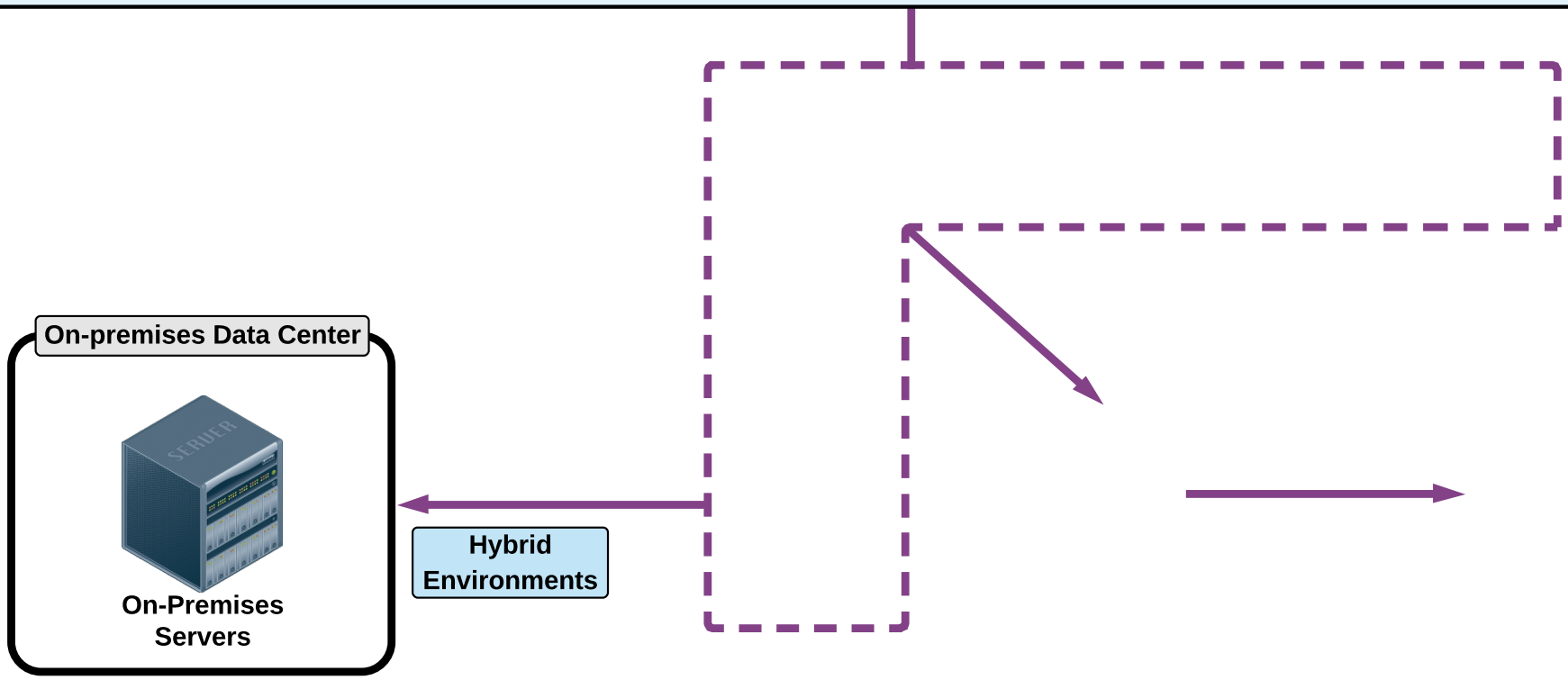
- Connects local data center software appliances to cloud based storage such as Amazon S3.

Gateway-Cached Volumes

- Create storage volumes and mount them as iSCSI devices on the on-premise servers.
- The gateway will store the data written to this volume in Amazon S3 and will cache frequently access data on-premise in the storage device.

Gateway-Stored Volumes

- Store all the data locally (on-premise) in storage volumes.
- Gateway will periodically take snapshots of the data as incremental backups and stores them on Amazon S3.



Simple Storage Service (S3):

S3 Essentials:

- As AWS' main storage service, S3 can serve many purposes when designing highly available, fault tolerant, and secure application architecture. Including:
  - Bulk (basically unlimited) static object storage
  - Various ***storage classes*** to optimize cost vs. needed object availability/durability
  - Object versioning
  - Access restrictions via S3 bucket policies/permissions
  - Object management via lifecycle policies
  - Hosting static files & websites
  - Origin for CloudFront CDN
  - File shares and backup/archiving for hybrid networks (via AWS Storage Gateway)

Important S3 Facts:

- Objects stay within an AWS region and are synced across all AZ's for extremely high availability and durability.
- You should always create an S3 bucket in a region that makes sense to it's purpose:
  - Serving content to customers
  - Sharing Data with EC2

S3 Read Consistency Rules:

- ALL regions now support read-after-write consistency for PUTS of new objects into S3.
  - Object can be immediately available after "putting" an object in S3
- All regions use eventual consistency for PUTS overwriting existing objects and DELETES of objects.

Simple Storage Service (S3):

X

S3 Objects:

- Objects are static files that contain metadata information:
  - Set of name-key pairs
  - Contain information specified by the user, and AWS information such as storage type
- Each object must be assigned a storage type, which determines the object's availability, durability, and cost.
- By default, all objects are private.
- Objects can:
  - Be as small as 0 bytes and as large as 5 TB.
  - Have multiple versions (if versioning is enabled)
  - Be made publicly available via a URL
  - Automatically switch to a different storage class or deleted (via lifecycle policies)
  - Encrypted
  - Organized into "sub-name" spaces called folders

Object Encryption:

- SSE (Server Side Encryption):
  - S3 can encrypt the object before saving it on the partitions in the data centers and decrypt it when it is downloaded
  - AES-256
- Or you can use your own encryption keys:
  - Considered client side encryption where you encrypt the data before upload
- SSL terminated endpoints for the API

Servers

Simple Storage Service (S3):

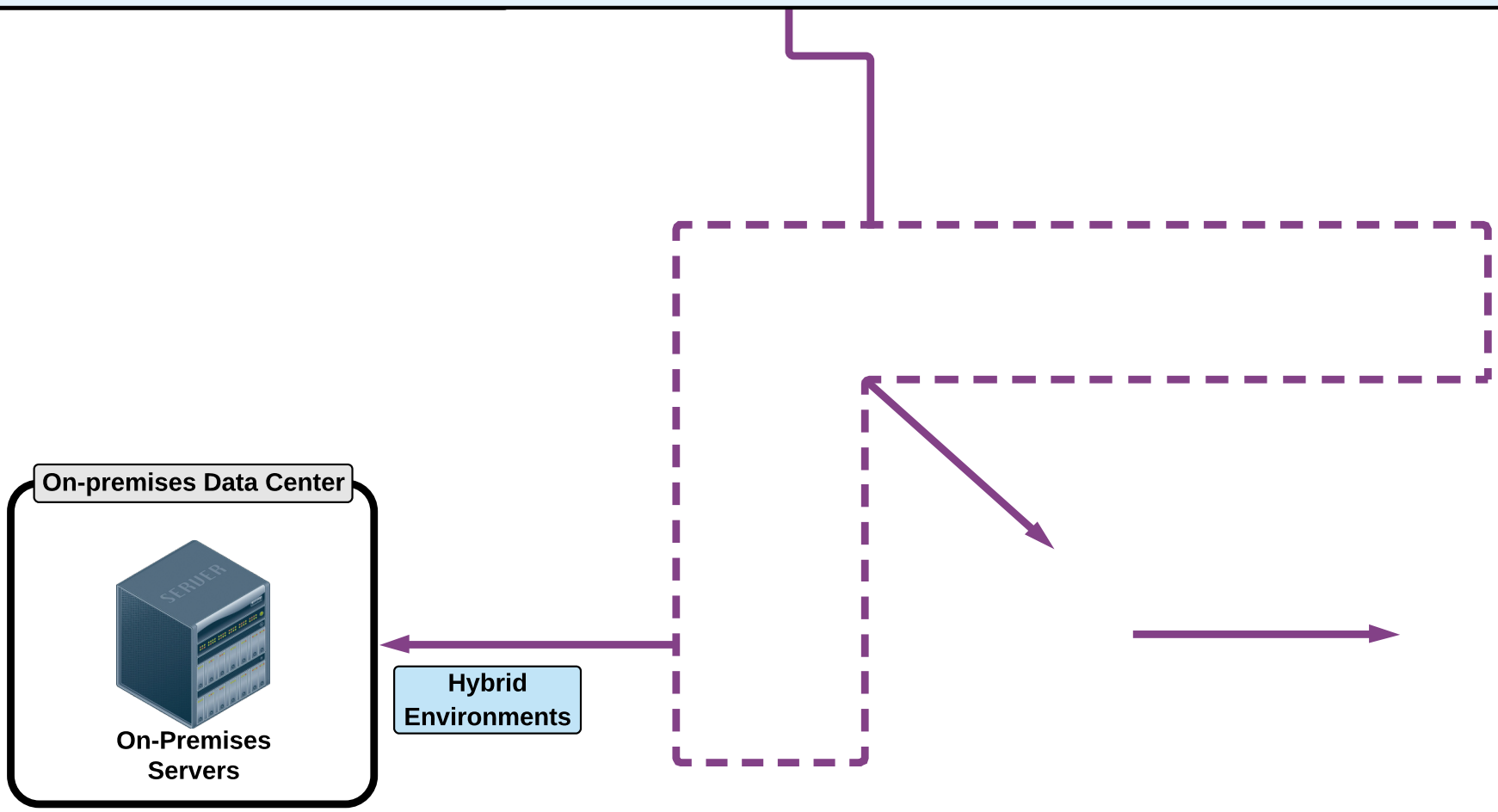
X

S3 Buckets:

- Buckets are the main storage container of S3, and contain a grouping of information and have sub name spaces that are similar to folders (but yet are called folders).
- Tags can be used to organize buckets (i.e. tag based on application the bucket belongs to).

- ***Each bucket must have a unique name across ALL of AWS.***

- ***Bucket Limitations***:
  - Only 100 buckets can be created in an AWS account at a time.
  - Bucket ownership cannot be transferred once a bucket is created.

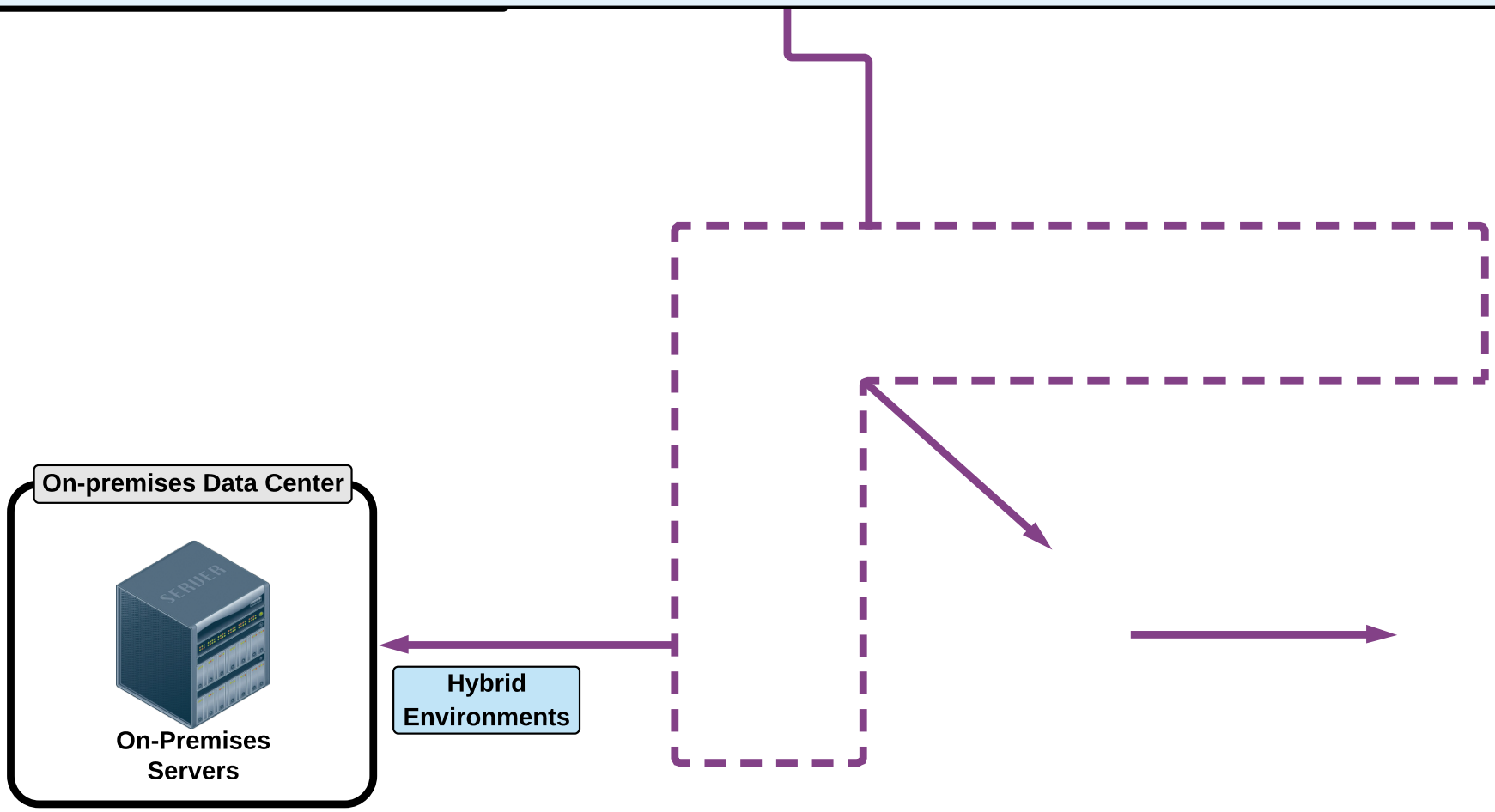


Simple Storage Service (S3):

S3 Folders:

- For simplicity, S3 supports the concept of "folders".
- This is done only as a means of grouping objects.
- Amazon S3 does this by using key-name prefixes for objects.

Amazon S3 has a flat structure, there is no hierarchy like you would see in a typical file system.



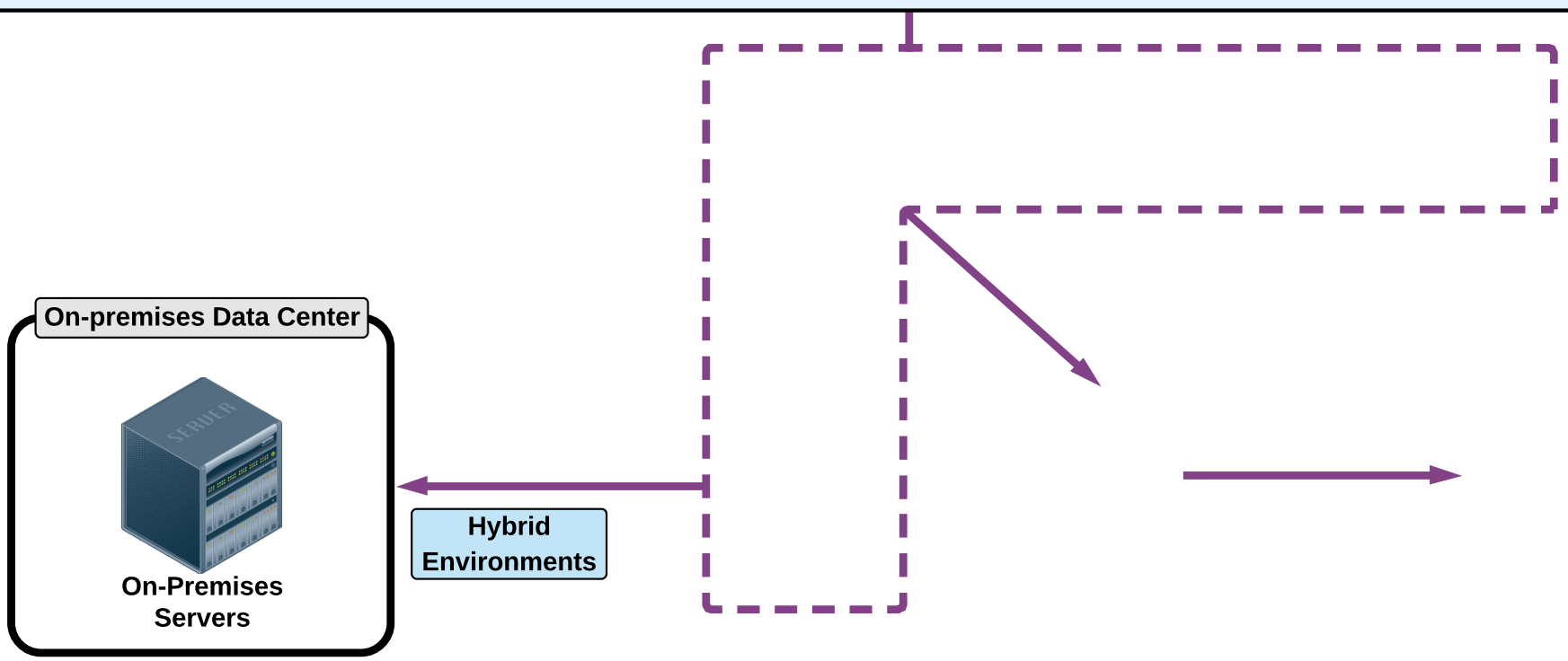
Simple Storage Service (S3):

S3 Event Notifications:

- S3 ***events notifications*** allow you to setup automated communication between S3 and other AWS services when a selected event occurs in an S3 bucket.

- Common event notification triggers include:
  - ***RRSObjectLost*** (Used for automating the recreation of lost RRS objects)
  - ***ObjectCreated*** (for all or the following specific APIs called)
    - Put
    - Post
    - Copy
    - ***CompleteMultiPartUpload***

- Events notification can be sent to the following AWS services:
  - SNS
  - Lambda
  - SQS Queue



Simple Storage Service (S3):

S3 Storage Classes:

- A ***storage class*** represents the "classification" assigned to each Object in S3. Current Storage Class types include:
  - Standard
  - Reduced Redundancy Storage (RRS)
  - Infrequent Access (S3-IA)
  - Glacier
- Each ***storage class*** has varying attributes that dictate things like:
  - Storage cost
  - Object ***availability***
  - Object ***durability***
  - Frequency of access (to the object)

Standard:

- Designed for general, all-purpose storage.
- Is the default storage option.
- ***99.999999999% object durability*** (“eleven nines”).
- ***99.99% object availability.***
- Is the most expensive storage class.

Reduced Redundancy Storage (RRS):

- Designed for non-critical, reproducible objects.
- ***99.99% object durability***.
- ***99.99% object availability***.
- Is less expensive than the standard storage class.

Infrequent Access (S3-IA):

- Designed for objects that you do not frequently access, but must be immediately available when accessed.
- ***99.999999999% object durability***.
- ***99.90% object availability.***
- Is less expensive than the standard/RRS storage classes.

Glacier:

- Designed for long-term archival storage (not to be used for backups).
- May take several hours for objects stored in Glacier to be retrieved.
- ***99.999999999% object durability***
- Is the cheapest S3 storage class (very low cost)

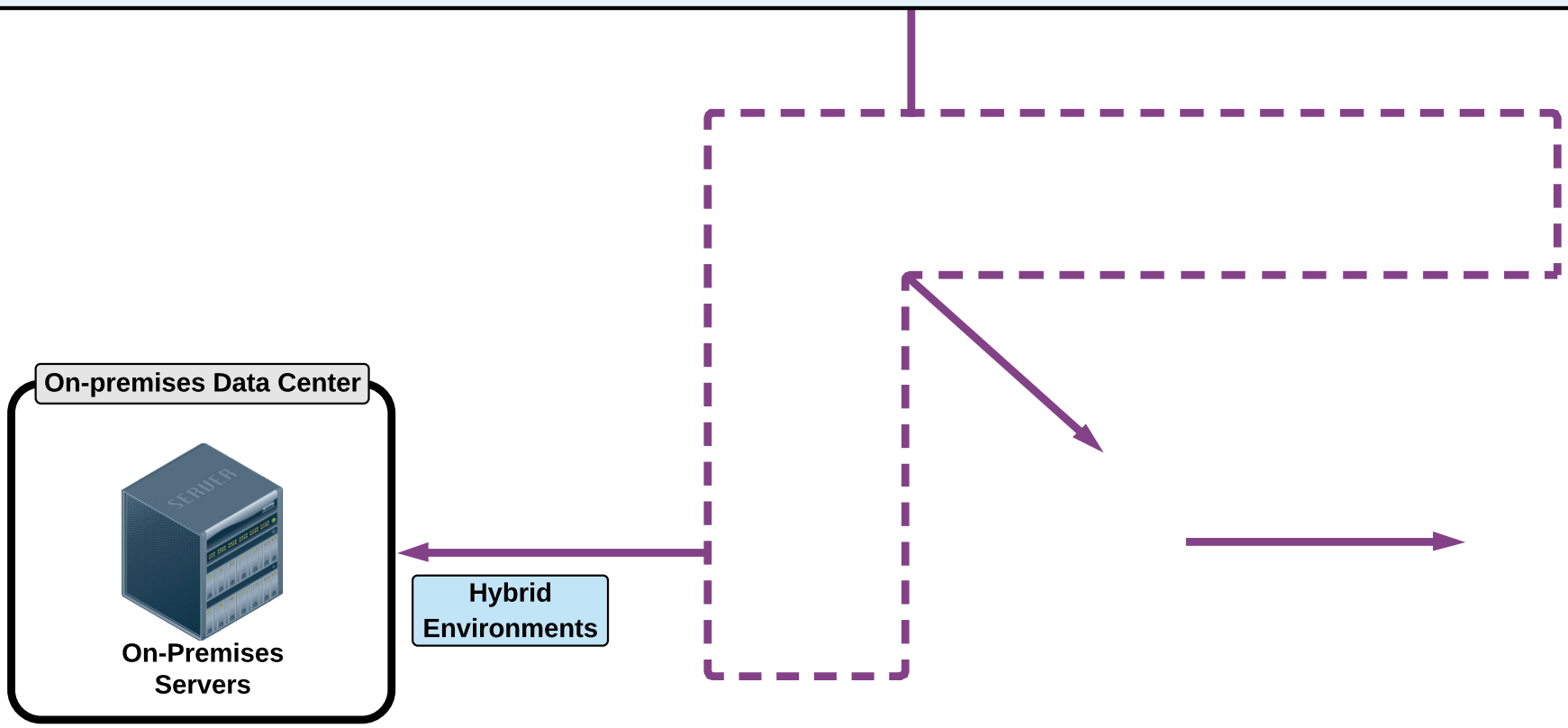
Simple Storage Service (S3):

S3 Versioning:

- S3 versioning is a feature to manage and store all old/new/deleted versions of an object.

- By default, versioning is disabled on all buckets/objects.
- Once versioning is enabled, you can only "suspend" versioning. It cannot be fully disabled.
- Suspending versioning only prevents new versions from being created. All objects with existing versions will maintain their older versions.
- Versioning can only be set on the bucket level and applies to ALL objects in the bucket.

- Lifecycle policies can be applied to specific versions of an object.
- Versioning and lifecycle policies can both be enabled on a bucket at the same time.
- Versioning can be used with lifecycle policies to create a great archiving and backup solution in S3.



Simple Storage Service (S3):

S3 Permissions:

- All buckets and objects are private by default - only the resource owner has access.
- The resource owner can grant access to the resource (bucket/objects) through S3 "resource based policies" OR access can be granted through a traditional IAM user policy.
- Resource based policies (for S3) are:

  - ***Bucket policies***
    - Are policies that are attached ***only*** to the S3 bucket (not an IAM user).
    - The permissions in the policy are applied to all objects in the bucket.
    - The policy specifies what actions are allowed or denied for a particular user of that bucket - such as:
      - Granting access to an anonymous User.
      - Who (a "principal") can execute certain actions like PUT or DELETE.
      - Restricting access based off of IP address (generally used for CDN management).

  - ***S3 access control lists***
    - Grant access to users in other AWS accounts or to the public.
    - Both buckets and objects has ACLs.
    - Object ACLs allow us to share an S3 object with the public via a URL link.

Simple Storage Service (S3):

S3 Static Web Hosting:

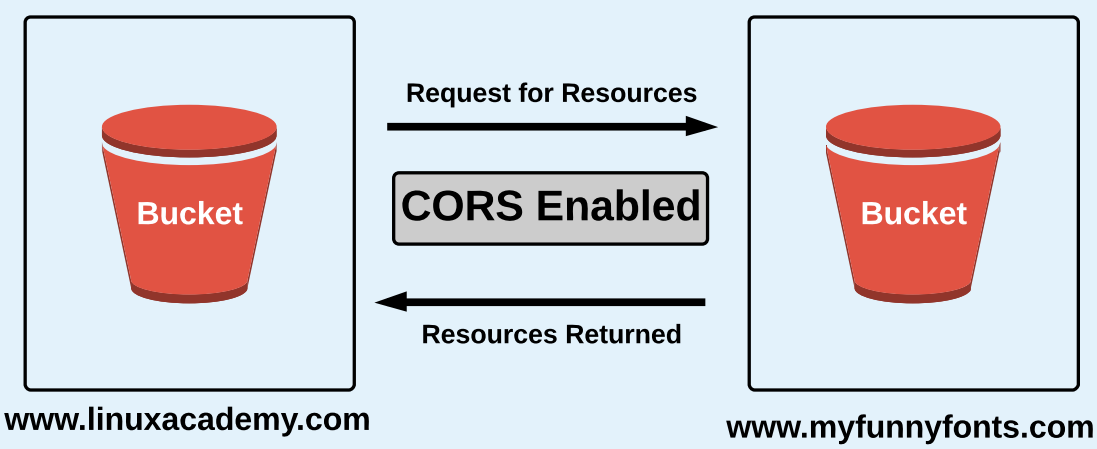
- Amazon S3 provides an option for a low-cost, highly reliable web hosting service for static websites (content that does not change frequently).

- When enabled, static web hosting will provide you with an unique endpoint (url) that you can point to any properly formatted file stored in an S3 bucket. Supported formats include:
  - HTML
  - CSS
  - JavaScript

- Amazon Route 53 can also map human-readable domain names to static web hosting buckets, which are ideal for DNS failover solutions.

Cross-Origin Resource Sharing (CORS):

- CORS is a method of allowing a web application located in one domain to access and use resources in another domain.
- This allows web applications running JavaScript or HTML5 to access resources in an S3 bucket without using a proxy server.
- For AWS, this (commonly) means that a web applications hosted in one S3 bucket can access resources in another S3 bucket.



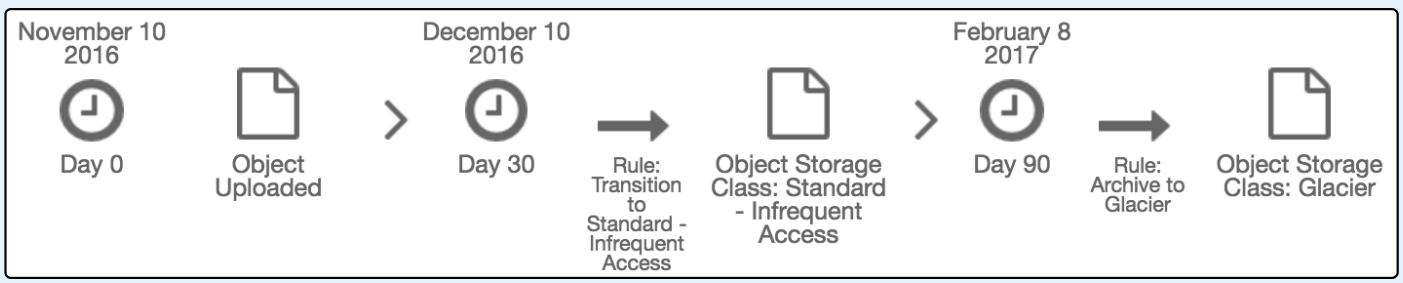
Simple Storage Service (S3):

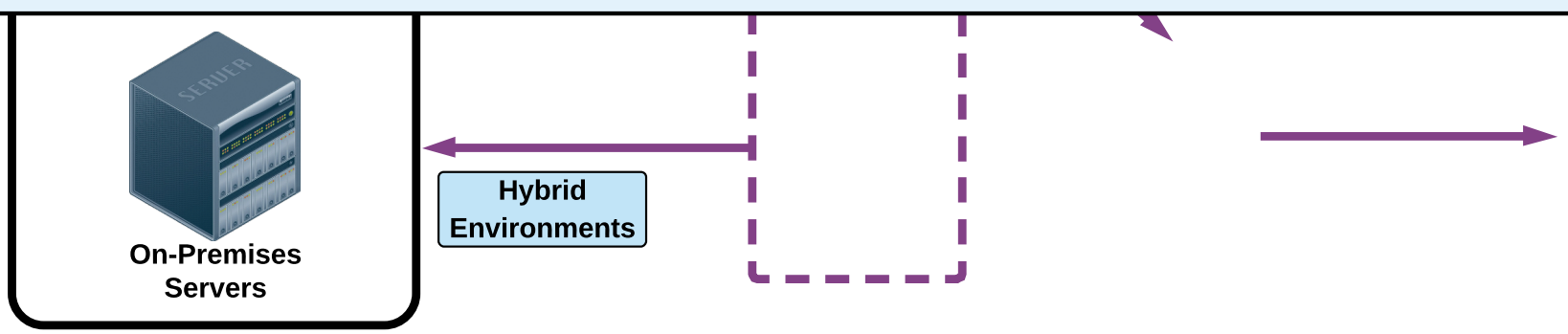
Lifecycle Policies:

An object lifecycle policy is a ***set of rules that automate*** the migration of an object's storage class to a different storage class (or deletion), based on specified time intervals.

- By default, lifecycle policies are disabled on a bucket/object.
- Are customizable to met your company's data retention policies.
- Great for automating the management of object storage and to be more cost efficient.
- Can be used with versioning to create a great archiving and backup solution in S3.

Example:
(1) I have a work file that I am going to access every day for the next 30 days.
(2) After 30 days, I may only need to access that file once a week for the 60 next days.
(3) After which (90 days total) I will probably never access the file again but want to keep it just in case.





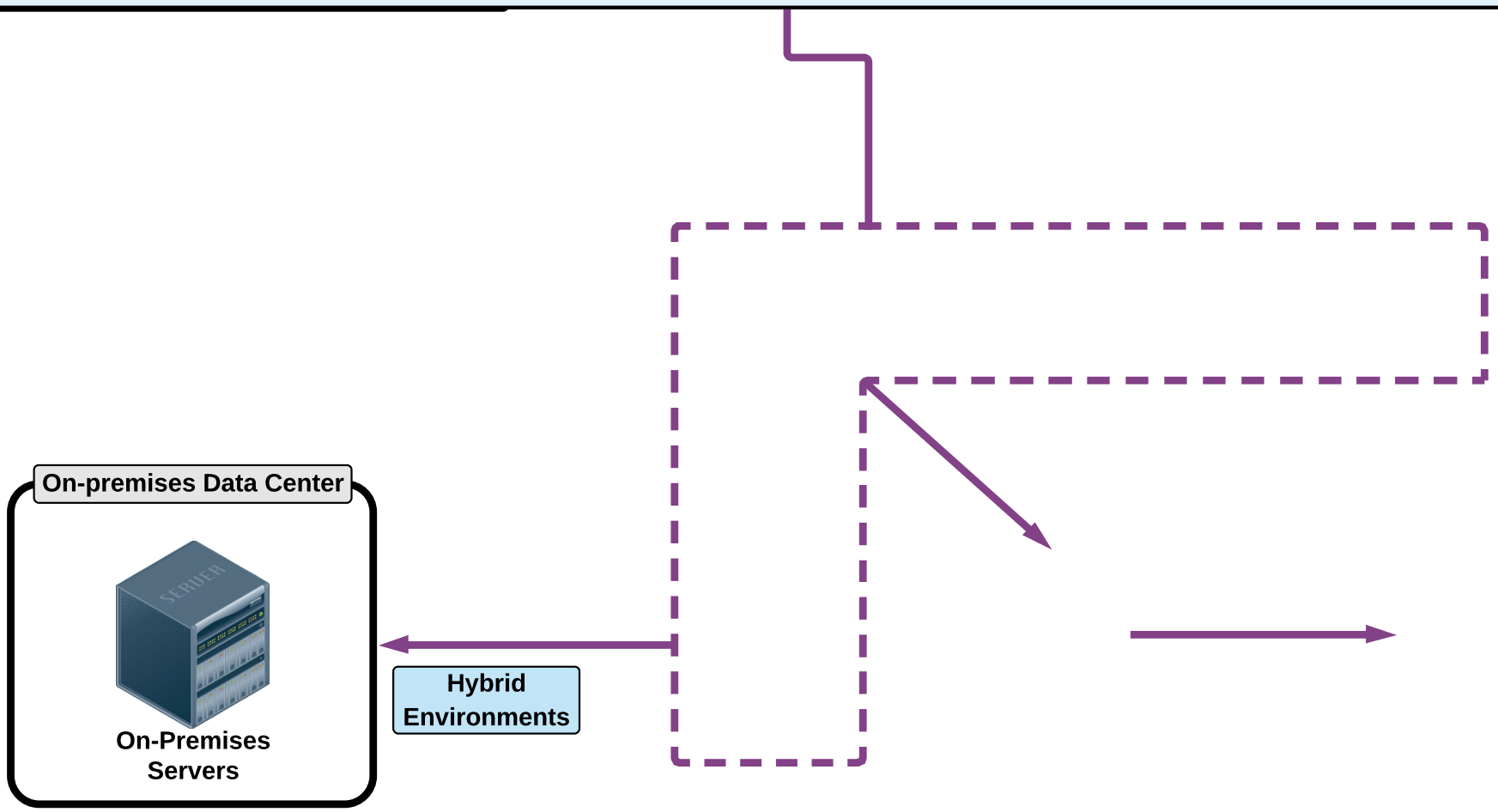
Simple Storage Service (S3):

X

AWS Import/Export:

- AWS Import/Export gives the ability to take on-premise data and physically snail mail it to AWS (using a device that you own).
- AWS will import the data to either S3, EBS, or Glacier within one business day of the physical device arriving at AWS.

- Benefits:
  - Off-site backup policy
  - Quickly migrate LARGE amounts of data to the cloud (up to 16TB per job)
  - Disaster recovery (AWS will even take s3 data and ship it back to you)



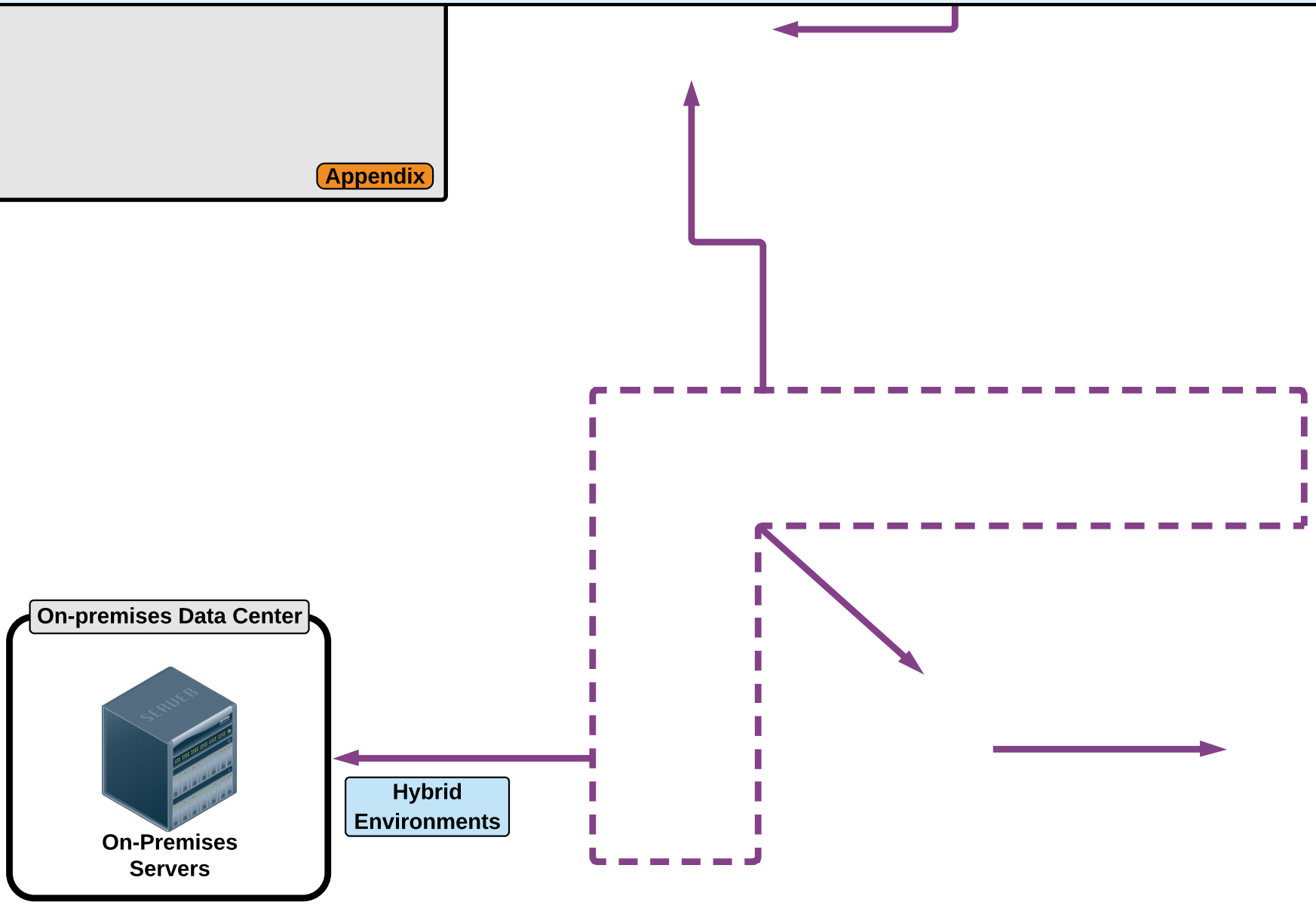
Simple Storage Service (S3):

X

Single Operation Upload:

- A single operation upload is "traditional" upload where you upload the file in one part.
- A single operation upload can upload a file up to 5GB in size, however any file over 100MB should use multipart upload.

Appendix

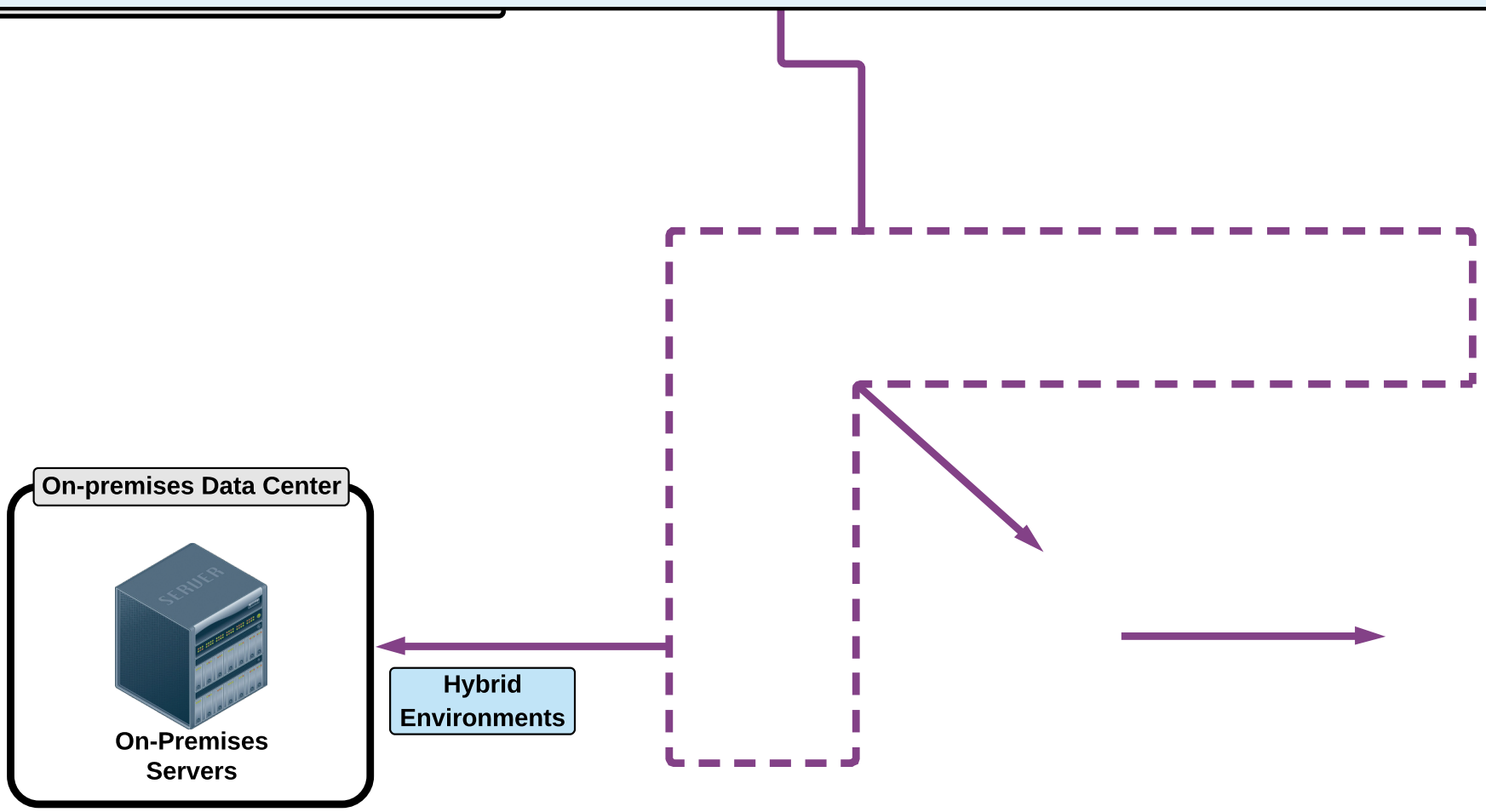


Simple Storage Service (S3):

X

Multipart Upload:

- Multipart upload allows you to upload a single object as a set of parts.
- Allows for uploading parts of a file concurrently
- Allows for stopping/resuming file uploads
- If transmission of any part fails, you can retransmit that part without affecting other parts.
- After all parts of your object are uploaded, Amazon S3 assembles these parts and creates the object.
- Required for objects 5GB and large, and ***highly suggested*** for use when objects are 100MB and larger.
- Can be used to upload a file up to 5TB in size.

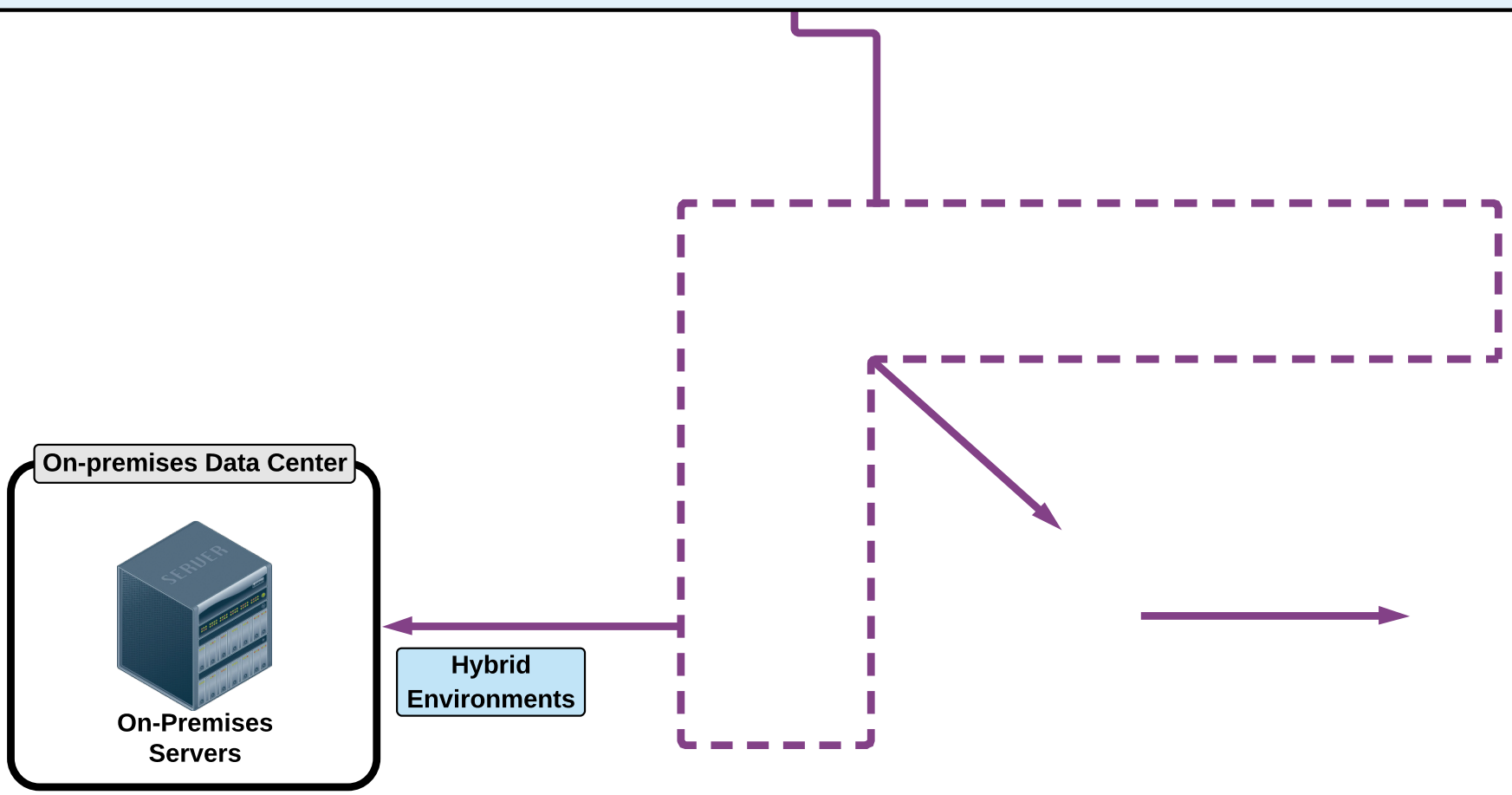


Glacier:

- Amazon Glacier is an ***archival*** storage type.
- Used for data that is NOT accessed frequently.
- "Check out" and "check in jobs" can take several hours, meaning how long it can take for the data to be changed and/or retrieved.
- Integrates with Amazon S3 lifecycle polices for easy archiving.
- Very inexpensive and cost effective archival storage solution.
- Glacier should NOT be used as a backup solution.

NOTE: Glacier now offers three levels of data retrieval (pricing varies):

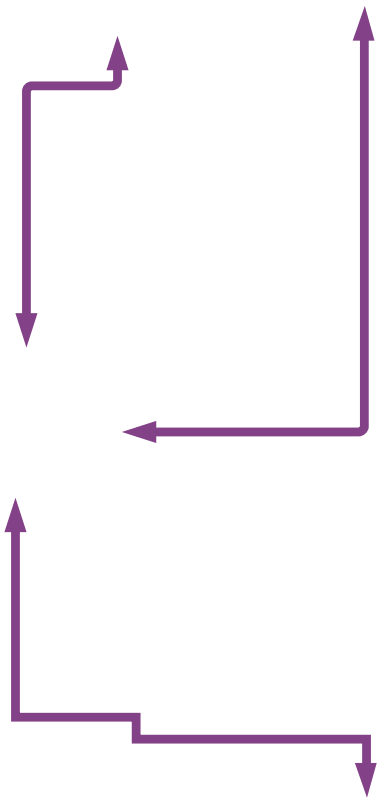
- ***Expedited***: 1-5 minutes
- ***Standard***: 3-5 hours
- ***Bulk***: 5-12 hours



Account & Services Layer

Physical & Networking Layer

Appendix



On-premises Data Center

On-Premises Servers

Hybrid Environments

Account & Services Layer

Physical & Networking Layer

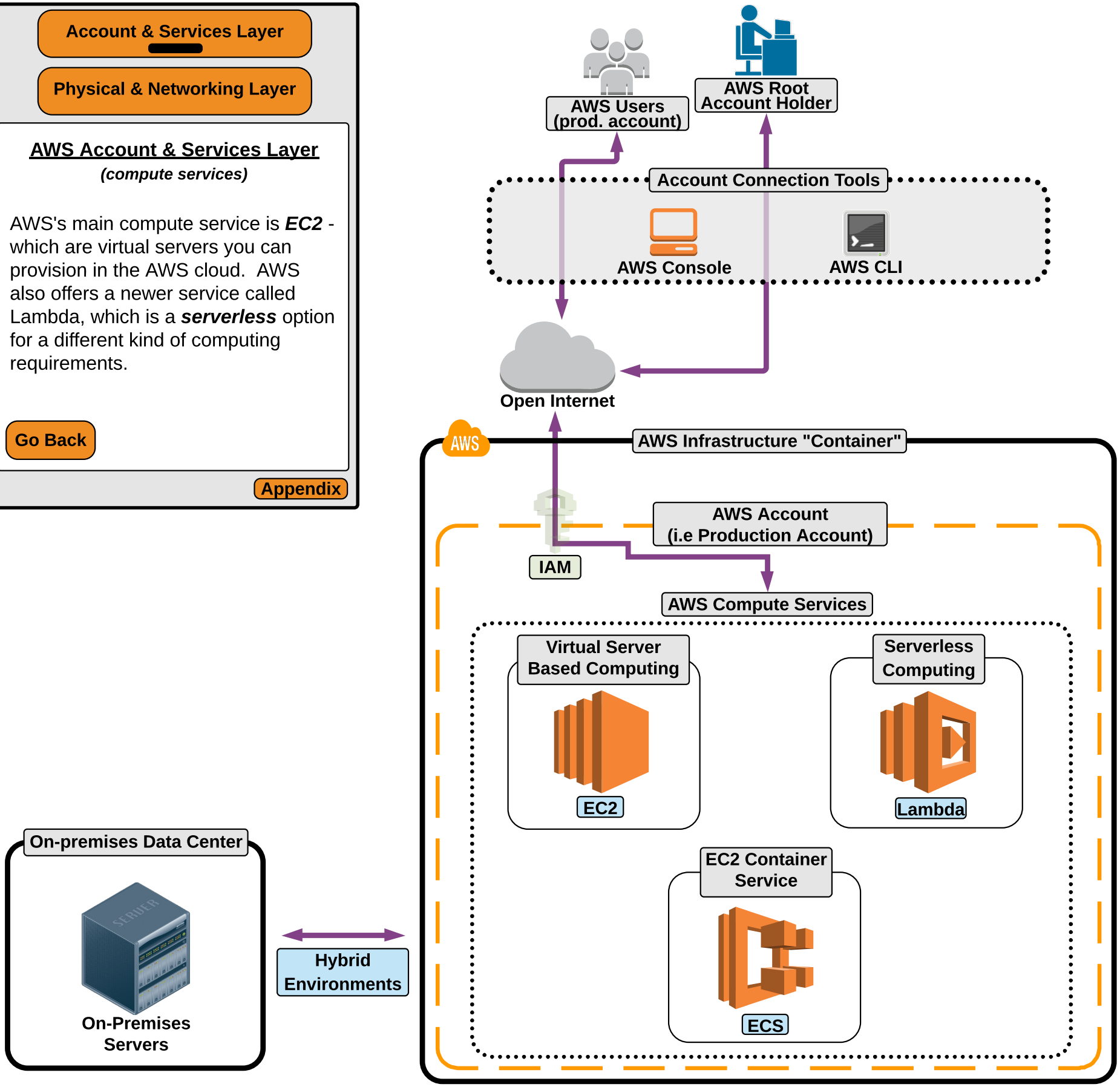
AWS Account & Services Layer

(compute services)

AWS's main compute service is ***EC2*** - which are virtual servers you can provision in the AWS cloud. AWS also offers a newer service called Lambda, which is a ***serverless*** option for a different kind of computing requirements.

Go Back

Appendix



Account & Services Layer

Physical & Networking Layer

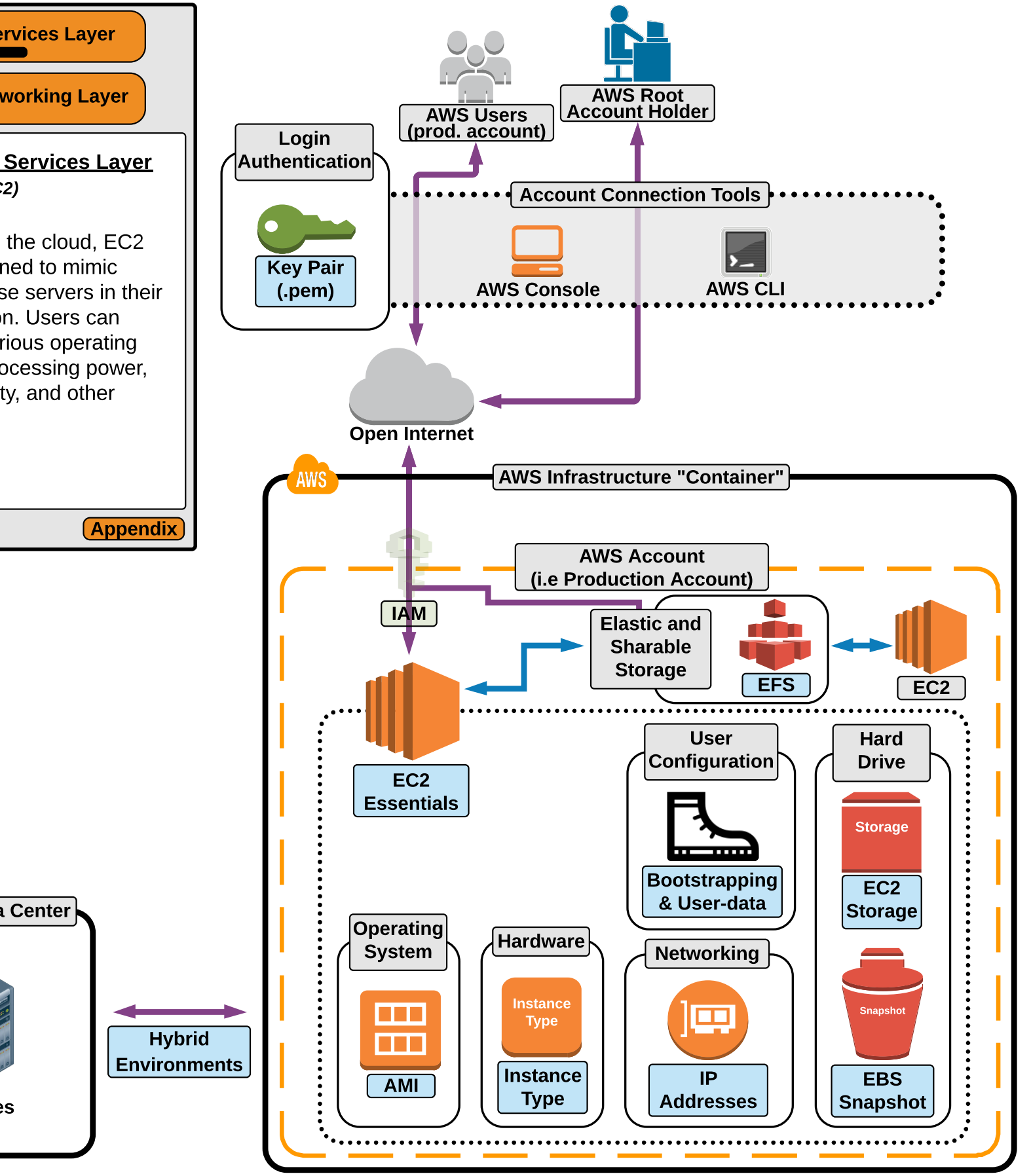
AWS Account & Services Layer

(EC2)

As virtual servers in the cloud, EC2 instances are designed to mimic traditional on-premise servers in their design and operation. Users can choose between various operating systems, level of processing power, storage type/capacity, and other customizations.

Go Back

Appendix



Account & Services Layer

Physical & Networking Layer

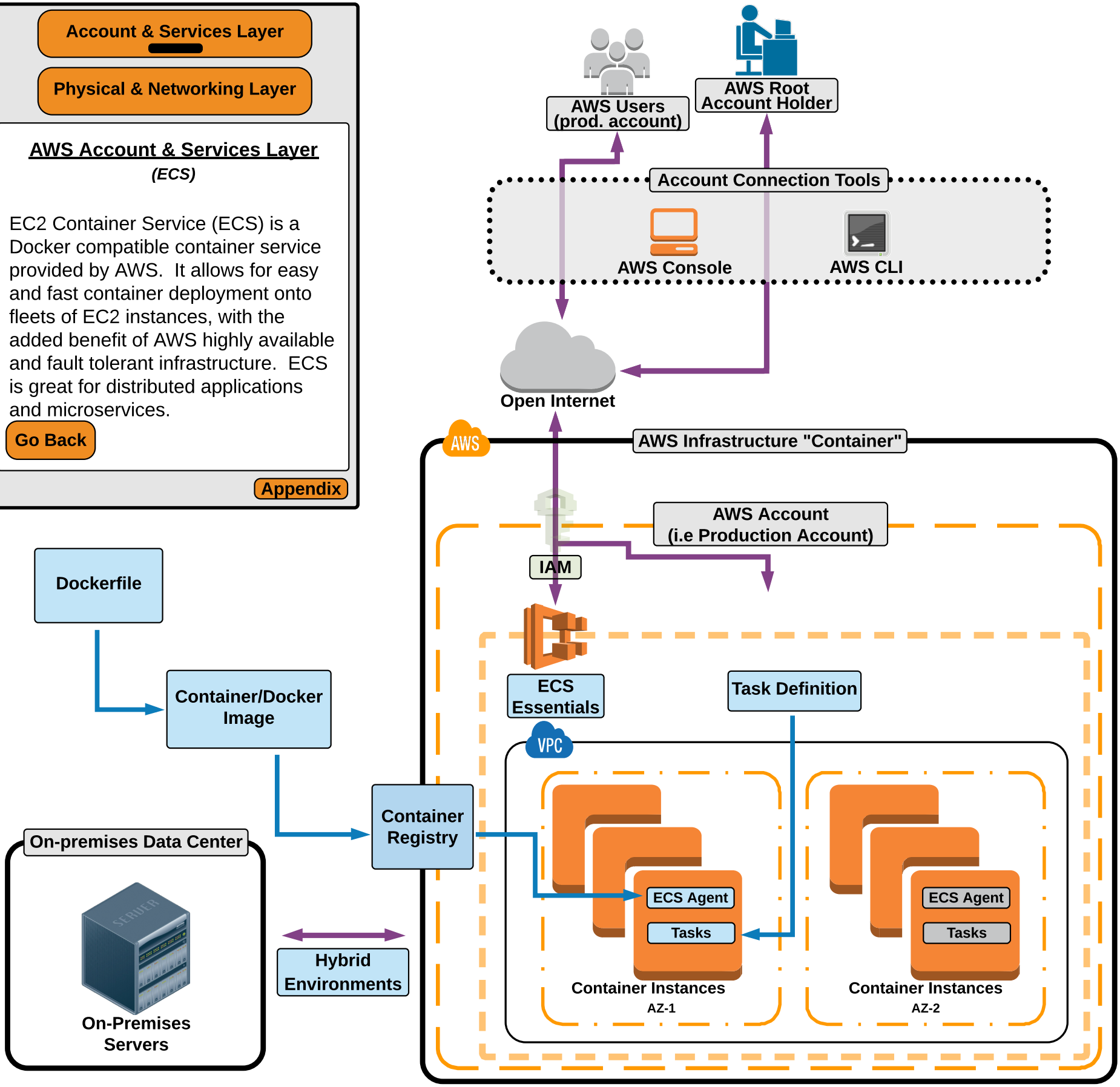
AWS Account & Services Layer

(ECS)

EC2 Container Service (ECS) is a Docker compatible container service provided by AWS. It allows for easy and fast container deployment onto fleets of EC2 instances, with the added benefit of AWS highly available and fault tolerant infrastructure. ECS is great for distributed applications and microservices.

Go Back

Appendix



Elastic Cloud Compute (EC2):

EC2 Key Pairs:

- EC2 key pairs are two cryptographically secure keys that are used by AWS to authenticate a client when logging into and EC2 instance.
- Each key pair consists of a public key and a private key.
- AWS stores the public key on the instance, and you are responsible for storing the private key.

To log into an EC2 instance, you must create and authenticate with a key pair.

- Linux instances have no password, and you use a key pair to log in (using SSH).
- With Windows instances, you use a key pair to obtain the administrator password and then log in using RDP (remote desktop protocol).

- During the creation process of an EC2 instance, you are required to either create a new key pair OR use an existing key pair - which will be tied to the instance.
- The private key is available for download and stored on your local device, however it is only available once (in the form of a ***.pem*** file).

For SSH login to linux instances:

- You will most likely need to change permissions on the .pem file before you can use it to login via SSH.
- This is done by running the following command "***chmod 400 [key name].pem***"

AWS Compute Services:

X

Lambda Essentials:

- Lambda is a "serverless" computing platform.
- Serverless means that you can *run code without provisioning or managing servers*.
  - So if you want to run code, you don't have to spin up an EC2 instance and install software, you can just create a "Lambda function", drop your code in it, and execute it.
- Lambda scales the required compute power automatically with your code.
- *You pay only for the compute time you consume (to the millisecond).*
- By default, it is highly available, fault-tolerant, scalable, elastic, and cost efficient.
- *Lambda integrates with many other AWS services.*
- *Current supported languages include:*
  - *Node.js*
  - *Java*
  - *C#*
  - *Python*

When should you use Lambda over Ec2?

- *Generally you want to use Lambda when you want to run code that is in response to events, such as*
  - *Changes to Amazon S3 buckets.*
  - *Updates to an Amazon DynamoDB table.*
  - *Custom events generated by your applications or devices.*

Elastic Cloud Compute (EC2):

EC2 Instance Type:

- Instances Types describe the "hardware" components that an EC2 instance will run on:
  - Compute power (processor/vCPU)
  - Memory (ram)
  - Storage Options/optimization (hard drive)
  - Network Performance (bandwidth)

- As an architect, it's important to use the proper instance type to handle your application's workload.

- There is a collection of preconfigured instance types that are grouped into families and types that you can choose from:

TYPE	FAMILY	Notes
t2	General Purpose	"Burstable" performance instances
m3	General Purpose	Nice Balance
c3/c4	Compute Optimized	For high traffic front end fleets, web servers
d2	Storage Optimized	For large-scale data warehouse or parallel file systems
i2	Storage Optimized	For large-scale data warehouse or parallel file systems
g2	GPU Optimized	For machine learning, high performance databases, rendering
p2	GPU Optimized	For machine learning, high performance databases, rendering
r3/r4	Memory Optimized	For databases, memcached, large deployments of enterprise applications
x1	Memory Optimized	For databases, memcached, large deployments of enterprise applications

Elastic Cloud Compute (EC2):

EC2 IP Addresses:

Private IP Address:

- All EC2 instances are automatically created with a PRIVATE IP address.
- The private IP address is used for internal (inside the VPC) communication between instances.

Public IP Address:

- When creating an EC2 instance, you have the option to enable (or auto-assign) a public IP address.
- A public IP address is required if you want the EC2 instance to have ***direct*** communication with resources across the open internet.
  - i.e. If you want to directly SSH into the instance or have it directly serve web traffic.
- Auto-assigning is based on the setting for the selected subnet that you are provisioning the instance in.

Elastic IP Address (EIP):

- An EIP is a static IPv4 address designed for dynamic cloud computing.
- An EIP is a public IPv4 address.
- With an EIP you can attach a public IP address to an EC2 instance that was created with only a private IP address OR....
- You can mask the failure of an instance or software by rapidly remapping the address to another instance in your account (i.e. detaching the EIP from one instance and attaching it to another).
- Attaching an EIP to an instance will replace it's default public IP address for as long as it is attached.

Elastic Cloud Compute (EC2):

Bootstrapping & User-Data/Meta-Data:

Bootstrapping:

- Refers to a self-starting process or set of commands without external input.
- With EC2, we can bootstrap the instance (during the creation process) with custom commands (such as installing software packages, running updates, and configuring other various settings)

User-Data:

- A step/section during the EC2 instance creation process where you can include your own custom commands via a script (i.e. a bash script)

- Here is an example of a bash script that will automate the process of updating the yum package installer, install Apache Web Server, and start the Apache service.

```
#!/bin/bash
yum update -y
yum install -y httpd
service httpd start
```

Viewing User-Data & Instance Meta-Data:

- When logged into an EC2 instance, you can view the instance user-data used during creation, or meta-data by executing one of the the following commands:
  - ***curl http://169.254.169.254/latest/user-data/*** (displays bootstrapping commands)
  - ***curl http://169.254.169.254/latest/meta-data/*** (displays AMI, instance type, etc)

Elastic Cloud Compute (EC2):

Elastic File System (EFS):

- EFS is a storage option for EC2 that allows for a ***scalable*** storage option.
- EFS storage capacity is ***elastic***.
  - The storage capacity will increase and decrease as you add or remove files.
  - Applications running on an EC2 instance using EFS will always have the storage they need, without having to provision and attach larger storage devices.
- EFS is fully-managed (no maintenance required)
- Supports the Network File System version 4.0 and 4.1 (NFSv4) protocols when mounting.
- Best performance when using an EC2 AMI with Linux kernel 4.0 or newer.

Benefits of EFS:

- The EFS file system can be accessed by one (or more) EC2 instances at the same time.
  - Shared file access across all your EC2 instances.
  - Application that span multiple EC2 instances can access the same data.
- EFS file systems can be mounted to on-premise servers (when connected to your VPC via AWS Direct Connect).
  - This allows you to migrate data from on-prem severs to EFS and/or use it as a backup solution.
- EFS can scale to petabytes in size, while maintaining low-latency and high levels of throughput.
- You pay only for the amount of storage you are using.

Security:

- Control file system access through POSIX permissions.
- VPC for network access control, and IAM for API access control.
- Encrypt data at rest using AWS Key Management Service (KMS).

When to use:

- Big Data and analytics
- Media processing workflows
- Web Serving & Content Management

Elastic Cloud Compute (EC2):

EBS Snapshots:

- Snapshots are point-in-time backups of EBS volumes that are stored in S3.

Snapshot properties:

- Snapshots are incremental in nature.
- A snapshot only stores the changes since the most recent snapshot, thus reducing costs (by only having to pay for storage for the “incremental changes” between snapshots).
- However, if the "original" snapshot is deleted, all data is still available in all the other snapshots.
- Even though snapshot storage only charges you for the amount of incremental data in each snapshot all prior data is still there.
- Snapshots can be used to create fully restored EBS volumes

A few other important snapshot notes:

- Frequent snapshots of your data increases data durability - so highly recommended.
- When a snapshot is being taken against the EBS volume, it can degrade performance so snapshots should occur during non-production or non-peak load hours.

Elastic Cloud Compute (EC2):

EC2 Container Service (ECS) Essentials:

- ECS is a container management service that supports Docker.
- It allows you to easily create and manage a fleet of Docker containers on a cluster of EC2 instances.

Why use ECS/Containers?

- *Create distributed applications and Microservices:*
  - Create application architecture comprised of independent tasks or processes (micro services).
  - For example, you can have separate containers for various components of your application
    - Webserver
    - Application server
    - Message queue
    - Backend Servers
  - This allows you to start, stop, manage, monitor, and scale each container independently.

- *Batch and ETL Jobs:*
  - Package batch and ETL jobs into containers and deploy them into an shared EC2 cluster(s).
  - Run different versions of the same job or multiple jobs on the same cluster.
  - Share cluster capacity with other processes and or grow job dynamically on-demand to improve resource utilization.

- *Continuous Integration and Deployment:*
  - By using Dockers Image versioning, you can use containers for continuous integration and deployment.
  - Build processes can pull, build, and create a Docker Image that can be deployed into your containers.
  - This allows you to avoid an application from working in a developer environment and not working in a production environment because the Docker daemon is the same across all environments.

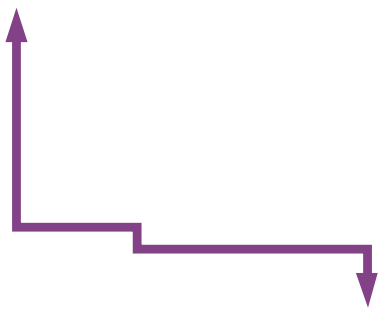
Elastic Cloud Compute (EC2):

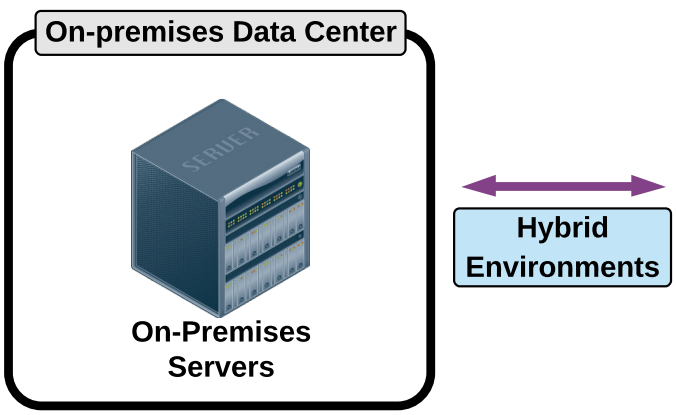
X

Dockerfile:

- A plain text file (script) that specifies all of the components that are included in the container.
- Basically, it's the instructions for what will be placed inside a given container.

Appendix





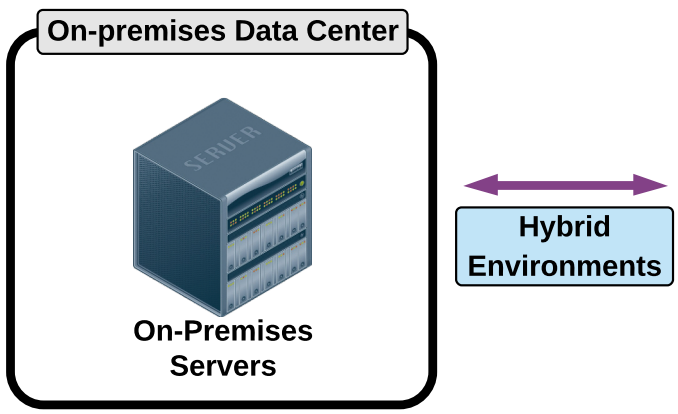
Elastic Cloud Compute (EC2):

X

Container Registry:

- A container registry is a repository where container/docker images are stored and accessed from when needed.
- A container registry can be:
  - Located on AWS via the ECR service (EC2 Container Registry)
  - A 3rd party repository like Docker Hub
  - Self-hosted registry.

Appendix



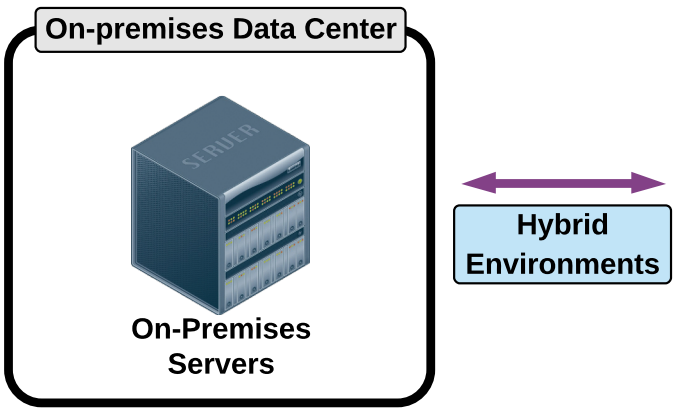
Elastic Cloud Compute (EC2):

X

ECS Task Definition:

- A JSON formatted text file that contains the "blueprint" for your application, including:
  - Which container/docker image to use.
  - The repository (container registry) the image is located in.
  - Which ports should be open on the container instance.
  - What data volumes should be used with the containers.

Appendix



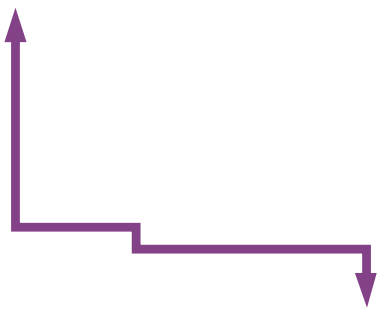
Elastic Cloud Compute (EC2):

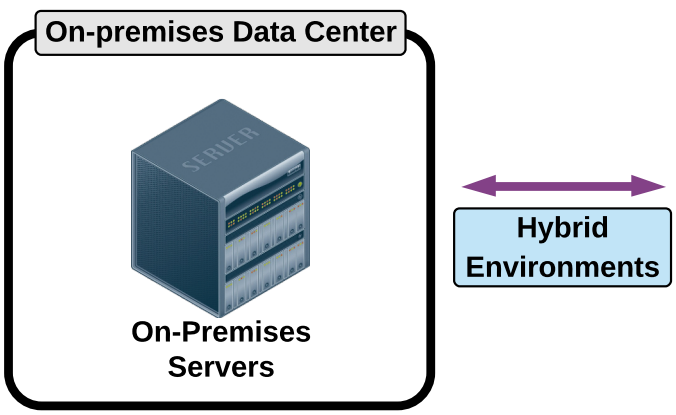
X

ECS Agent:

- The ECS Agent runs on each EC2 instance in the ECS cluster.
- It communicates information about the instances to ECS, including:
  - Running tasks
  - Resource Utilization
- The ECS Agent is also responsible for starting/stopping tasks (when told to by ECS).

Appendix





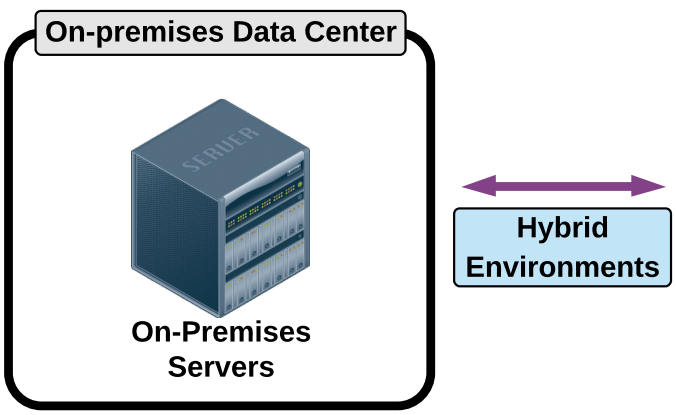
Elastic Cloud Compute (EC2):

X

ECS Task:

- An ECS Task is the actual representation of the Task Definition on an EC2 instance inside of your container cluster.
- The ECS Agent will start/stop these tasks based on instruction/schedule.

Appendix



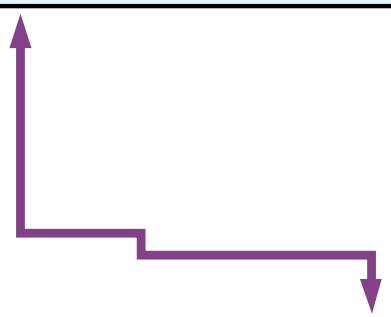
Elastic Cloud Compute (EC2):

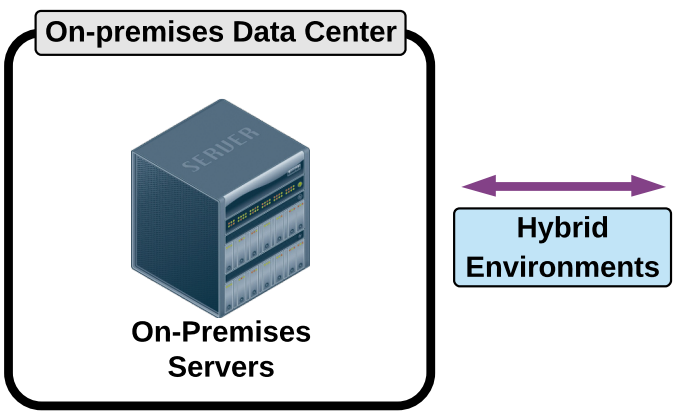
X

Container/Docker Image:

- A container/Docker image is built from a Dockerfile.
- The container/Docker image contains all the downloaded software, code, runtime, system tools, and libraries (as outlined in the Dockerfile).
  - i.e. If the Dockerfile specifies PHP to be downloaded and installed, then the container/Docker Image will have PHP downloaded an installed.

Appendix





Elastic Cloud Compute (EC2):

X

EBS Basics | EBS Types | Instance Store

Elastic Cloud Compute (EC2):

X

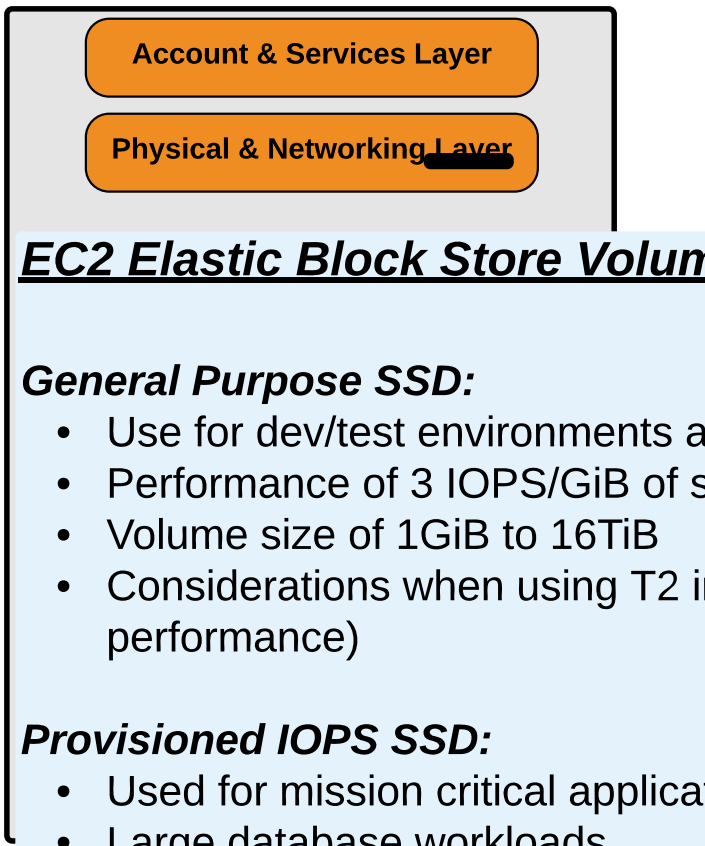
Essentials | Buying Options | Shared Responsibility | Placement Groups

Elastic Cloud Compute (EC2):

X

AMI Essentials

Virtualization



EC2 Elastic Block Store Volumes:

General Purpose SSD:

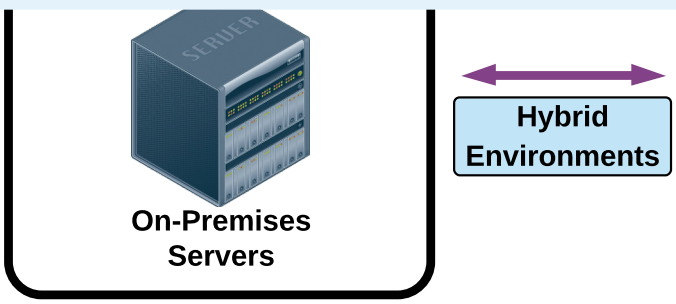
- Use for dev/test environments and smaller DB instances
- Performance of 3 IOPS/GiB of storage size (burstable with baseline performance)
- Volume size of 1GiB to 16TiB
- Considerations when using T2 instances with SSD root volumes (burstable vs. baseline performance)

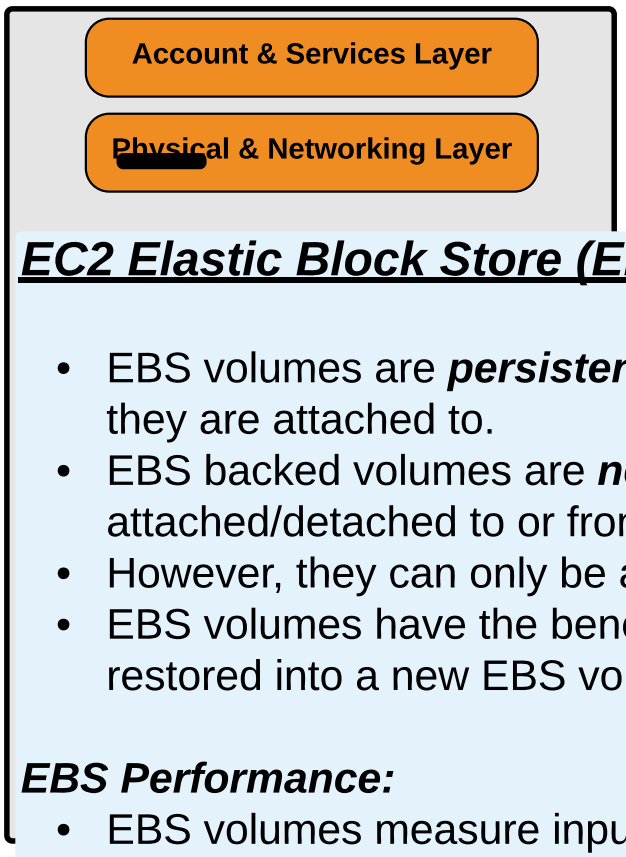
Provisioned IOPS SSD:

- Used for mission critical applications that require sustained IOPS performance
- Large database workloads
- Volume size of 4GiB to 16TiB
- Performs at provisioned level and can provision up to 20,000 IOPS per volume

Magnetic:

- Low storage cost
- Used for workloads where performance is not important or data is infrequently accessed
- Volume size of Min 1GiB Max 1 TiB





EC2 Elastic Block Store (EBS) Basics:

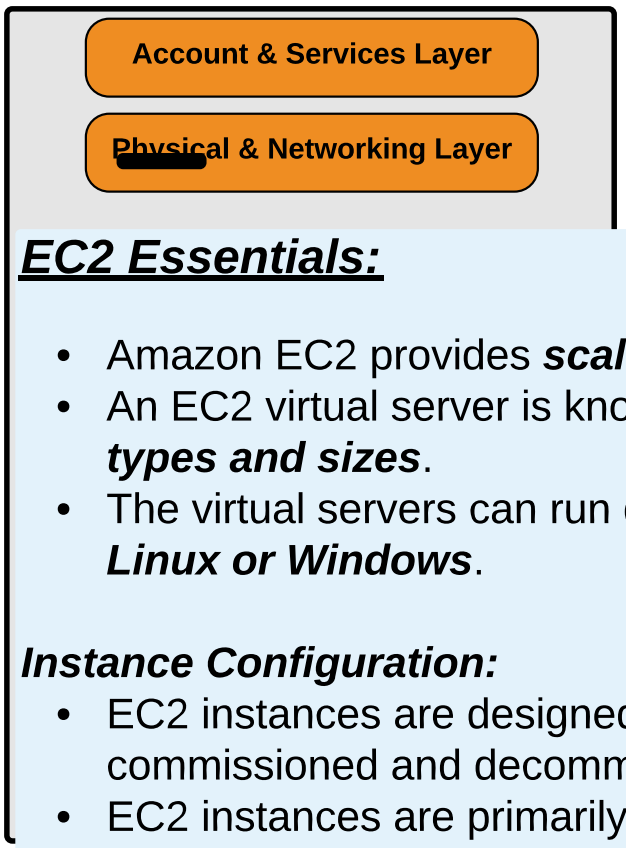
- EBS volumes are ***persistent***, meaning that they can live beyond the life of the EC2 instance they are attached to.
- EBS backed volumes are ***network attached storage***, meaning they can be attached/detached to or from various EC2 instances.
- However, they can only be attached to ONE EC2 instance at a time.
- EBS volumes have the benefit of being backed up into a ***snapshot*** - which can later be restored into a new EBS volume.

EBS Performance:

- EBS volumes measure input/output operations in IOPS:
  - IOPS are input/output operations per second
  - AWS measures IOPS in ***256KB chunks*** (or smaller)
  - Operations that are greater than 256KB are separated into individual 256KB chunks
  - For example, A 512KB operation would count as 2 IOPS
- The type of EBS volume you specify greatly influences the I/O performance (IOPS) your device will receive.
- It is important as architects to understand if your application requires more (or less) I/O when selecting an EBS volume.
- Even volumes with "***provisioned IOPS***" may not produce the performance you expect. If this is the case, an EBS optimized instance type is required, which prioritizes EBS traffic.

Initializing EBS Volumes:

- ***New*** EBS volumes no longer need to be "pre-warmed".
- ***New*** volumes will receive their maximum performance at the moment they are created.
- Volumes created from an EBS snapshot must be ***initialized***.
- Initializing occurs the first time a storage block on the volume is read - and the performance impact can be impacted by up to 50%.
- You can avoid this impact in production environments by manually reading all the blocks.



EC2 Essentials:

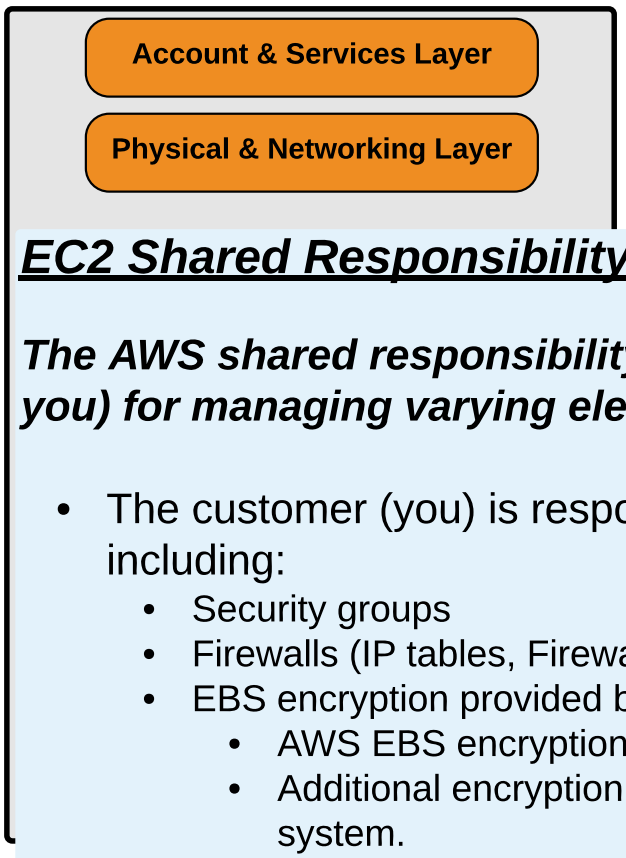
- Amazon EC2 provides ***scalable virtual servers*** in the cloud.
- An EC2 virtual server is known as an "***instance***" and can be made up of different ***instance types and sizes***.
- The virtual servers can run different operating systems, but most commonly run a flavor of ***Linux or Windows***.

Instance Configuration:

- EC2 instances are designed to mimic traditional on-premise servers, but with the ability to be commissioned and decommissioned on-demand for easy scalability and elasticity.
- EC2 instances are primarily comprised of the follow components:
  - ***Amazon Machine Image*** (AMI): The operating system (and other settings).
  - ***Instance Type***: The hardware (compute power, ram, network bandwidth, etc).
  - ***Network interface*** (public, private, or elastic IP addresses).
  - ***Storage:*** The instances "hard drive" (including two options).
    - Elastic Block Store (EBS) - which is "network persistent storage"
    - Instance Store - which is "ephemeral storage"

Other Important EC2 Facts:

- A security group must be assigned to an instance during the creation process.
- Each instance must be placed into an existing VPC, availability zone and subnet.
- Automated (bootstrapping) custom launch commands can be passed into the instance during launch via "user-data" scripts.
- "Tags" can be used to help name and organize provisioned instances.
- Encrypted key-pairs are used to manage login authentication.
- There are limits on the amount of instances you can have running in a region at any particular time.

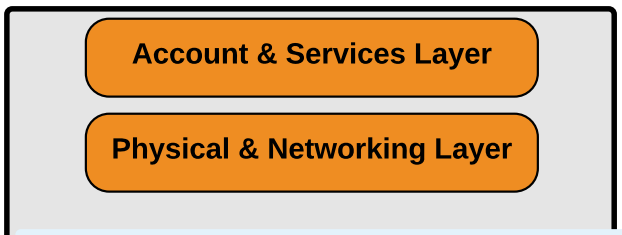


EC2 Shared Responsibility Model:

The AWS shared responsibility model describes the responsibilities of both parties (AWS & you) for managing varying elements of AWS resources.

- The customer (you) is responsible for managing the software level security on instances, including:
  - Security groups
  - Firewalls (IP tables, Firewalld, etc)
  - EBS encryption provided by AWS (snapshots can also use EBS encryption)
    - AWS EBS encryption utilizes AWS Key Management Service.
    - Additional encryption methods include encrypting the entire file system using an encrypted file system.
  - Applying an SSL Certificate to the ELB (Elastic Load Balancer).

- AWS is responsible for managing the hypervisor and physical layer of security for EC2:
  - DDOS protection
  - Port scanning protection (*not allowed even in your own environment without permission from AWS*)
  - Ingress network filtering



EC2 Placement Groups:

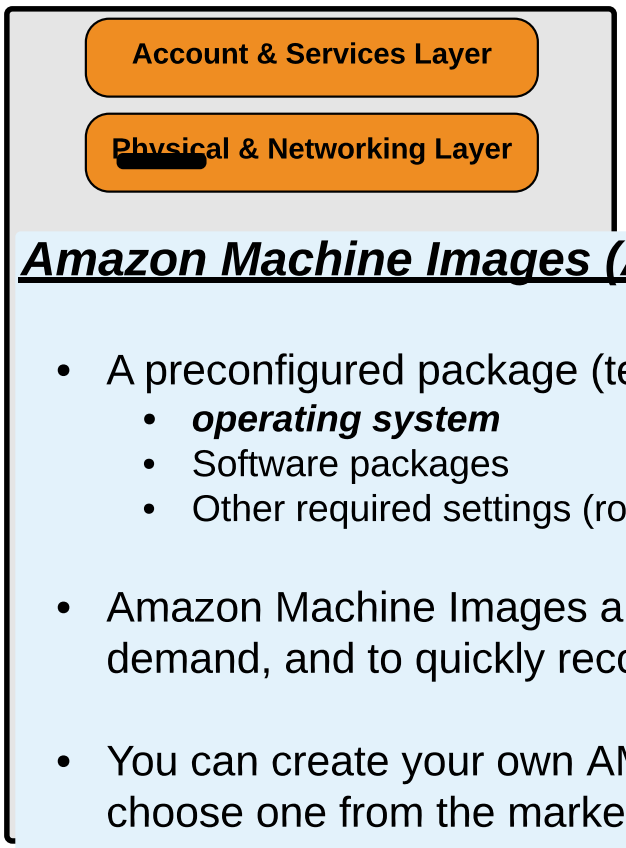
- A placement group is a cluster of instances within the same availability zone.
- Used for applications that require an extremely low latency network between them.
- AWS attempts to place all the instances as close as physically possible in the data center to reduce latency.
- Instances within a placement group have a low-latency, 10 Gbps network connection between them.
- Instances that are in the placement group need to have enhanced networking in order to maximize placement groups.

Troubleshooting Placement Groups:

- If an instance in a placement group is stopped, once it is started again it will continue to be a member of the placement group.
- It is suggested to launch all the required instances within a placement group in a single request, and that the same instance type is used for all instances within the placement group.
- It is possible, if more instances are added at a later time to the placement group OR if a placement group instance is stopped and started again, to receive an “insufficient capacity error”.
    - Resolve the capacity error by stopping all instances in the member group and attempting to start them again.

Important facts about Placement Groups:

- Instances not originally launched/created in the placement group cannot be moved into the placement group.
- Placement groups cannot be merged together.
- A placement group cannot span multiple availability zones.
- Placement group names must be unique within your own AWS account.
- Placement groups can be “connected”.
- Instances must have 10 gigabit network speeds in order to take advantage of placement groups (proper instance type).



Amazon Machine Images (AMIs):

- A preconfigured package (template) required to launch and EC2 instance; includes an:
  - ***operating system***
  - Software packages
  - Other required settings (root storage type & virtualization type)

- Amazon Machine Images are used with Auto Scaling to quickly launch new servers on demand, and to quickly recover from disaster.

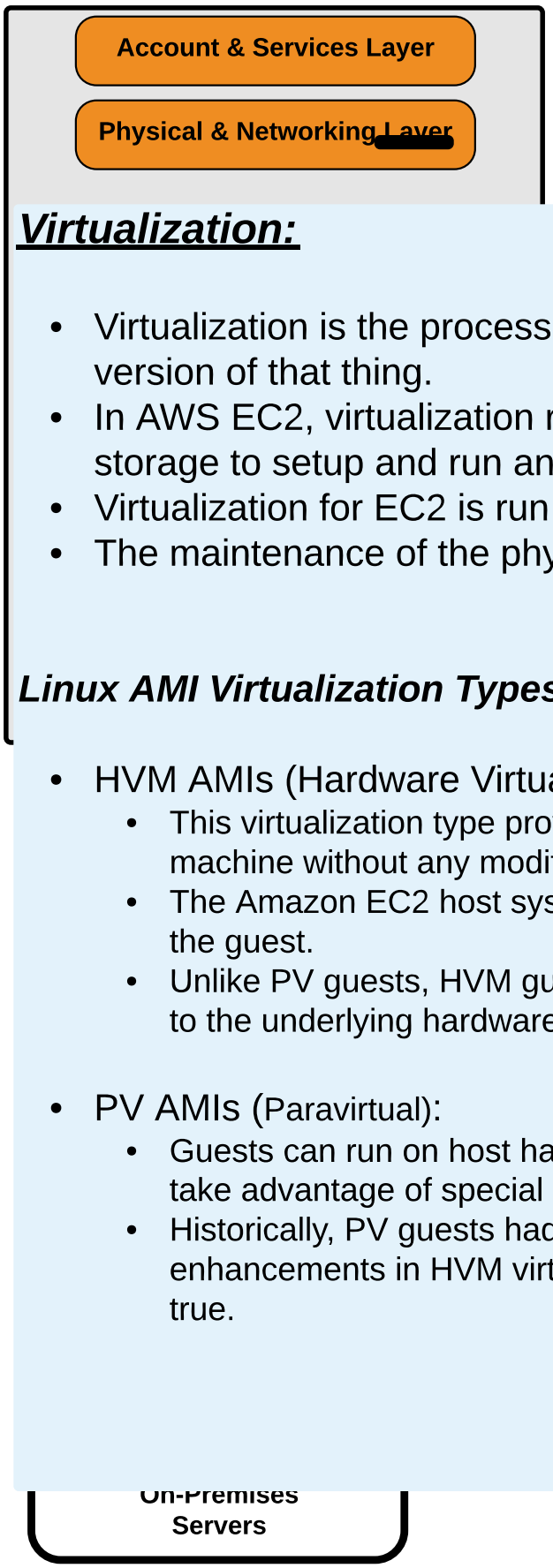
- You can create your own AMIs, choose from a list of free AWS/community provided AMIs, or choose one from the marketplace.

AMIs come in three main categories:

- Community AMIs:
  - Free to use
  - Generally with these AMIs you are just selecting the OS you want

- AWS Marketplace AMIs:
  - Pay to use
  - Generally comes packaged with additional, licensed software

- My AMIs:
  - AMIs that you create yourself



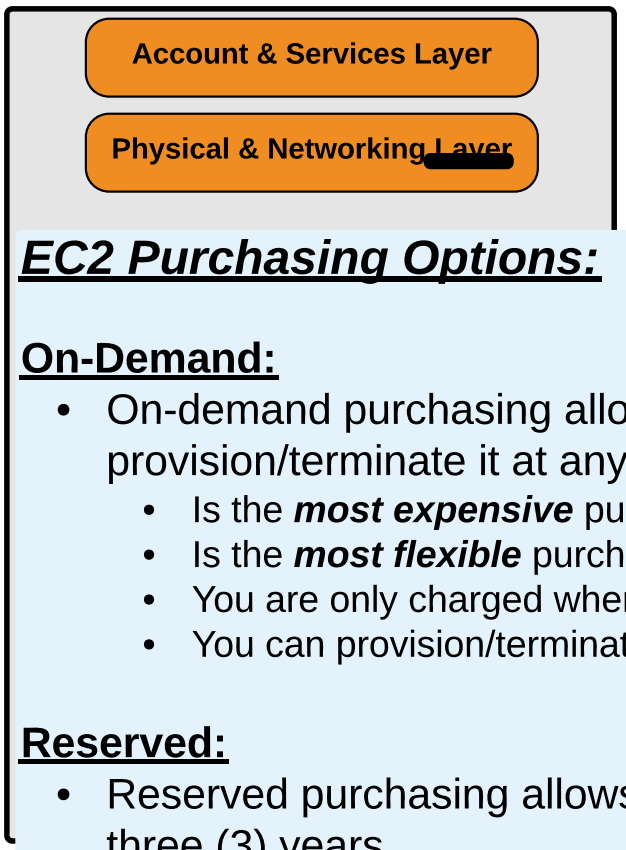
Virtualization:

- Virtualization is the process of creating a "virtual" version of something rather than the actual version of that thing.
- In AWS EC2, virtualization refers to using a "portion" of a server's computing power and storage to setup and run an operating system.
- Virtualization for EC2 is run using the Xen Hypervisor software.
- The maintenance of the physical AWS server and the Xen Hypervisor is handled by AWS.

Linux AMI Virtualization Types:

- HVM AMIs (Hardware Virtual Machine):
  - This virtualization type provides the ability to run an operating system directly on top of a virtual machine without any modification, as if it were run on the bare-metal hardware.
  - The Amazon EC2 host system emulates some or all of the underlying hardware that is presented to the guest.
  - Unlike PV guests, HVM guests can take advantage of hardware extensions that provide fast access to the underlying hardware on the host system.

- PV AMIs (Paravirtual):
  - Guests can run on host hardware that does not have explicit support for virtualization, but they cannot take advantage of special hardware extensions such as enhanced networking or GPU processing.
  - Historically, PV guests had better performance than HVM guests in many cases, but because of enhancements in HVM virtualization and the availability of PV drivers for HVM AMIs, this is no longer true.



EC2 Purchasing Options:

On-Demand:

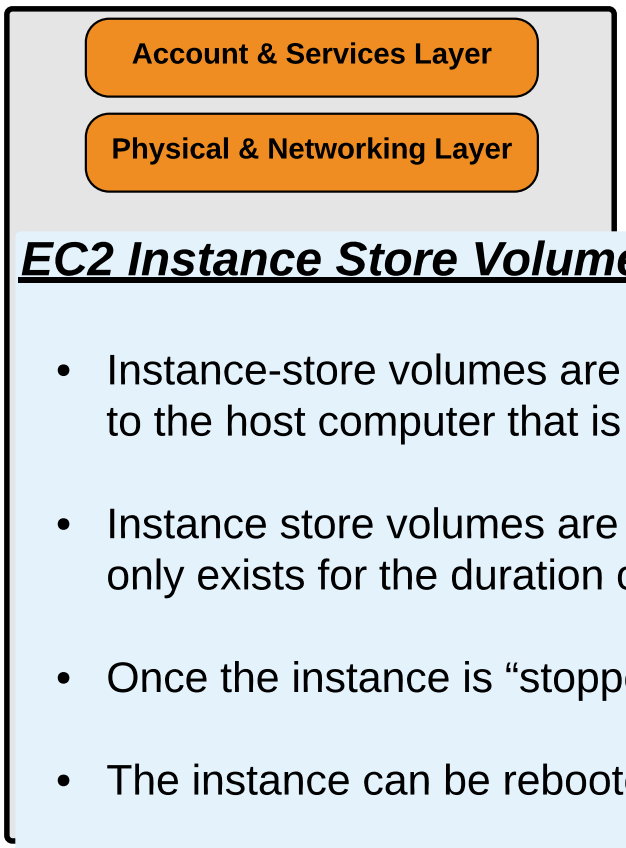
- On-demand purchasing allows you to choose any ***instance type*** you like and provision/terminate it at any time (on-demand).
  - Is the ***most expensive*** purchasing option.
  - Is the ***most flexible*** purchasing option.
  - You are only charged when the instance is ***running*** (and billed by the hour).
  - You can provision/terminate an on-demand instance at anytime.

Reserved:

- Reserved purchasing allows you to purchase an instance for a ***set time period*** of one (1) or three (3) years.
  - This allows for a ***significant price discount*** over using on-demand.
  - You can select to pay upfront, partial upfront, no upfront.
  - Once you buy a reserved instance, you own it for the selected time period and are r***esponsible for the entire price*** - regardless of how often you use it.

Spot:

- Spot pricing is a way for you to ***"bid"*** on an instance type, and only pay for and use that instance when the spot price is ***equal to or below*** your "bid" price.
  - This option allows Amazon to sell the use of ***unused instances***, for short amounts of time, at a ***substantial discount***.
  - ***Spot prices fluctuate*** based on supply and demand in the spot marketplace.
  - You are ***charged by hour (with conditions)***.
  - When you have an active bid, an instance is ***provisioned for you when the spot price is equal to or less than you bid price***.
  - A provisioned instances ***automatically terminate when the spot price is greater than your bid price***.
  - Bid on unused EC2 instances for “non production applications”.



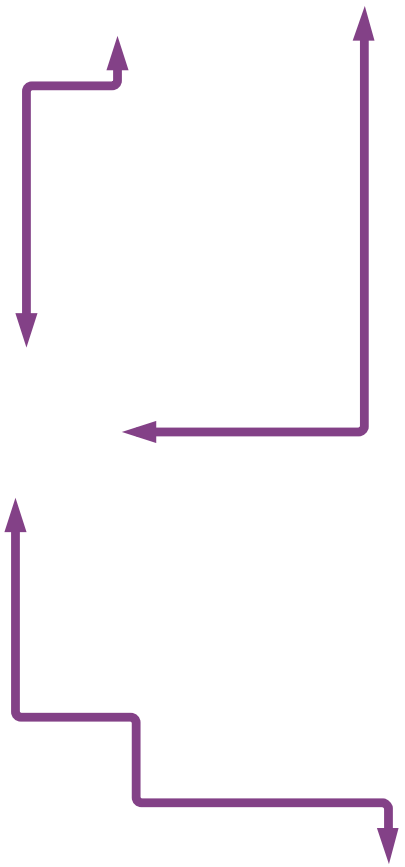
EC2 Instance Store Volumes:

- Instance-store volumes are virtual devices whose underlying hardware is physically attached to the host computer that is running the instance.
- Instance store volumes are considered ***ephemeral data***, meaning the data on the volumes only exists for the duration of the life of the instance.
- Once the instance is “stopped” or “shutdown” the data is erased.
- The instance can be rebooted and still maintain its ephemeral data.

Account & Services Layer

Physical & Networking Layer

Appendix



On-premises Data Center

On-Premises Servers

Hybrid Environments

Account & Services Layer

Physical & Networking Layer

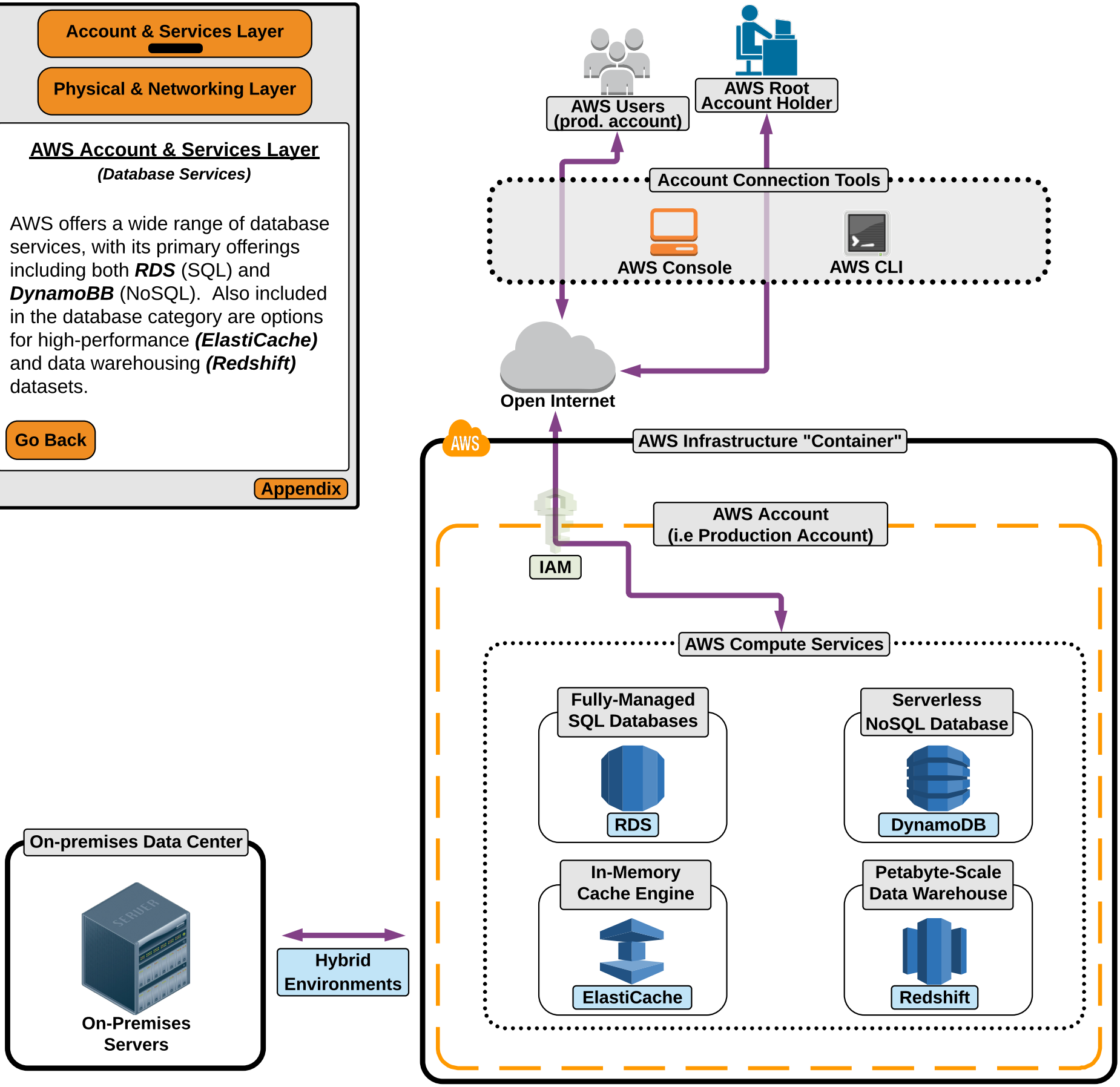
AWS Account & Services Layer

(Database Services)

AWS offers a wide range of database services, with its primary offerings including both ***RDS*** (SQL) and ***DynamoBB*** (NoSQL). Also included in the database category are options for high-performance ***(ElastiCache)*** and data warehousing ***(Redshift)*** datasets.

Go Back

Appendix



Account & Services Layer

Physical & Networking Layer

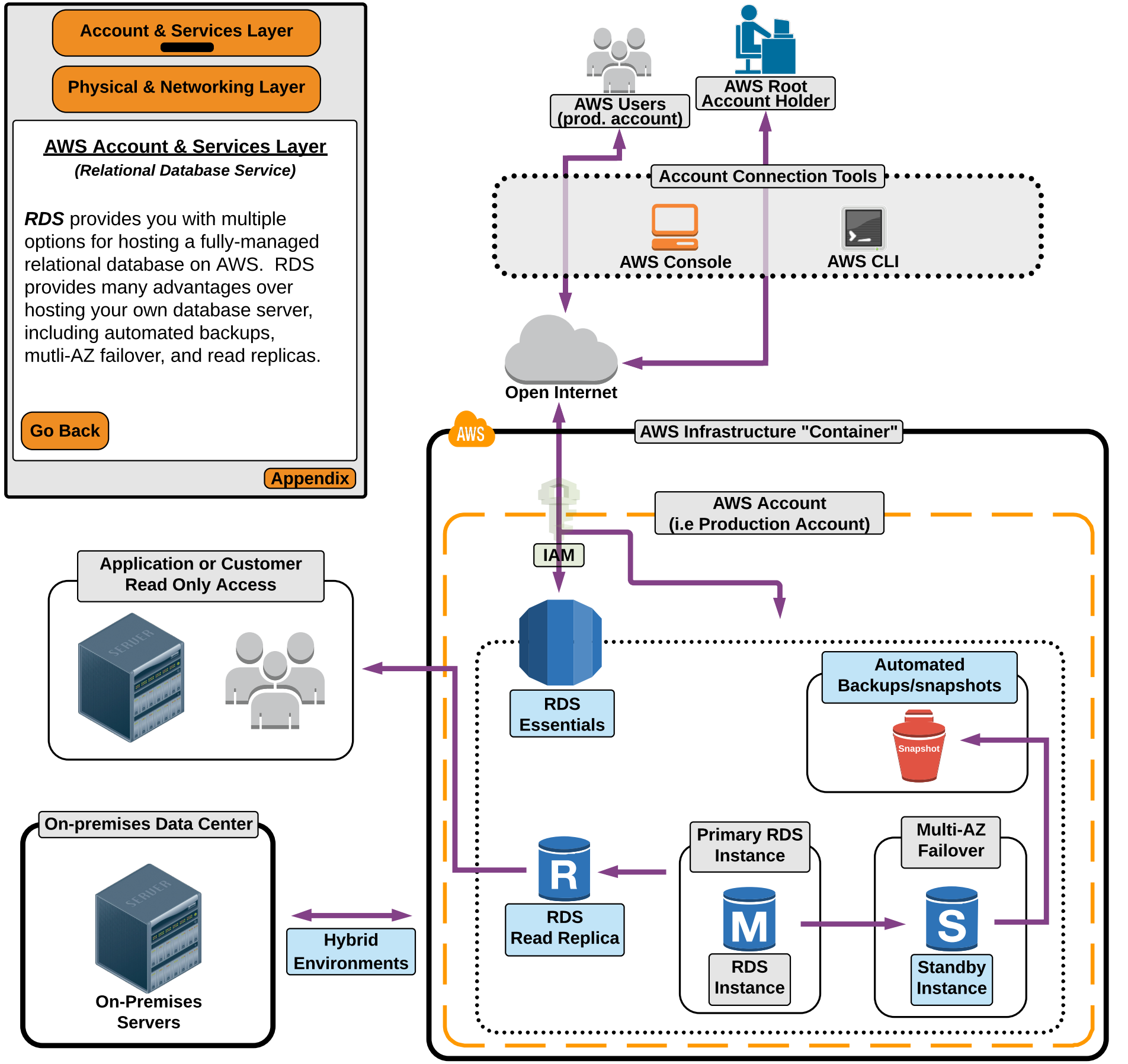
AWS Account & Services Layer

(Relational Database Service)

RDS provides you with multiple options for hosting a fully-managed relational database on AWS. RDS provides many advantages over hosting your own database server, including automated backups, mutli-AZ failover, and read replicas.

Go Back

Appendix

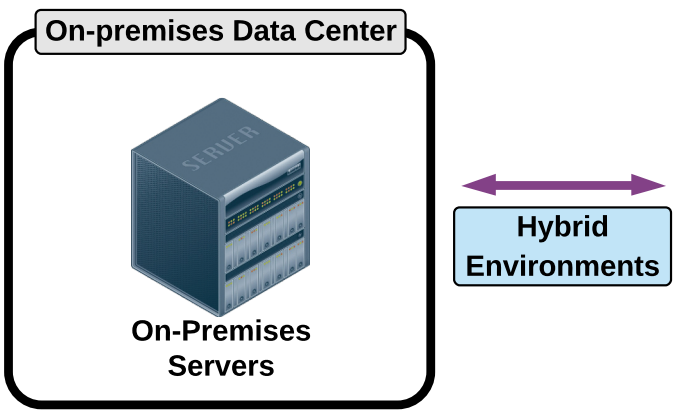


Databases:

ElastiCache Essentials:

- ElasticCache is a fully managed, in-memory cache engine.
- ElastiCache is used to improve database performance by caching results of queries that are made to a database.
- ElasticCache is great for large, high-performance or high-taxing queries - and can store them inside of a cache (***Elastic Cache Cluster***) that can be accessed later (instead of repeat request continually hitting the primary database).
- So it reduces load on the database, which increases performance.
- ElastiCache allows for managing web sessions, and also caching dynamic generated data.

- Available engines to power ElastiCache include:
  - ***Memcached*** (Mem-Cache-D)
  - ***Redis***
- Generally, the application needs to be built to work with either Redis or Memcached.
- Popular options like MySQL have Memcached plugins, which allow an application to easily work with ElastiCache (if using Memcached as the engine).



Databases:

RDS Essentials:

- RDS is a fully managed Relational Database Service:
  - Does not allow access to the underlying operating system (fully-managed).
  - You connect to the RDS database server in the same way you would connect to a traditional on-premise database instance (i.e MySQL command line).
  - RDS has the ability to provision/resize hardware on demand for scaling.
  - You can enable Multi-AZ deployments for backup and high available solutions.
  - Utilize Read Replicas (MySQL/PostgreSQL/Aurora) - to help offload hits on your primary database.
  - Relational databases are databases that organize stored data into tables.
  - The associated tables have defined relationships between them.

- Databases Supported By RDS:
  - MySQL
  - MariaDB
  - PostgreSQL
  - Oracle
  - MS SQL Server
  - ***Aurora***:
    - Is a home grown Relational Database that has been forked from, and fully compatible with MySQL.
    - It has fives times better performance then MySQL. and a lower price point than commercial databases.

- ***Benefits*** of running RDS instead of a database on your own instance:
  - Automatic minor updates
  - Automatic backups (point-in-time snapshots)
  - Not required to managed operating system
  - Multi-AZ with a single click
  - Automatic recovery in event of a failover

Databases:

X

RDS Read Replicas:

- Read replicas are asynchronous copies of the primary database that are used for read only purposes (only allow "read connections").
- When you write new data to the primary database, AWS copies it for you to the read replica.
- You can create, and have multiple read replicas for a primary database.
- Read replicas can be created from other read replicas (so no performance hit on the primary database)
- MySQL, MariaDB, PostgreSQL, and Aurora currently support read replicas.
- You can monitor replication lag using CloudWatch.

Benefits of using Read Replicas

- Read replicas allow for all read traffic to be redirected from the primary database to the read replica. This will greatly improve performance on the primary database.
- Read replicas allow for elasticity in RDS - you can add more read replicas as demand increases.
- You can promote a read replica to a primary instance
- MySQL:
  - Replicate for importing/exporting data to RDS
  - Can replicate across regions

When should you use Read Replicas?

- High volume, non-cached database read traffic (elasticity).
- Running business function such as data warehousing.
- Importing/Exporting data into RDS.
- Rebuilding indexes:
  - Ability to promote a read replica to a primary instance

Databases:

RDS Multi-AZ Failover:

- Multi-AZ failover (***Automatic*** AZ-Failover) synchronously replicates data to a backup (stand-by) database instance located in another availability zone (but in the same region).

- In the event of:
  - Service outage in an availability zone
  - Primary DB instance failure
  - Instance server type is changed
  - Manual failover initiated
  - Updating software version
  - ***AWS will automatically switch the CNAME DNS record from the primary instance to the stand-by instance.***

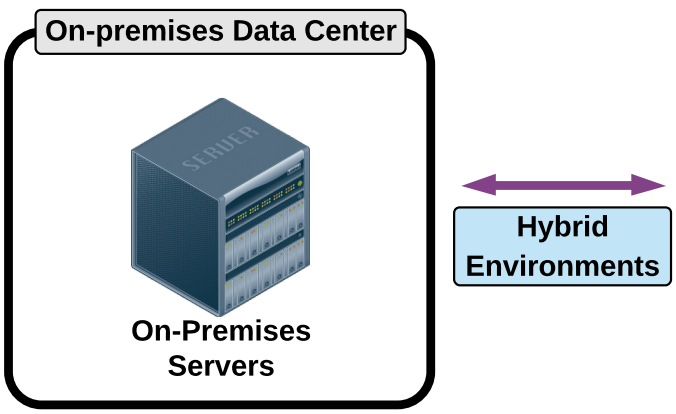
- RDS backups are taken against the stand-by instance to reduce I/O freezes and slow down IF multi-az is enabled.

- In order for multi-az to work, your primary database instance must be launched into a "***subnet group***".
  - NOTE: An RDS instance must be launched into a subnet (inside a VPC), just like an EC2 instance. So the same security/connectivity rules, and highly available/fault tolerant concepts apply.

Databases:

RDS Backups:

- AWS provides automated point-in-time backups against the RDS database instance.
- Automated backups are deleted once the database instance is deleted and cannot be recovered (but you can take your own snapshots of backups before deleting).
- Backups on database engines only work correctly when the database engine is “transactional”, but do currently work for all supported database types.
- MySQL requires InnoDB for reliable backups.

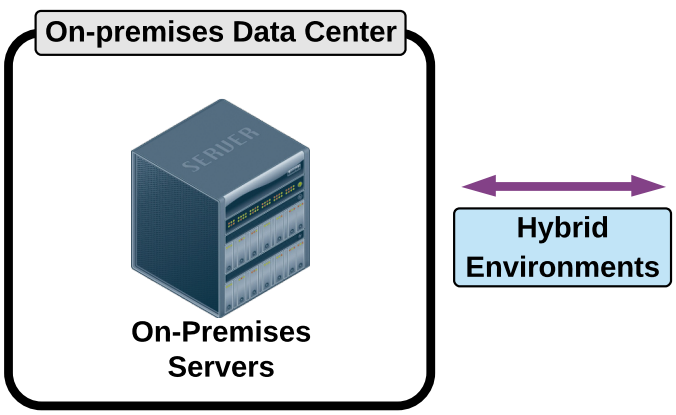


Databases:

RDS Monitoring with CloudWatch & Notifications:

- Subscribe to be notified when specific events take place:
  - Snapshots
  - Parameter group changes
  - Option changes
  - Security group changes

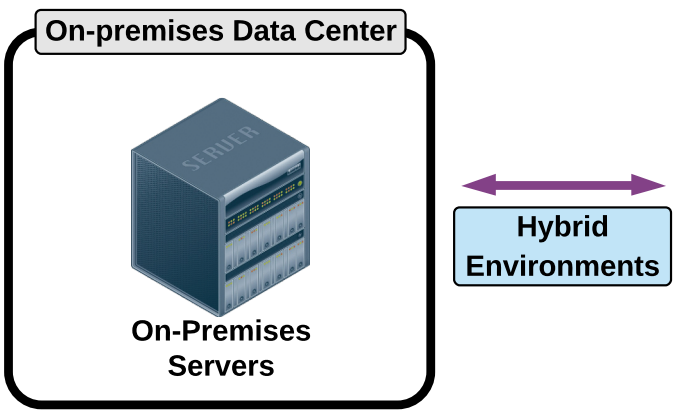
- Integrates with CloudWatch for hardware utilization and performance monitoring:
  - CPU Utilization – Freeable Memory – Swap Usage
  - Database Connections and binary log disk usage
  - Read/Write IOPS
  - Read Replicate latency
  - Read/Write throughput



Databases:

Redshift Essentials:

- Amazon Redshift is a petabyte-scale data warehousing service.
- It is fully-managed and scalable.
- Generally used for big-data analytics, and it can integrate with most popular business intelligence tools, including:
  - Jaspersoft
  - Microstrategy
  - Pentaho
  - Tableau
  - Business Objects
  - Cognos



Databases:

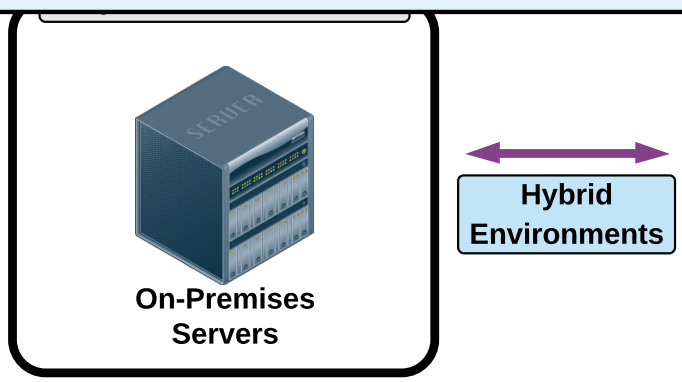
DynamoDB Essentials:

- DynamoDB is a fully-managed, NoSQL database service provided by AWS.
- It is similar to MongoDB, but is a home-grown AWS solution.
- Is schemaless, and uses a key-value store.
- You specify the required throughput capacity, and DynamoDB does the rest (being fully-managed).

- Being fully-managed means:
  - Service manages all provisioning (and scaling) of underlying hardware.
  - Fully distributed, and scales automatically with demand and growth.
  - Built as a fault tolerant highly available service.
    - On the back end, it fully synchronizes the data across all of the availability zones within the region you created the DynamoDB tables in.

- DynamoDB also easily integrates with other AWS services, such as Elastic MapReduce.
  - Can easily move data to a hadoop cluster in Elastic MapReduce.

- Popular use cases include:
  - IOT (storing meta data)
  - Gaming (storing session information, leaderboards)
  - Mobile (Storing user profiles, personalization)



Elastic Cloud Compute (EC2):

X

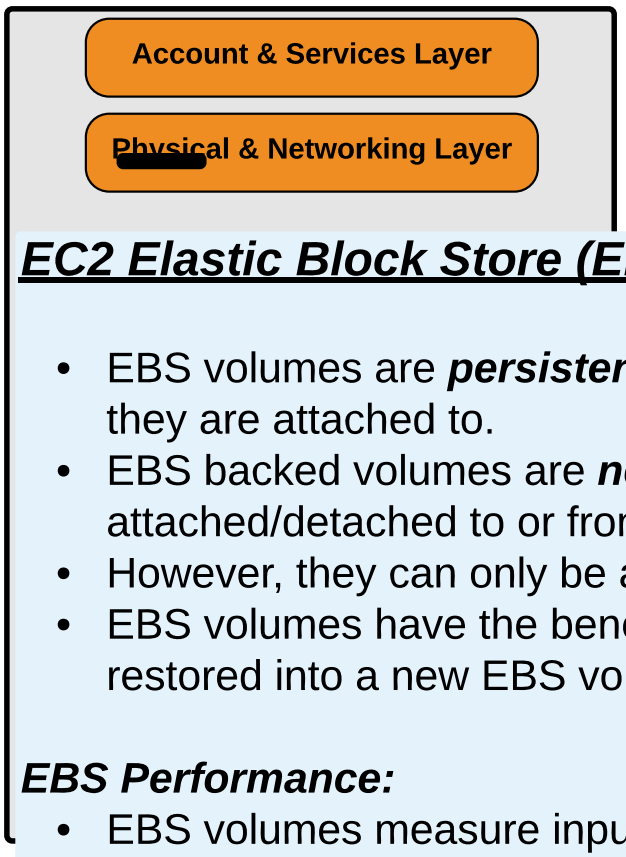
Essentials | Buying Options | Shared Responsibility | Placement Groups

Elastic Cloud Compute (EC2):

X

AMI Essentials

Virtualization



EC2 Elastic Block Store (EBS) Basics:

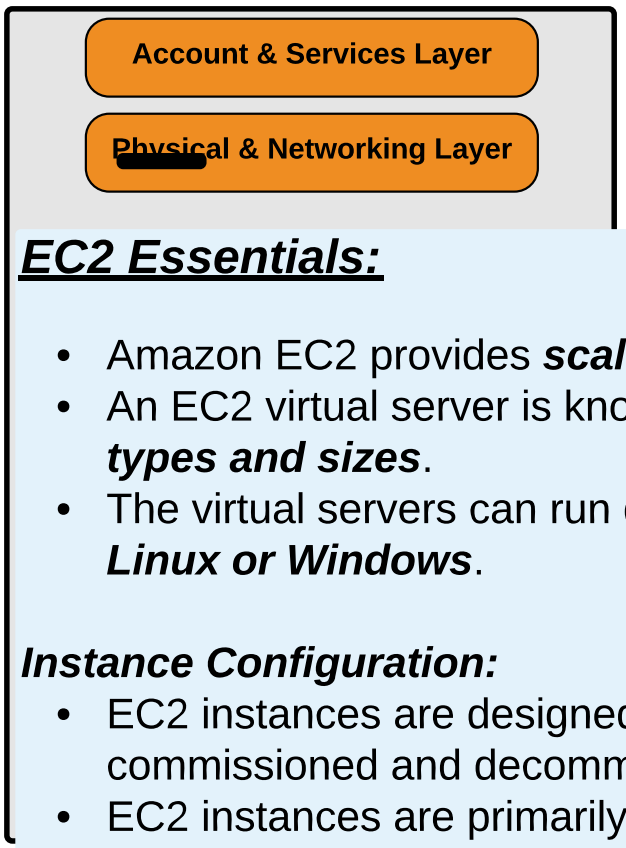
- EBS volumes are ***persistent***, meaning that they can live beyond the life of the EC2 instance they are attached to.
- EBS backed volumes are ***network attached storage***, meaning they can be attached/detached to or from various EC2 instances.
- However, they can only be attached to ONE EC2 instance at a time.
- EBS volumes have the benefit of being backed up into a ***snapshot*** - which can later be restored into a new EBS volume.

EBS Performance:

- EBS volumes measure input/output operations in IOPS:
  - IOPS are input/output operations per second
  - AWS measures IOPS in ***256KB chunks*** (or smaller)
  - Operations that are greater than 256KB are separated into individual 256KB chunks
  - For example, A 512KB operation would count as 2 IOPS
- The type of EBS volume you specify greatly influences the I/O performance (IOPS) your device will receive.
- It is important as architects to understand if your application requires more (or less) I/O when selecting an EBS volume.
- Even volumes with "***provisioned IOPS***" may not produce the performance you expect. If this is the case, an EBS optimized instance type is required, which prioritizes EBS traffic.

Pre-warming EBS Volumes:

- The first time data is written to an EBS volume, AWS runs a industry standard deletion process to overwrite any existing data that may be on the volume.
- Volume performance will be degraded during this process.
- As a solutions, you can "pre-warm" volumes by writing dummy data onto the volume before it is used in a production environment.



EC2 Essentials:

- Amazon EC2 provides ***scalable virtual servers*** in the cloud.
- An EC2 virtual server is known as an "***instance***" and can be made up of different ***instance types and sizes***.
- The virtual servers can run different operating systems, but most commonly run a flavor of ***Linux or Windows***.

Instance Configuration:

- EC2 instances are designed to mimic traditional on-premise servers, but with the ability to be commissioned and decommissioned on-demand for easy scalability and elasticity.
- EC2 instances are primarily comprised of the follow components:
  - ***Amazon Machine Image*** (AMI): The operating system (and other settings).
  - ***Instance Type***: The hardware (compute power, ram, network bandwidth, etc).
  - ***Network interface*** (public, private, or elastic IP addresses).
  - ***Storage:*** The instances "hard drive" (including two options).
    - Elastic Block Store (EBS) - which is "network persistent storage"
    - Instance Store - which is "ephemeral storage"

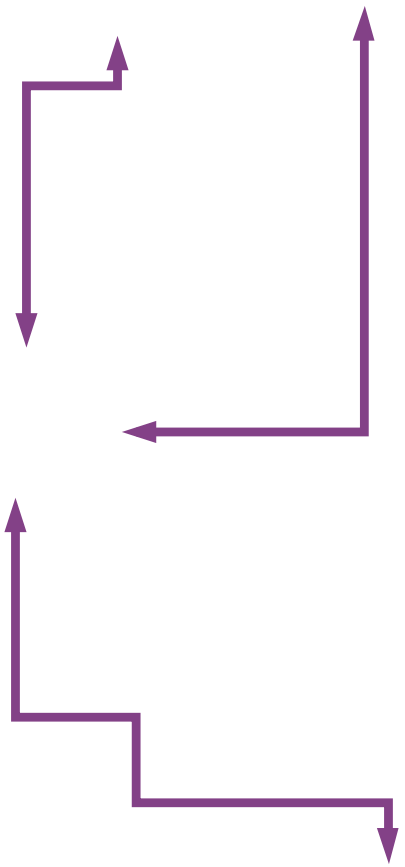
Other Important EC2 Facts:

- A security group must be assigned to an instance during the creation process.
- Each instance must be placed into an existing VPC, availability zone and subnet.
- Automated (bootstrapping) custom launch commands can be passed into the instance during launch via "user-data" scripts.
- "Tags" can be used to help name and organize provisioned instances.
- Encrypted key-pairs are used to manage login authentication.
- There are limits on the amount of instances you can have running in a region at any particular time.

Account & Services Layer

Physical & Networking Layer

Appendix



On-premises Data Center

On-Premises Servers

Hybrid Environments

Account & Services Layer

Physical & Networking Layer

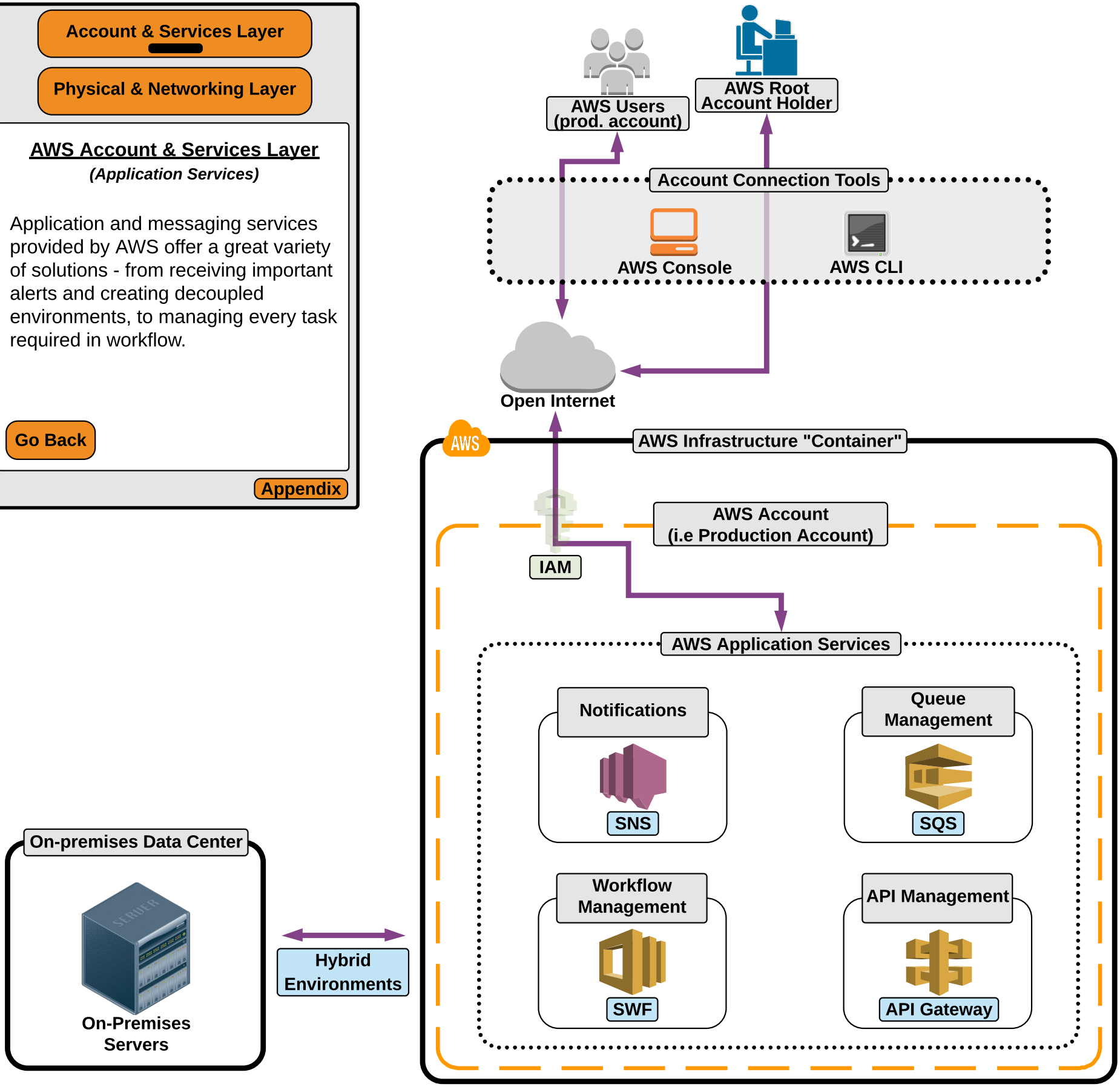
AWS Account & Services Layer

(Application Services)

Application and messaging services provided by AWS offer a great variety of solutions - from receiving important alerts and creating decoupled environments, to managing every task required in workflow.

Go Back

Appendix



Account & Services Layer

Physical & Networking Layer

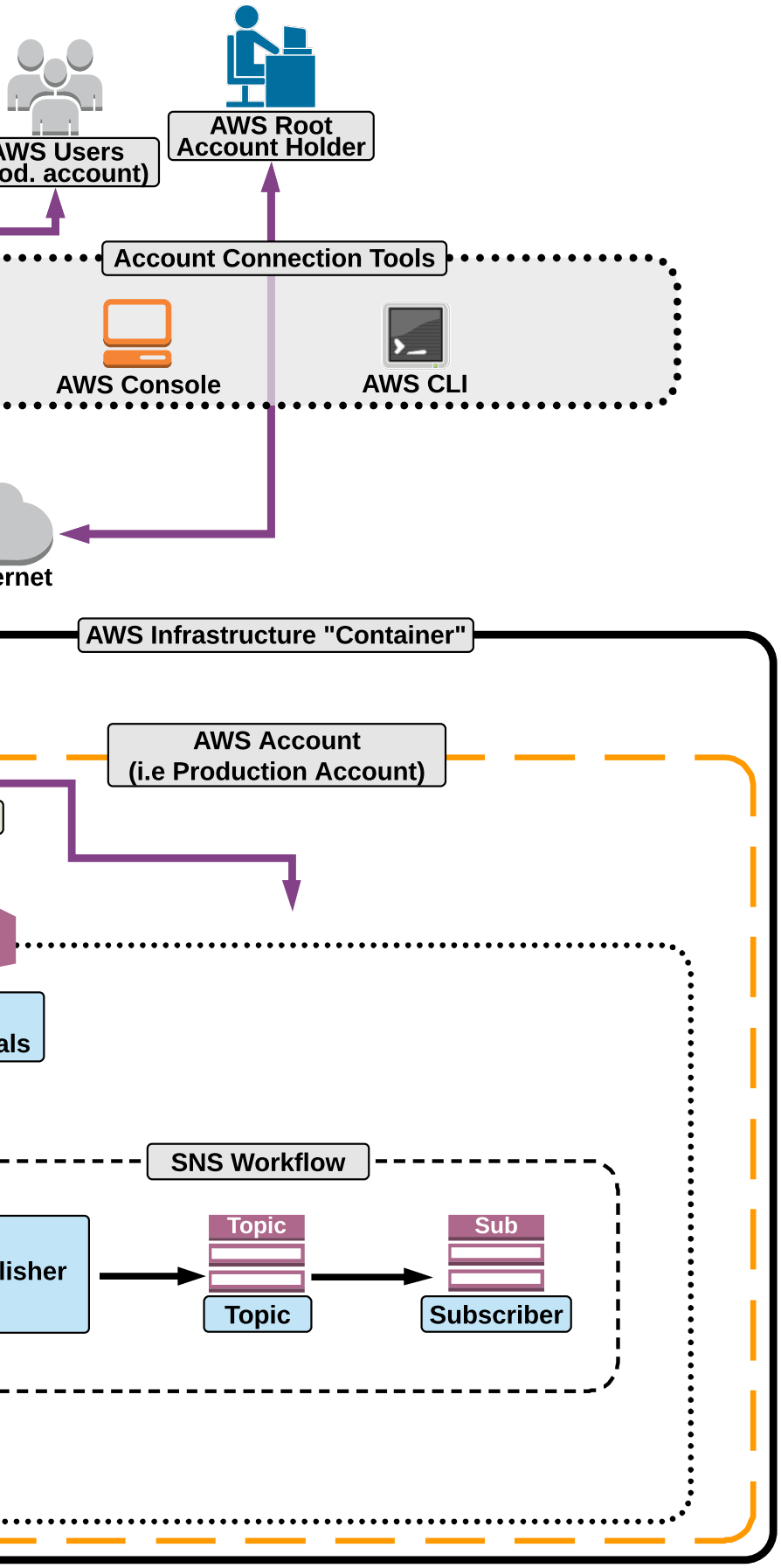
AWS Account & Services Layer

(Simple Notification Service)

Simple Notification Service (SNS) is an integrated notification service provided by AWS that allows for the sending of messages to various endpoints. Generally these messages are used for alert notifications to system admins, or to create automation.

Go Back

Appendix



Account & Services Layer

Physical & Networking Layer

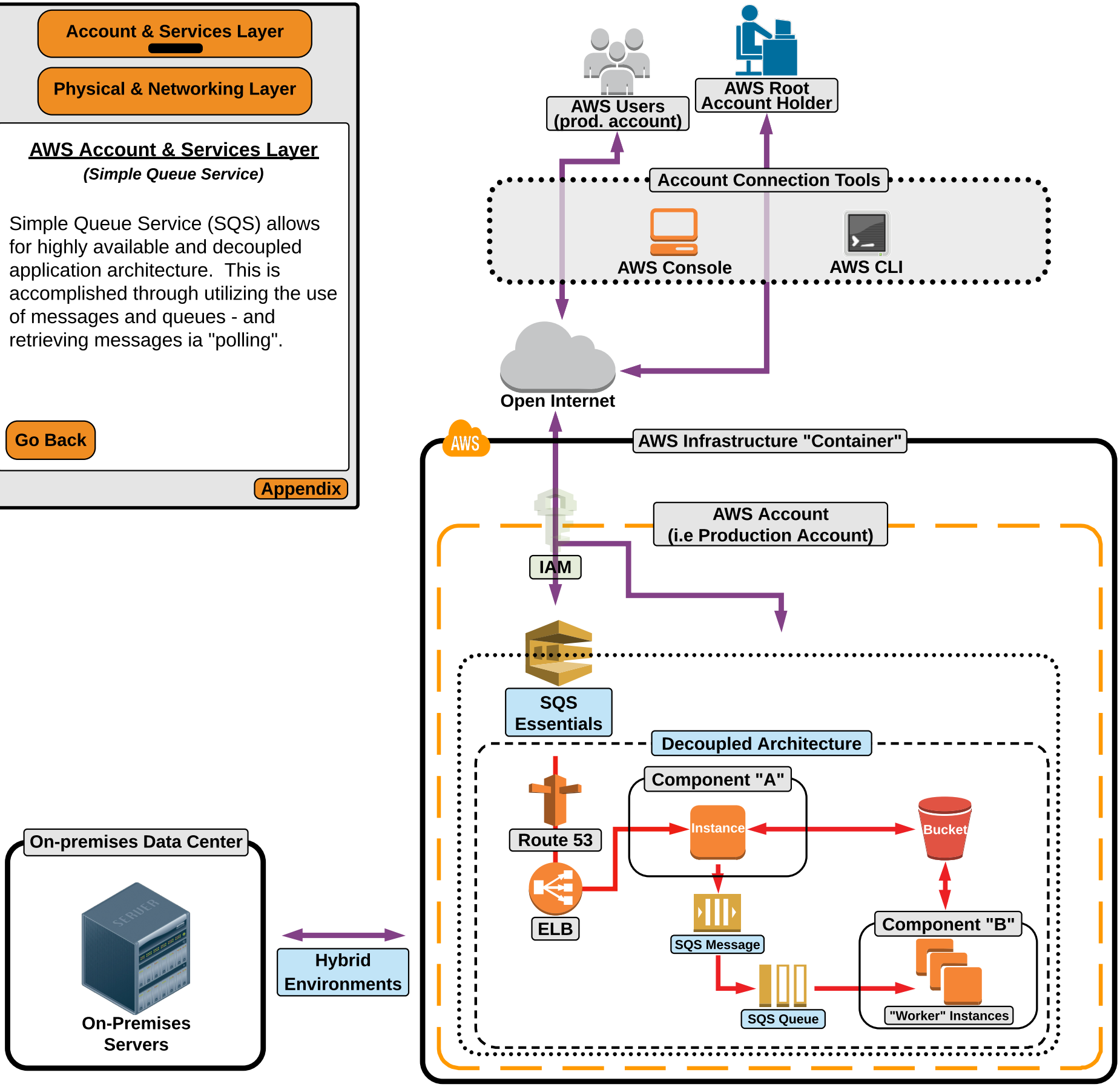
AWS Account & Services Layer

(Simple Queue Service)

Simple Queue Service (SQS) allows for highly available and decoupled application architecture. This is accomplished through utilizing the use of messages and queues - and retrieving messages ia "polling".

Go Back

Appendix



Account & Services Layer

Physical & Networking Layer

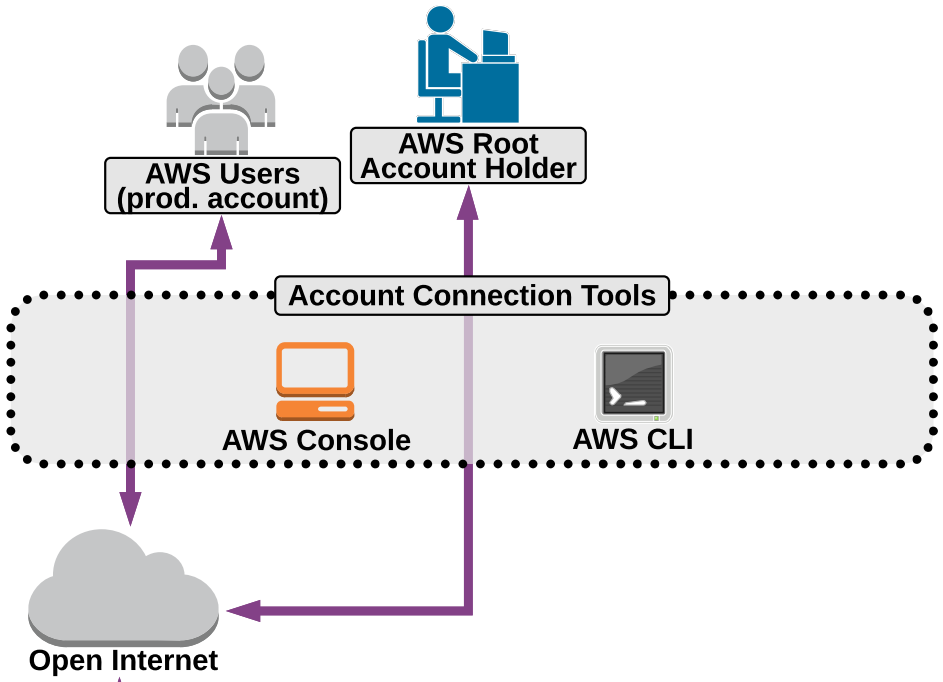
AWS Account & Services Layer

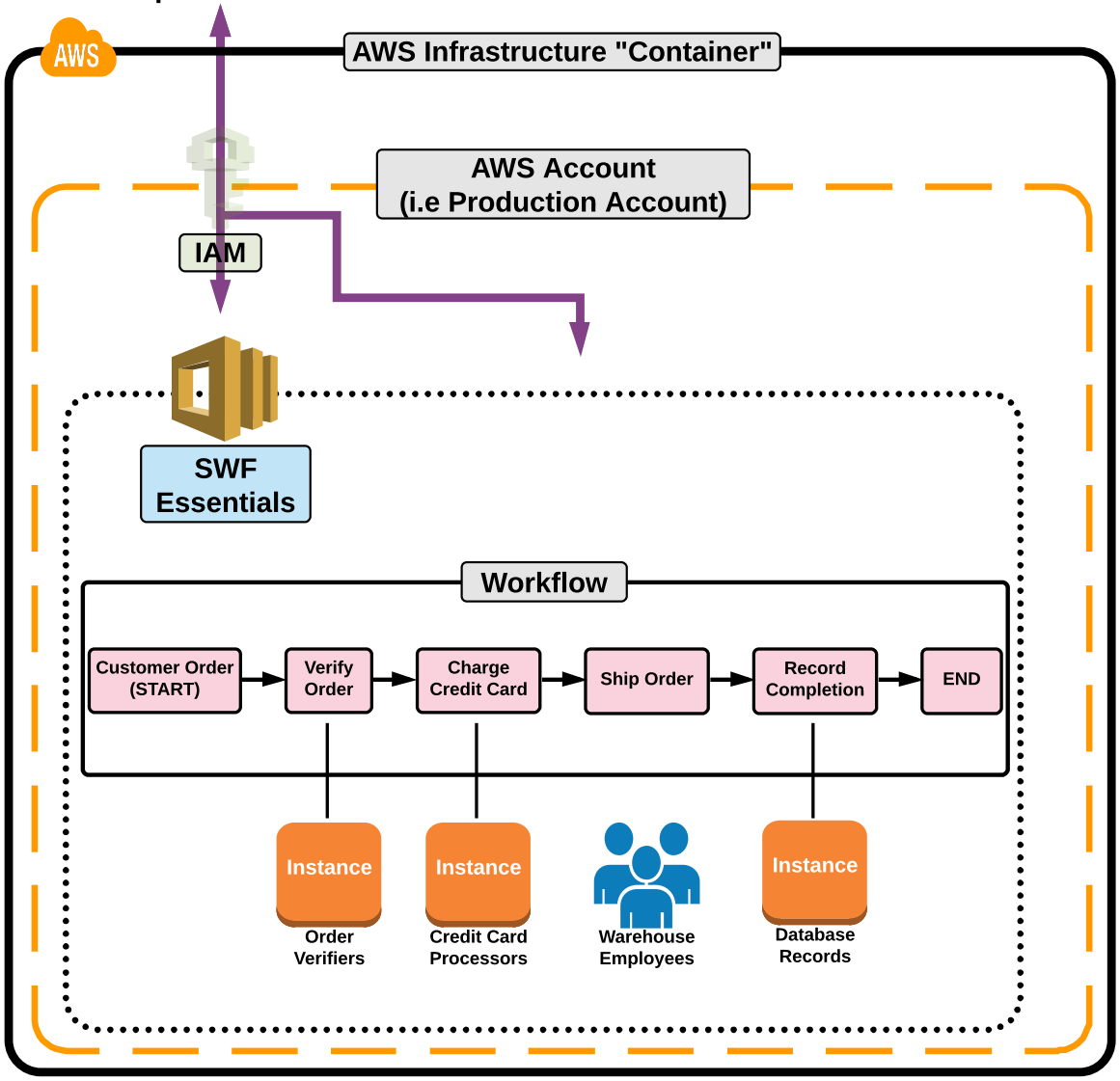
(Simple Workflow)

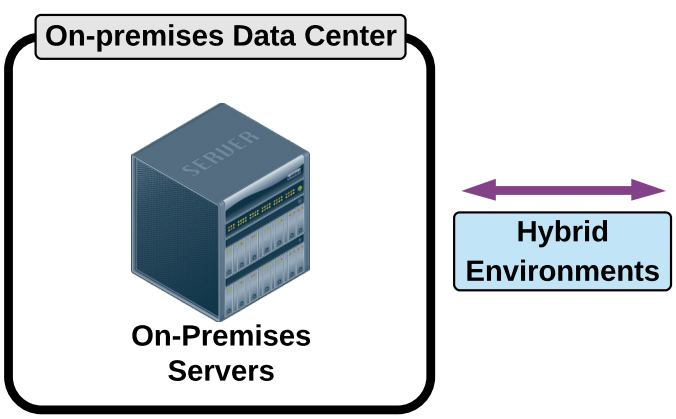
Simple Workflow (SWF) coordinates and manages the execution of activities. It manages a specific "job" from start to finish - while still allowing for distributed/decoupled architecture.

Go Back

Appendix







Account & Services Layer

Physical & Networking Layer

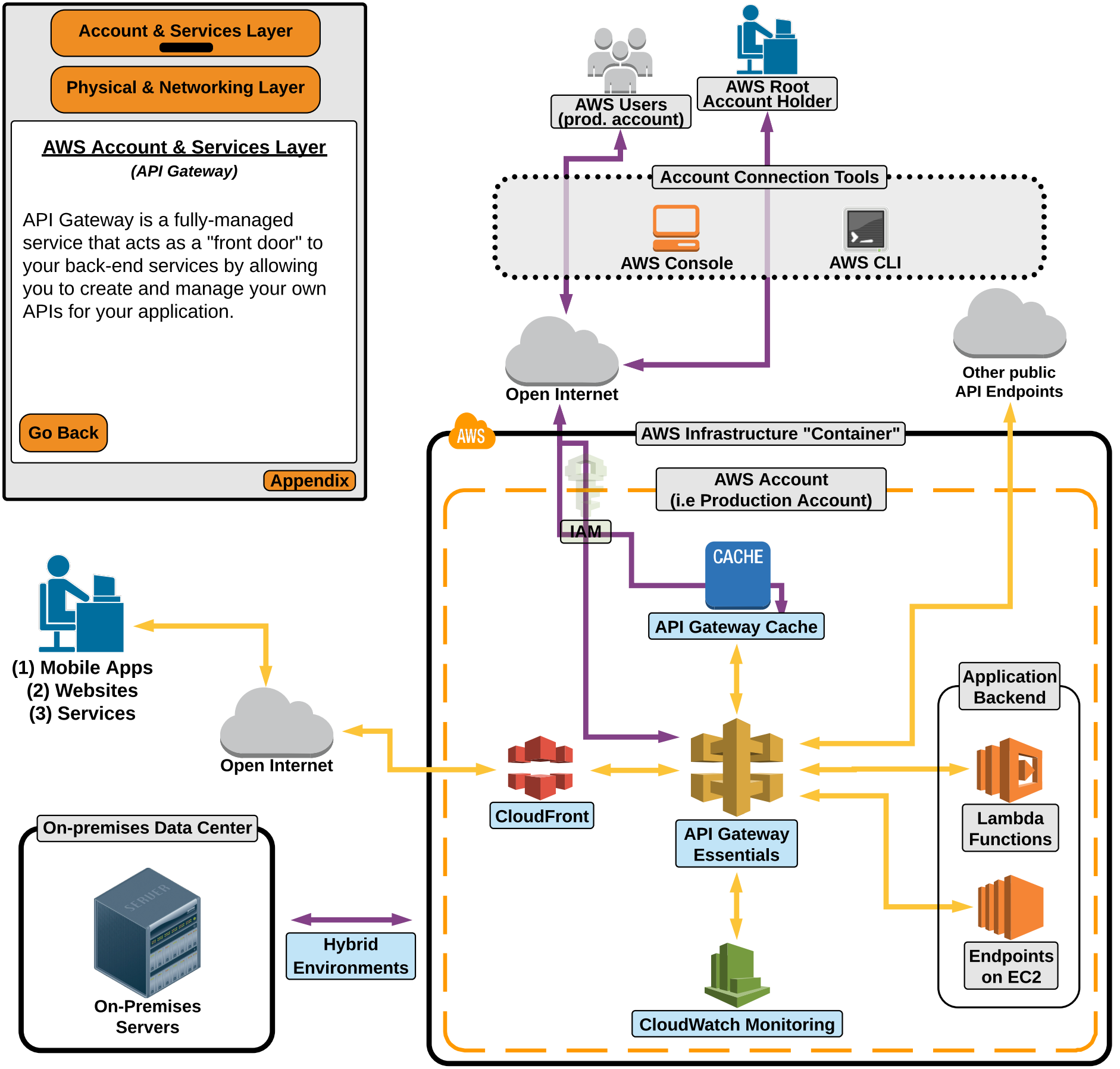
AWS Account & Services Layer

(API Gateway)

API Gateway is a fully-managed service that acts as a "front door" to your back-end services by allowing you to create and manage your own APIs for your application.

Go Back

Appendix



Application Services:

X

SNS Essentials:

- SNS coordinates and manages the sending and delivery of messages to specific endpoints.
- 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 very easy to setup 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 publishing IOS/Android app notifications, and creating automation based off of notifications.

SNS Components:

- Topic:
  - The group of ***subscriptions*** that you send a message to.
- Subscription:
  - An endpoint that a message is sent.
  - 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:
    - Human
    - S3 Event
    - Cloudwatch Alarm

Application Services:

X

SQS Essentials:

- SQS provides the ability to have hosted/highly available queues that can be be used for messages being sent between servers.
- This allows for the creation of ***distributed/decoupled*** application components.
- ***SQS is used to create decoupled application environments***.
- Messages between servers are retrieved through ***polling***.

Two types of polling

- Long Polling (1-20 seconds):
  - Allows the SQS service to wait until a message is available in a queue before sending a response, and will return all messages from all SQS services.
  - Long polling reduces API requests (over using short polling).
- Short Polling:
  - SQS samples a subset of servers and returns messages from just those servers.
  - Will not return all possible messages in a poll.
  - Increases API requests (over long polling), which increases costs.

Other important SQS facts

- Each message can contain up to 256KB of text (in any format).
- Amazon SQS guarantees delivery of each message at least once BUT DOES NOT guarantee the order (best effort) in which they are delivered to the queue.
- It does not guarantee first-in-first-out order.
- SQS is also highly available and redundant.

SQS Workflow

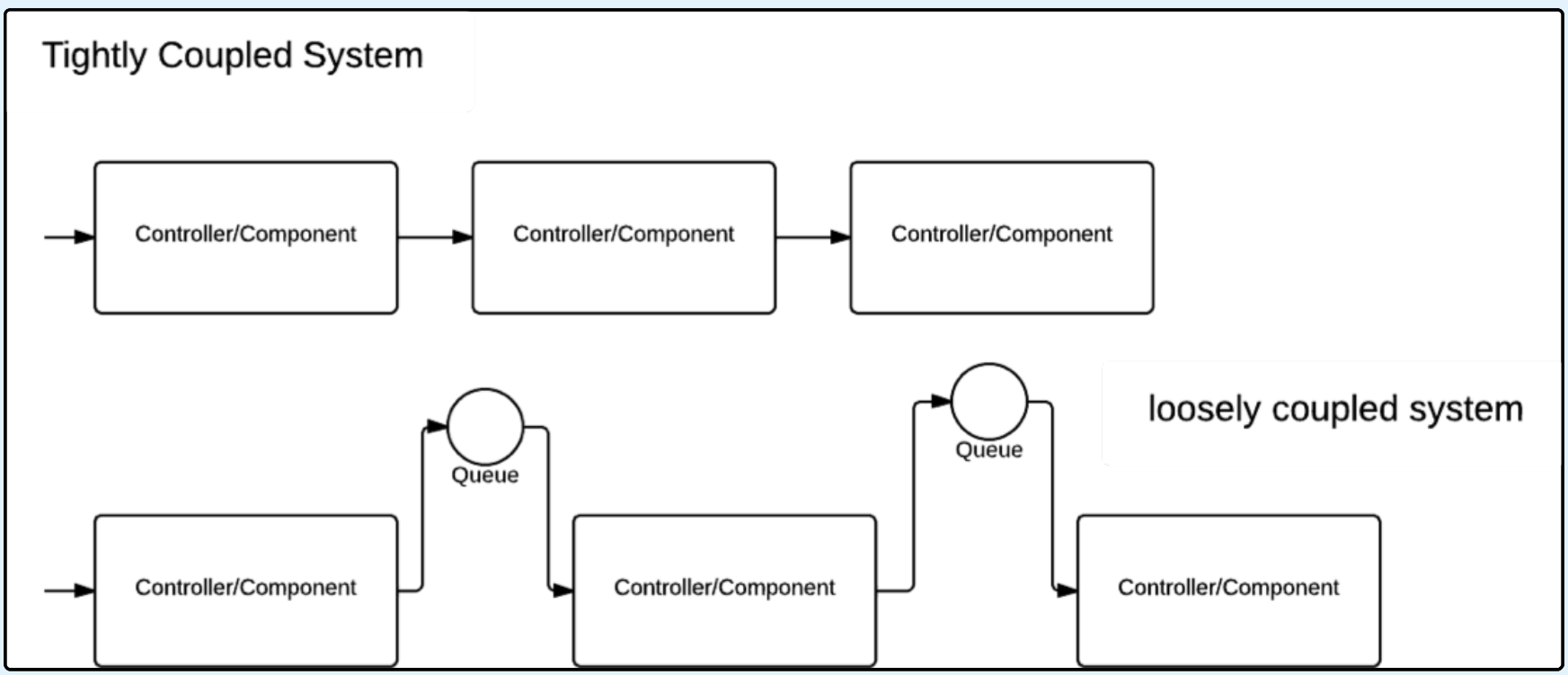
- Generally a “worker” instance will “poll” a queue to retrieve waiting messages for processing.
- Auto Scaling can be applied based off of queue size so that if a component of your application has an increase in demand, the number of worker instances can increase.

Application Services:

X

Decoupled Architecture:

- Tightly Coupled System:
  - A system architecture of components that are not just linked together but are also dependent upon each other.
  - If one component fails all components fail.
- Loosely Coupled/Decoupled Systems:
  - Multiple components that can process information without being connected.
  - Components are not connected - if one fails the rest of the system can continue processing (fault tolerant/highly available).
- AWS Services that are used for distributed/decoupled system architectures:
  - SWF (Simple Work Flow Service)
  - SQS (Simple Queue Service)



Application Services:

X

Simple Workflow Essentials:

- SWF is a fully-managed “work flow” service provided by AWS.
- A SWF ***workflow*** allows an architect/developer to implement distributed, asynchronous applications as a work flow.
- 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 which an application can integrate it's work flow service into. This allows the service to be used by non-AWS services, such as an on-premise data center.
- A workflow execution can last up to 1 year.

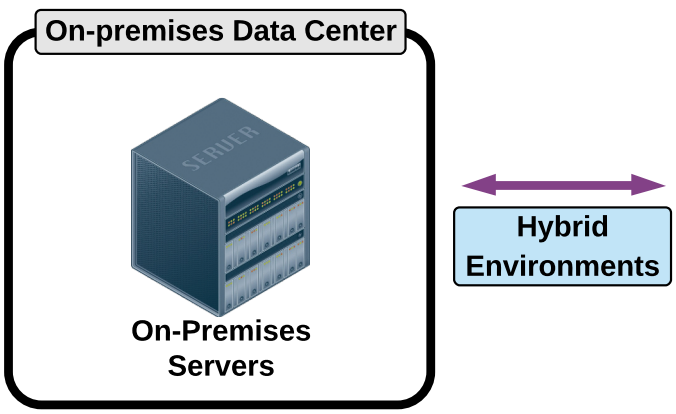
Components of SWF

- *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 work flow 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.

Application Services:

SNS Publisher:

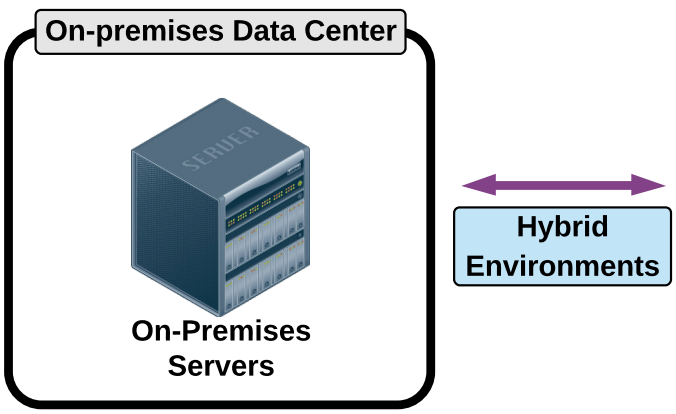
- The "entity" that triggers the sending of a message
- Examples include:
  - Human
  - S3 Event
  - Cloudwatch Alarm



Application Services:

SNS Topic:

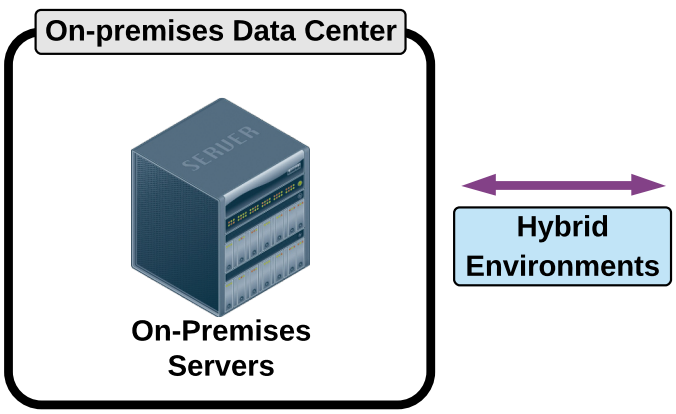
- The group of ***subscriptions*** that you send a message to.



Application Services:

SNS Subscriber:

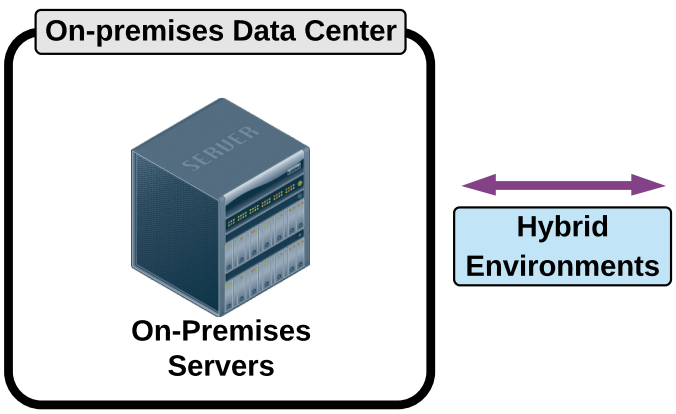
- An endpoint that a message is sent to.
- Available endpoints include:
  - HTTP
  - HTTPS
  - Email
  - Email-JSON
  - SQS
  - Application, Mobile APP notifications (IOS/Android/Amazon/Microsoft)
  - Lambda
  - SMS (cellular text message)



Application Services:

SQS Message:

- A set of instructions that will be relayed to the "worker" instances via the SNS Queue.
- Can be up to 256KB of text (in any format).
- Each message is guaranteed to be delivered at least once:
  - Order is not guaranteed
  - Duplicates can occur

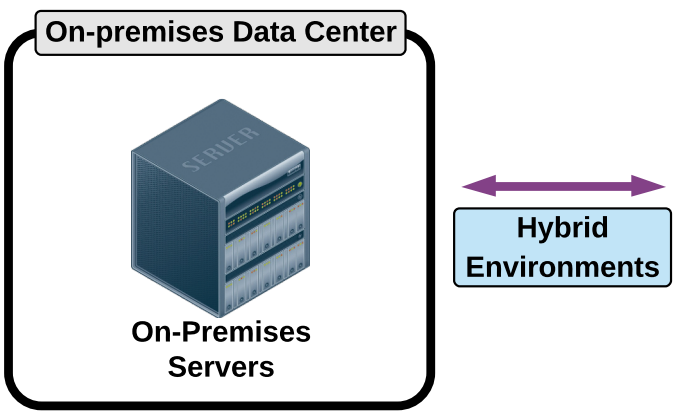


Application Services:

X

SQS Queue:

- An queue stores messages (for up to 14 days), that can be retrieved through "polling".
- Queues allows components of your application to work independently of each other (decoupled environments).



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 you application, allowing access to data/logic/functionality from your back-end services.

API Gateway Main Features:

- Build RESTful APIs with:
  - Resources
  - Methods (i.e. GET, POST, PUT)
  - Settings
- Deploy APIs to a "Stage" (different envs: i.e. dev, beta, production)
  - Each stage can have it's 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 request per second (for each HTTP method)
  - Request over the limit throttled (HTTP 429 response)
- *Security using Signature v.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 (i.e. JSON IN to XML OUT)

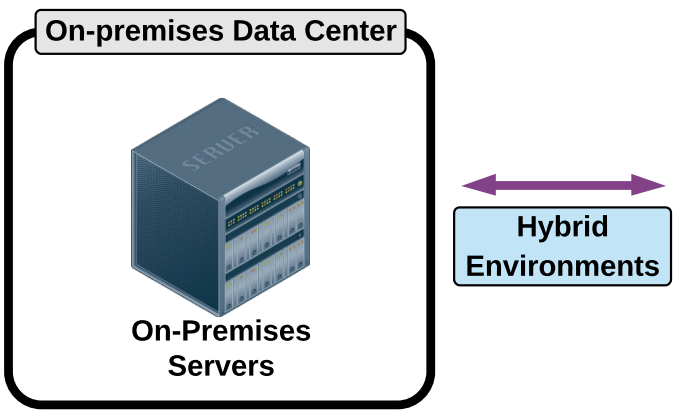
Application Services:

X

API Gateway Caching:

- API Gateway will cache API responses so that duplicate API request do not have to hit your back-end.
  - This reduces load on your back-end AND
  - Speeds up calls to your back-end
- You can configure a cache key and Time to Live (TTL) of the API response.
- Caching can be setup on a per API or per stage basis.

Appendix



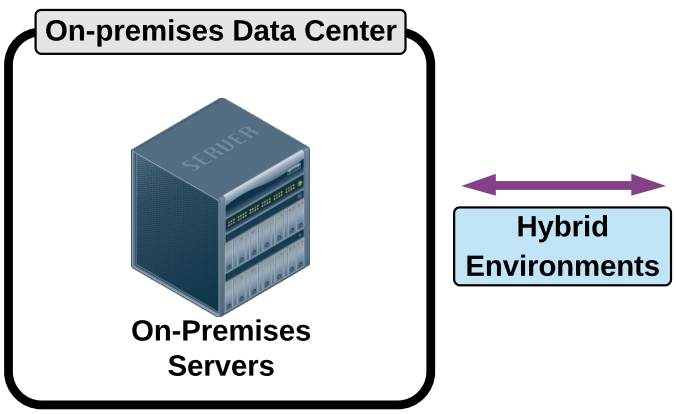
Application Services:

X

API Gateway: CloudFront

- API Gateway benefits from using CloudFront infrastructure:
  - Built in Distributed Denial of Service (DDoS) attack protection and mitigation.
  - All CloudFront Edge Locations become entry points for your API into your back-end.
- Summary: Benefits are reduced latency and improved projection.

Appendix



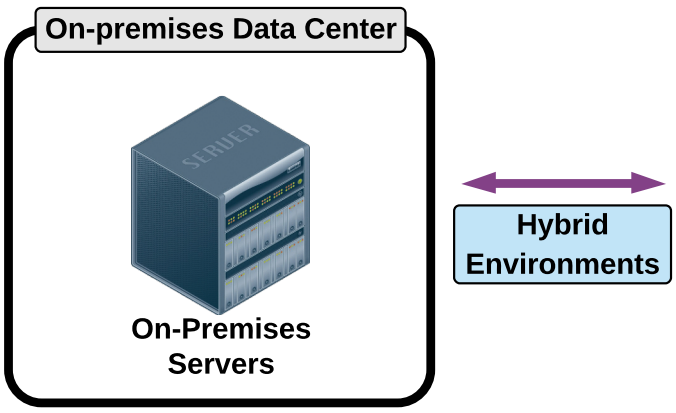
Application Services:

X

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.

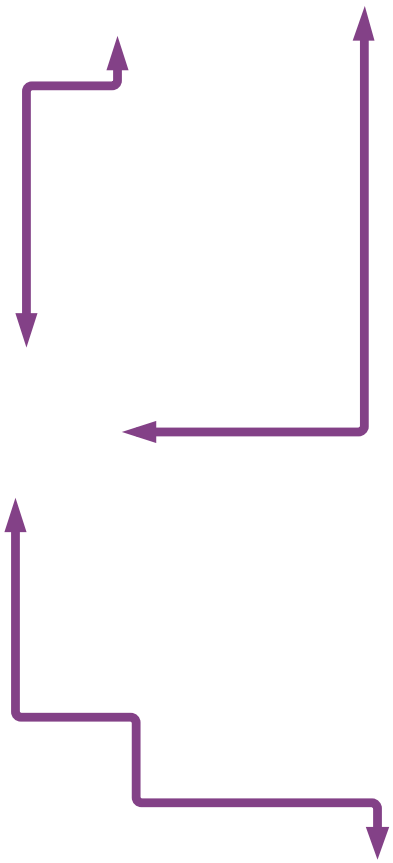
Appendix



Account & Services Layer

Physical & Networking Layer

Appendix



On-premises Data Center

On-Premises Servers

Hybrid Environments

Account & Services Layer

Physical & Networking Layer

AWS Account & Services Layer

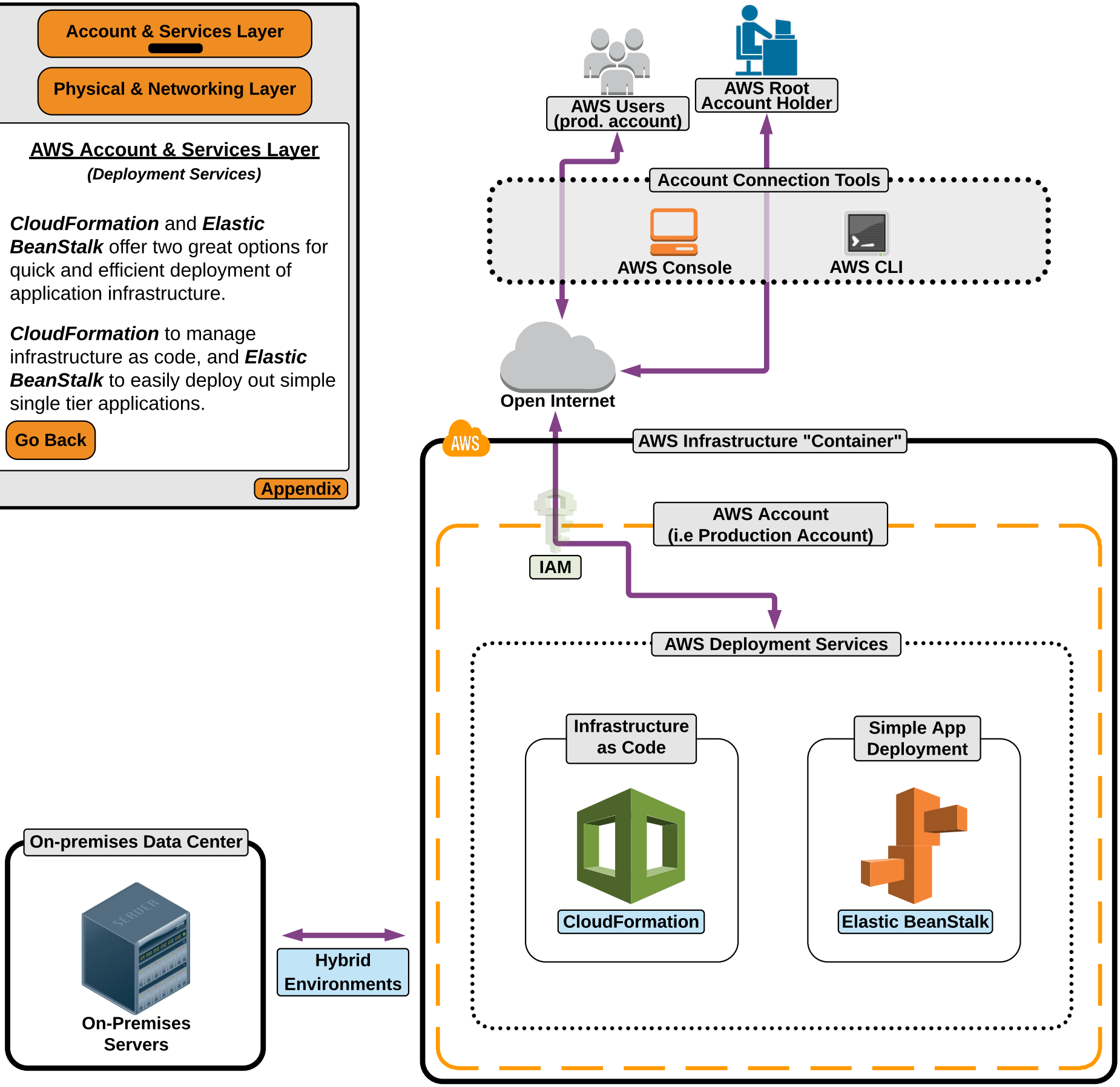
(Deployment Services)

CloudFormation and ***Elastic BeanStalk*** offer two great options for quick and efficient deployment of application infrastructure.

CloudFormation to manage infrastructure as code, and ***Elastic BeanStalk*** to easily deploy out simple single tier applications.

Go Back

Appendix



Account & Services Layer

Physical & Networking Layer

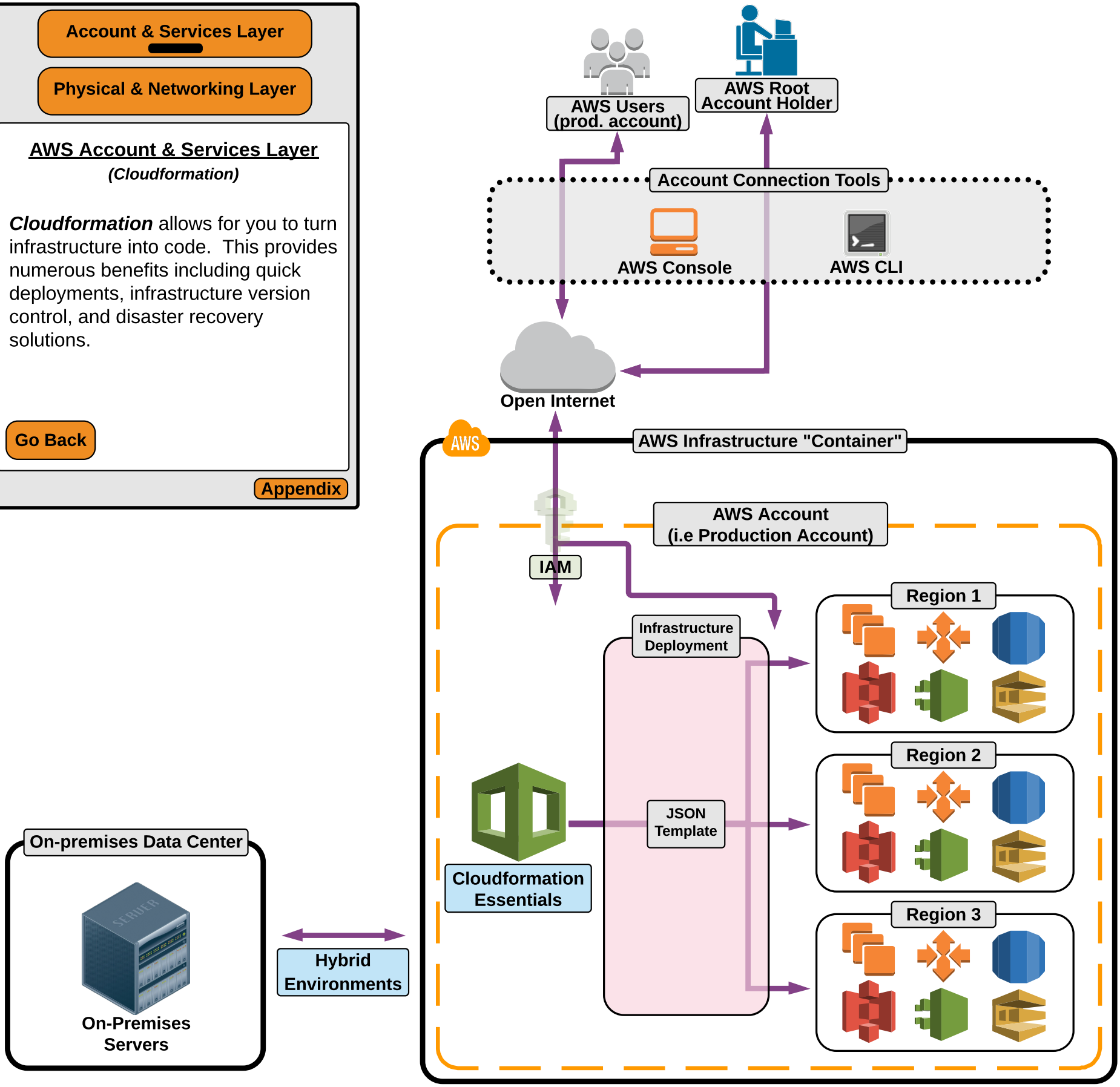
AWS Account & Services Layer

(Cloudformation)

Cloudformation allows for you to turn infrastructure into code. This provides numerous benefits including quick deployments, infrastructure version control, and disaster recovery solutions.

Go Back

Appendix



Account & Services Layer

Physical & Networking Layer

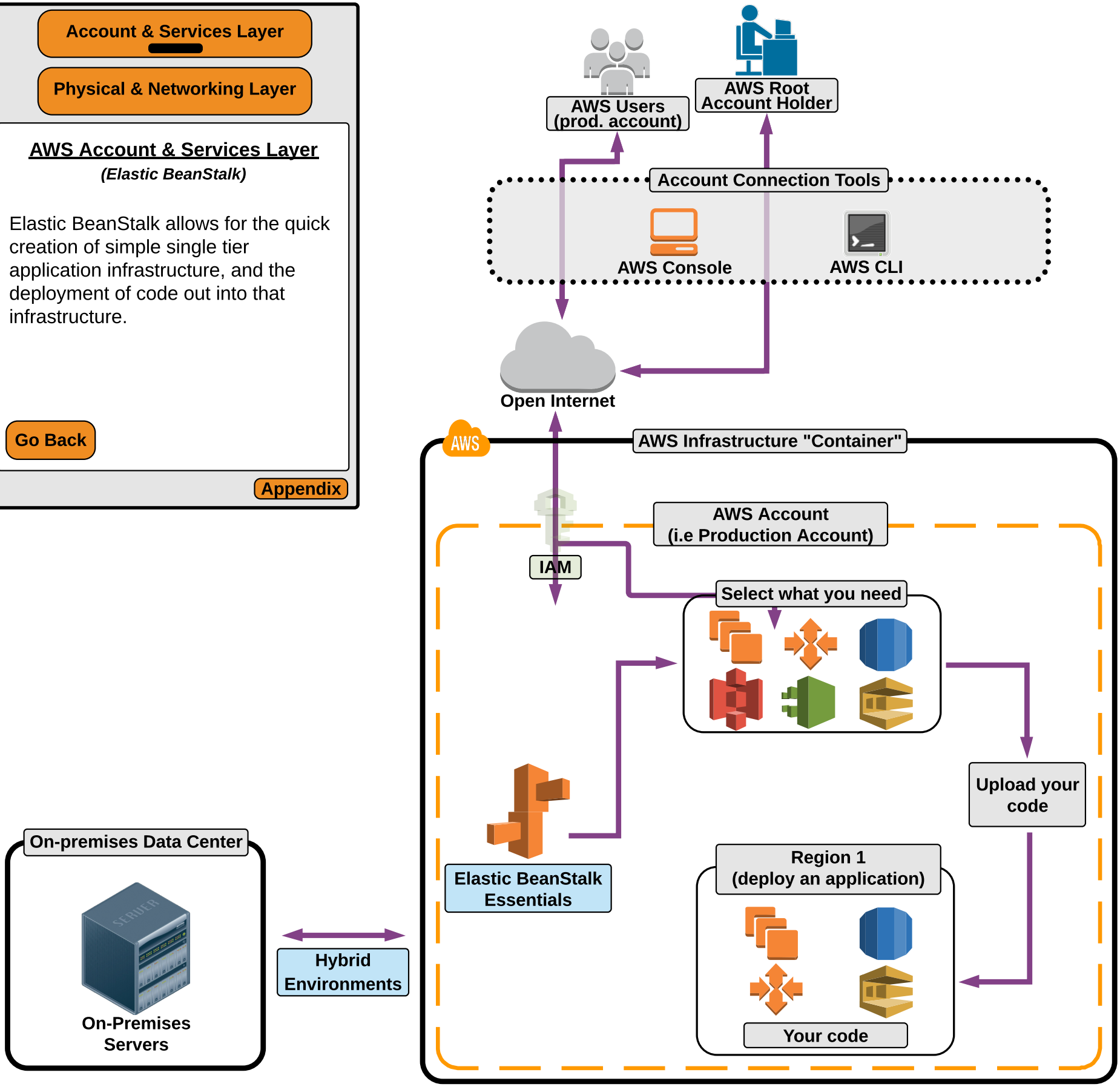
AWS Account & Services Layer

(Elastic BeanStalk)

Elastic BeanStalk allows for the quick creation of simple single tier application infrastructure, and the deployment of code out into that infrastructure.

Go Back

Appendix



Deployment Services:

Cloudformation Essentials:

- CloudFormation is the pure definition of infrastructure as code:
  - You can "convert" your application's architecture into a JSON formatted template (so your architecture is literally code).
  - You can then use that JSON template to deploy out updated or replicated copies of that architecture to multiple regions.

Benefits

- Saves time - you don't have to manually create duplicate architecture in additional regions.
- Since your infrastructure is now code, you can version control your infrastructure. Allowing for rollbacks to previous versions of your infrastructure if a new version has issues.
- Allows for backups of your infrastructure.
- Great solution for disaster recovery.

Deployment Services:

Elastic BeanStalk Essentials:

- Elastic Beanstalk is designed to make it easy to deploy less complex applications.
- This helps reduce the management required for building and deploying applications.
- Elastic Beanstalk is used to deploy out easy, single-tier applications that take advantage of core services such as:
  - EC2
  - Auto Scaling
  - ELB
  - RDS
  - SQS
  - CloudFront

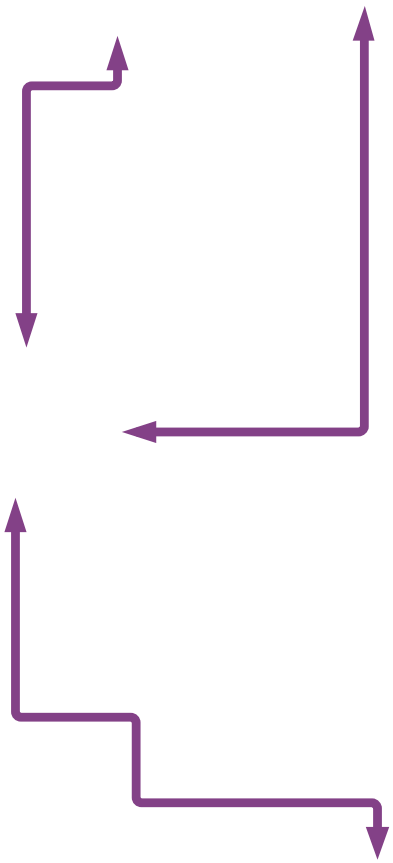
- Why/when to use Elastic Beanstalk:
  - In order to quickly provision an AWS environment that require little to no management.
  - The application fits within the parameters of the Beanstalk service.
  - Can deploy from repositories or from uploaded code files.
  - Easily update applications by uploading new code files or requesting a pull from a repository.

- Supported Platforms:
  - Docker
  - Java
  - Windows .NET
  - Node.js
  - PHP
  - Python
  - Ruby

Account & Services Layer

Physical & Networking Layer

Appendix



On-premises Data Center

On-Premises Servers

Hybrid Environments

Account & Services Layer

Physical & Networking Layer

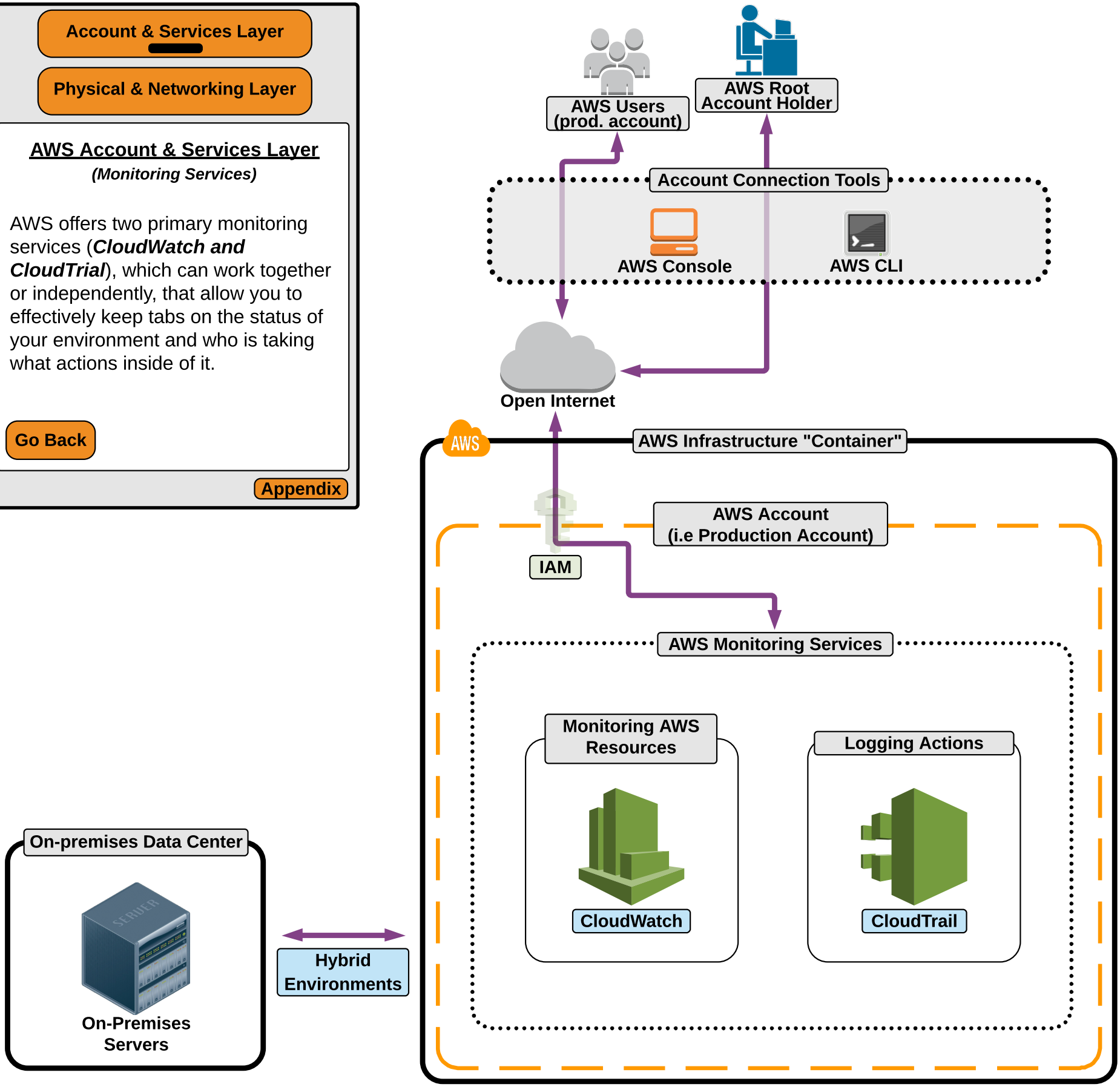
AWS Account & Services Layer

(Monitoring Services)

AWS offers two primary monitoring services (***CloudWatch and CloudTrial***), which can work together or independently, that allow you to effectively keep tabs on the status of your environment and who is taking what actions inside of it.

Go Back

Appendix



Account & Services Layer

Physical & Networking Layer

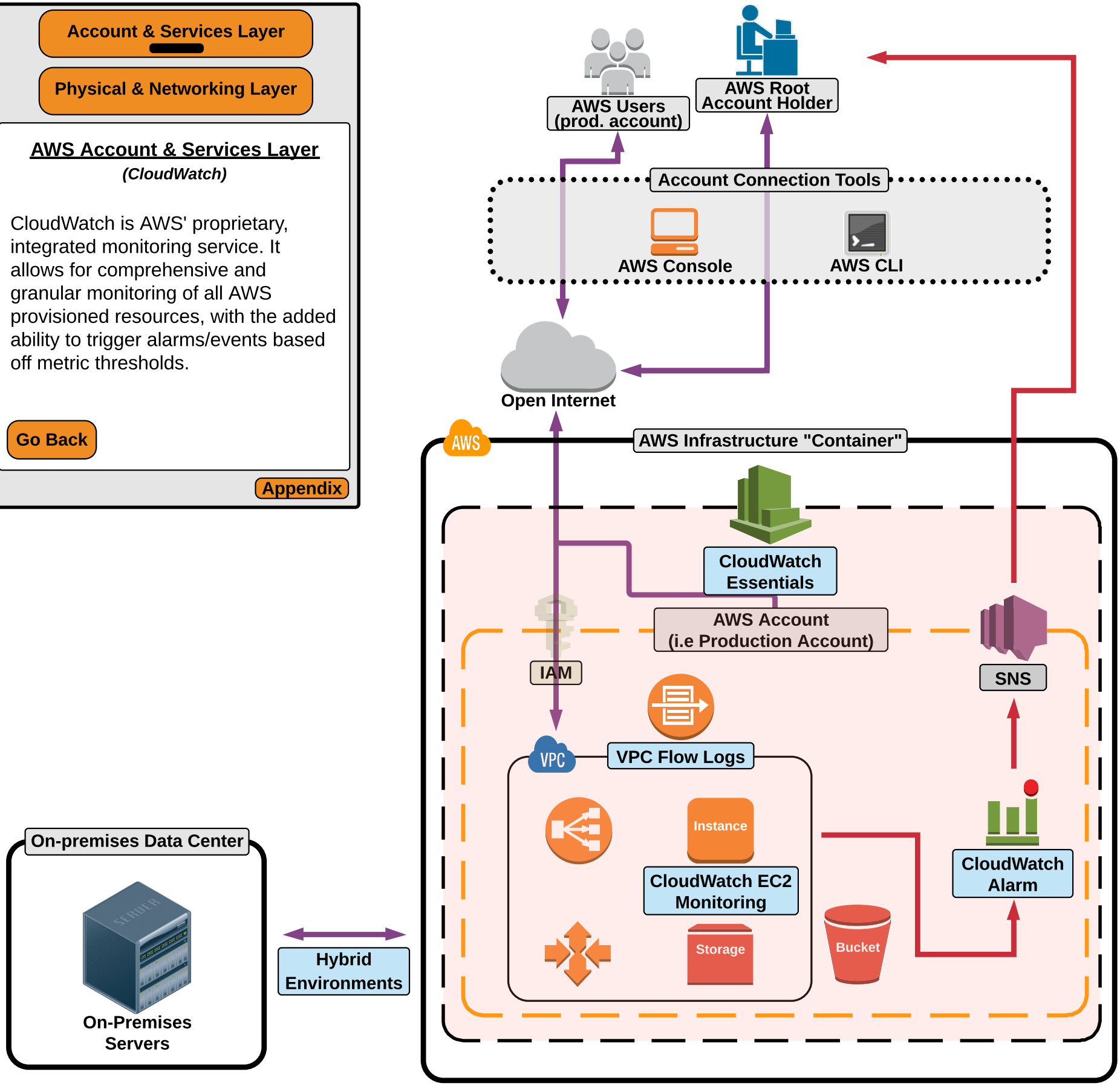
AWS Account & Services Layer

(CloudWatch)

CloudWatch is AWS' proprietary, integrated monitoring service. It allows for comprehensive and granular monitoring of all AWS provisioned resources, with the added ability to trigger alarms/events based off metric thresholds.

Go Back

Appendix



Account & Services Layer

Physical & Networking Layer

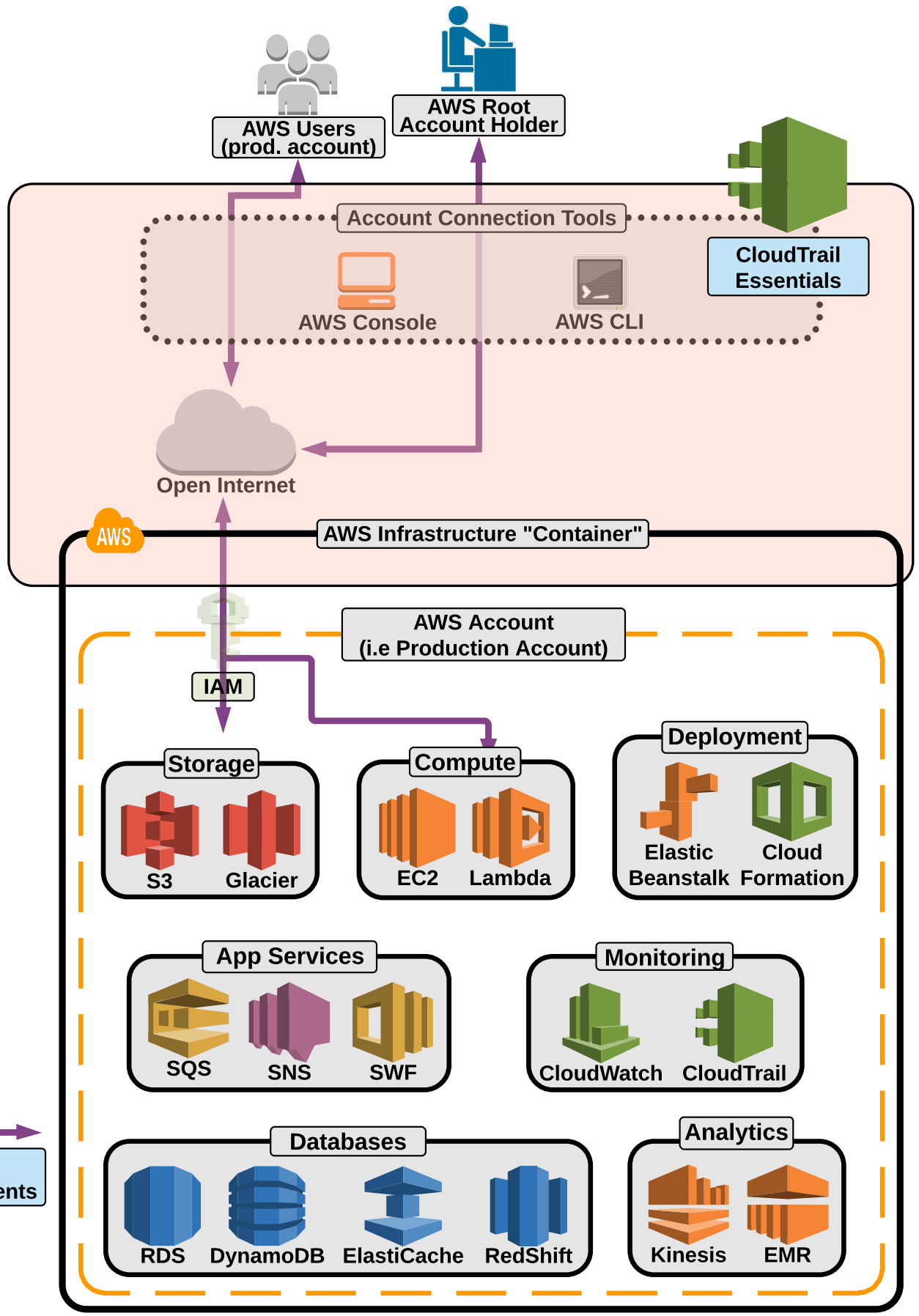
AWS Account & Services Layer

(CloudTrail)

AWS is just one big API that we (as users) make calls against any time we do something in the console, command line, or SDK. ***CloudTrail*** provides centralized logging so that we can log each action taken in our environment and store it for later user (if needed).

Go Back

Appendix



Monitoring Services:

CloudWatch Essentials:

- CloudWatch is used to monitor AWS services, such as EC2, ELB and S3.
- You monitor you environment by configuring and viewing CloudWatch ***metrics***.
- Metrics are specific to each AWS service or resource, and include such metrics as:
  - EC2 per-instance metrics:
    - *CPUUtilization*
    - *CPUCreditUsage*
  - S3 Metrics:
    - *NumberOfObjects*
    - *BucketSizeBytes*
  - ELB Metrics:
    - *RequestCount*
    - *UnhealthyHostCount*

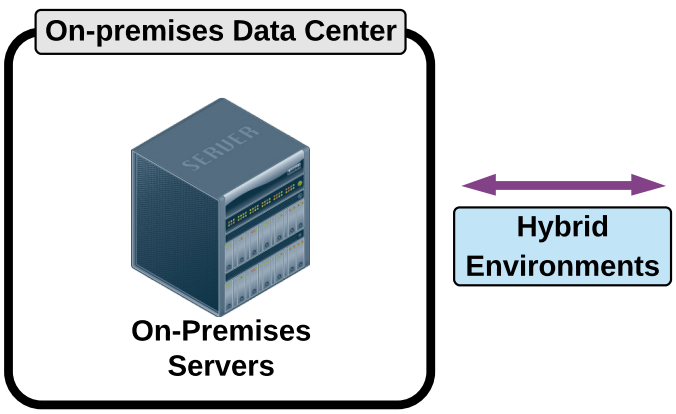
- Detailed vs. Basic level monitoring:
  - *Basic*: Data is available automatically in 5-minute periods at no charge
  - *Detailed*: Data is available in 1-minute periods

- CloudWatch Alarms can be created to trigger alerts (or other actions in your AWS accounts, such as an SNS topic), based on threshold you set on CloudWatch metrics.
- Auto Scaling heavily utilizes CloudWatch - relying on threshold and alarms to trigger the addition (or removal) of instances from an auto scaling group.

Monitoring Services:

CloudTrail Essentials:

- CloudTrail is an API logging service that logs ***all*** API calls made to AWS.
- It does not matter if the API calls from the command line, SDK, or console.
- All created logs are placed into a designated S3 bucket - so they are highly available by default.
- Cloudtrail logs help when addressing security concerns, by allowing you to view what actions users on your AWS account have performed.
- Since AWS is just one big API - CloudTrail can log ever single action taken in your account.



Monitoring Services:

CloudWatch EC2 Monitoring:

- ***System Status Checks:*** (things that are outside of our control)
  - Loss of network connectivity
  - Loss of system power
  - Software issues on the physical host
  - Hardware issues on the physical host
  - ***How to solve***: Generally stopping and restarting the instance will fix the issue. This causes the instance to launch on a different physical hardware device.

- ***Instance Status Checks:*** (software issues that we do control)
  - Failed system status checks
  - Misconfigured networking or startup configuration
  - Exhausted memory
  - Corrupted file system
  - Incompatible kernel
  - ***How to solve***: Generally a reboot, or solving the file system configuration issue.

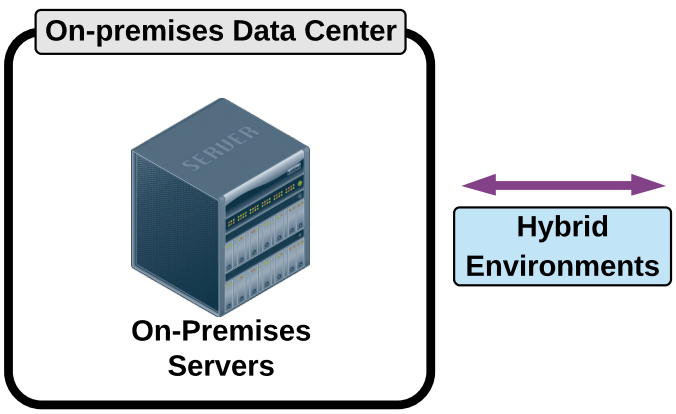
- By default, CloudWatch will automatically monitor metrics that can be viewed at the host level (NOT the software level), such as:
  - CPUUtilization
  - Network in/out
  - CPUCreditBalance
  - CPUCreditUsage

- OS level metrics that required a third party script (perl) to be installed (provided by AWS)
  - *Memory utilization, memory used, and memory available*
  - *Disk Swap utilization*
  - *Disk space utilization, disk space used, disk space available*

Monitoring Services:

CloudWatch Alarms:

- CloudWatch Alarms allow for you (or the system admin) to be notified when certain defined thresholds are met on CloudWatch Metrics.
- For example, you can setup an alarm to be triggered whenever the CPUUtilization metric on an EC2 instance goes above 70%.
- Alarms can also be used to trigger other events in AWS like publishing to an SNS topic or triggering auto scaling.



Monitoring Services:

X

VPC Flow Logs:

- VPC Flow Logs allow you to collect information about the IP traffic going to and from network interfaces in your VPC.
- VPC Flow Log data is stored in a log group in CloudWatch.
- Flow logs can be created on a specific VPC, Subnet or Network Interface.
- Flow logs created on a VPC or Subnet will include all network interfaces in that VPC or subnet.
- Each network interface will have its own unique log stream.
- You can set the log to capture data on accepted traffic, rejected traffic, or all traffic.
- Flow logs are NOT captured in “real-time”. The capture window is approx. 10 minutes, then data is published.
- VPC Flow Logs consist of network traffic for a specific ***5-tuple***.
- A 5-tuple is a set of five different values that comprise a TCP/IP connection. It includes:
  - (1) Source IP address and (2) source port number
  - (3) Destination IP address and (4) destination port number
  - (5) Protocol

Benefits of VPC Flow Logs:

- Troubleshoot why certain traffic is not reaching an EC2 instance.
- An added security layer by allowing you to monitor the traffic that reaches your EC2 instances.

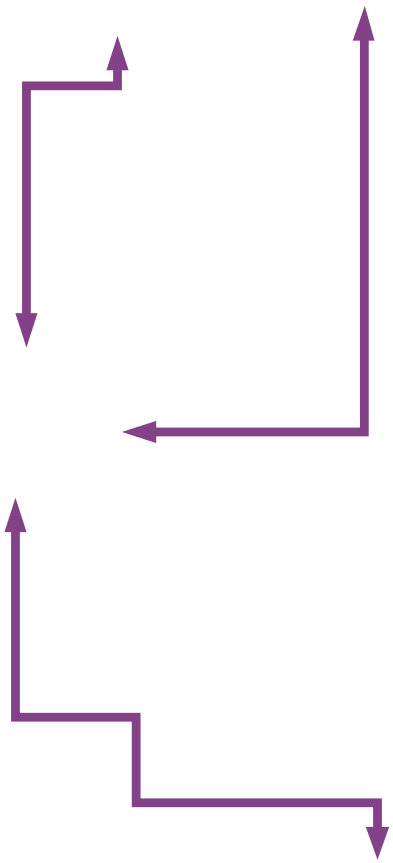
Limitations of VPC Flow Logs:

- Traffics NOT captured by VPC Flow Logs:
  - Traffic between an EC2 instance and an Amazon DNS Server
  - Traffic generated by request for instance metadata (request to 169.254.169.254)
  - DHCP Traffic

Account & Services Layer

Physical & Networking Layer

Appendix



On-premises Data Center

On-Premises Servers

Hybrid Environments

Account & Services Layer

Physical & Networking Layer

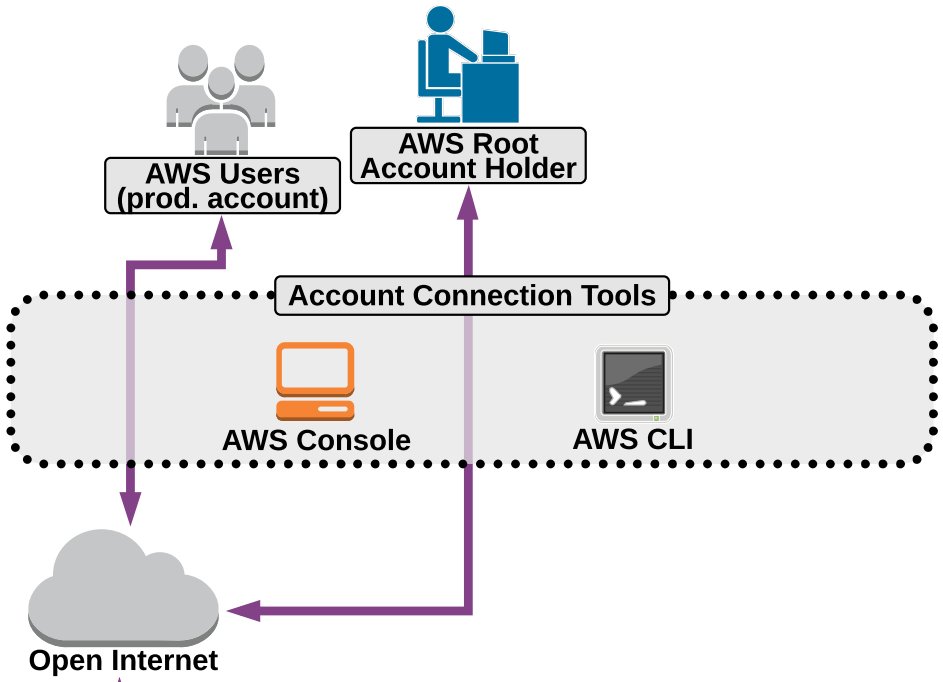
AWS Account & Services Layer

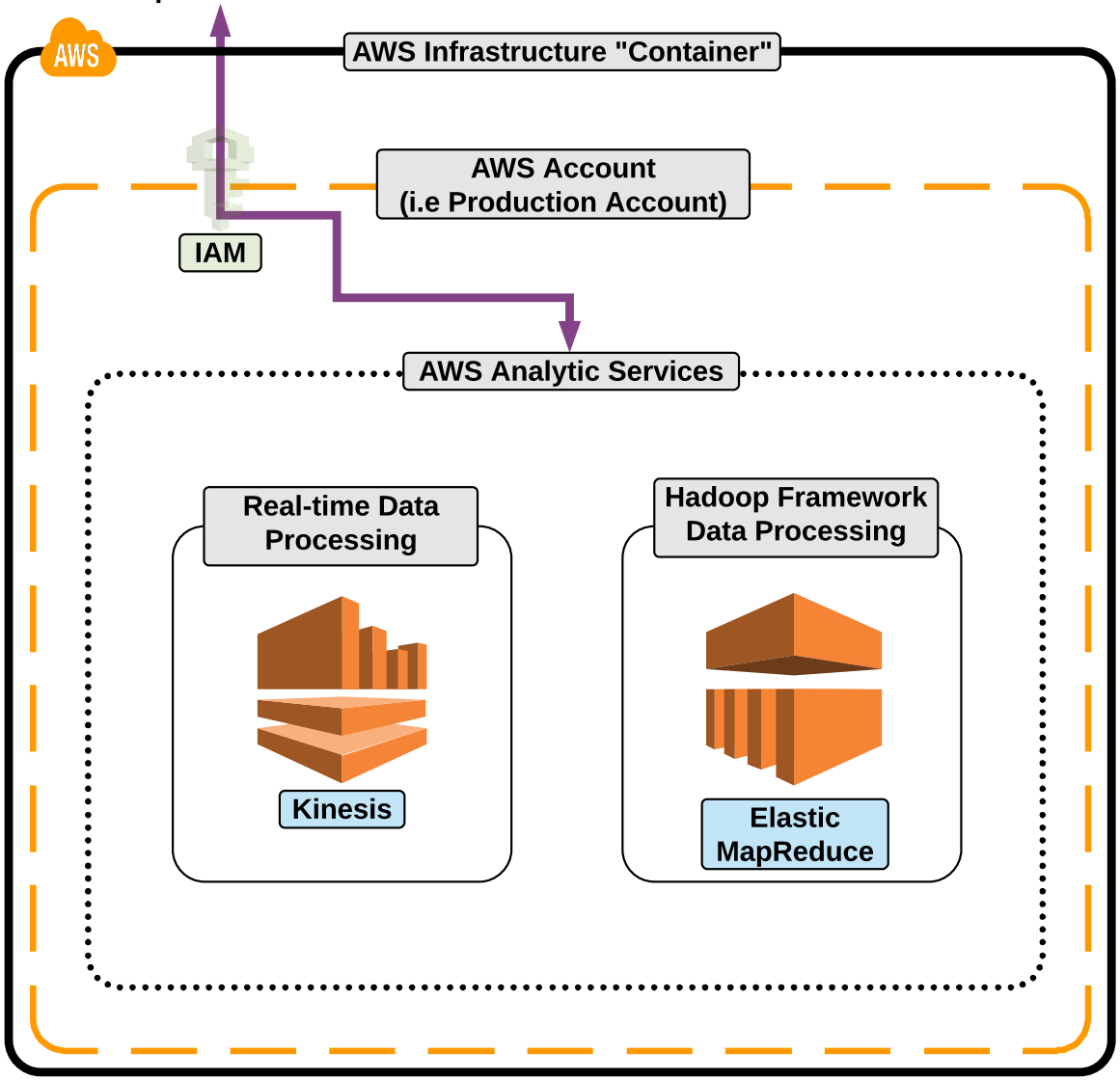
(Analytic Services)

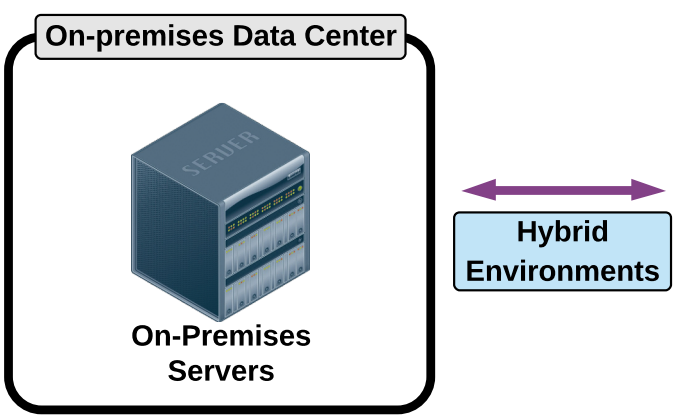
AWS provides to primary service for data analytics. ***Kinesis*** for real-time data processing, and ***Elastic MapReduce*** for Hadoop framework data processing.

Go Back

Appendix







Account & Services Layer

Physical & Networking Layer

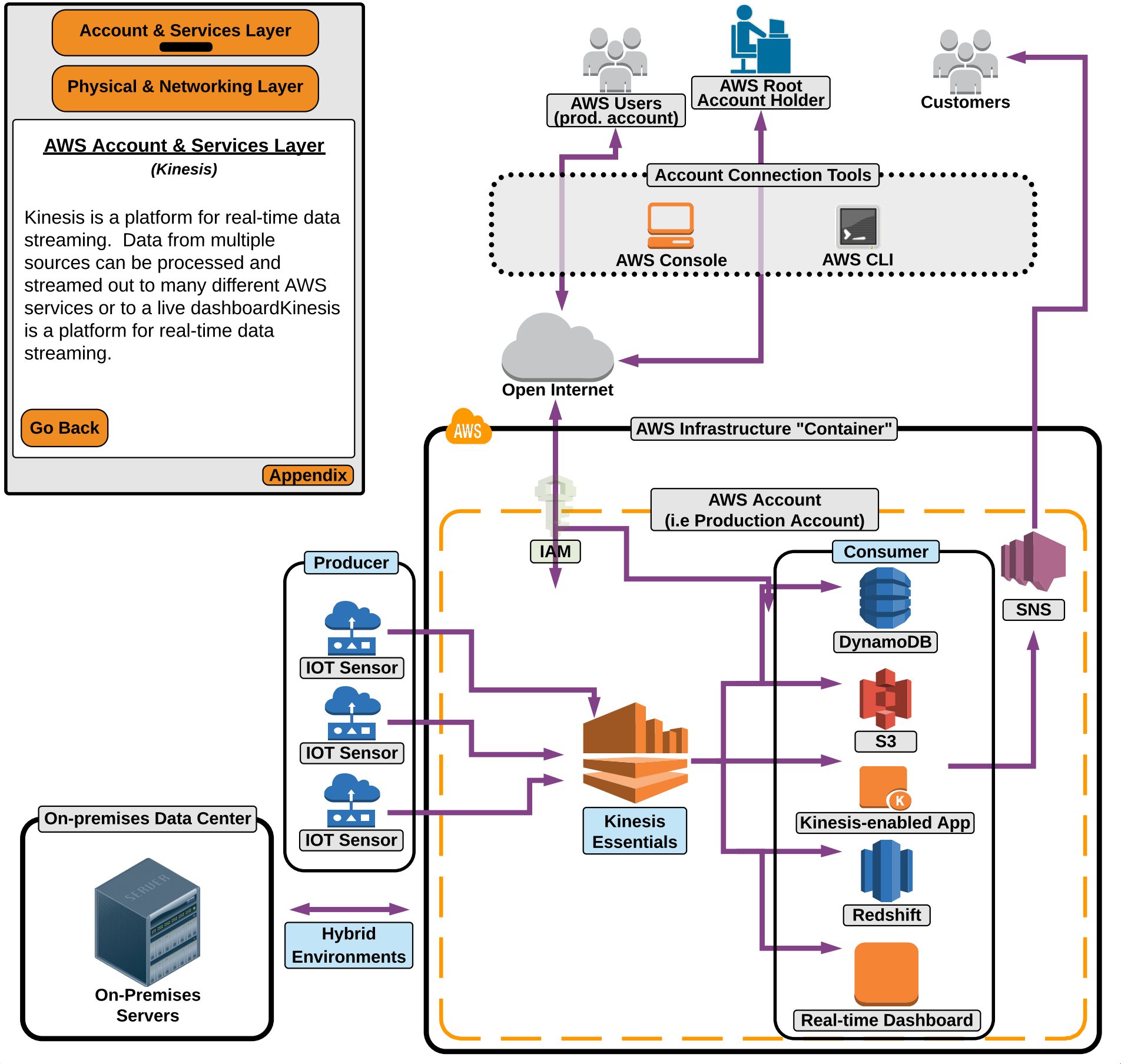
AWS Account & Services Layer

(Kinesis)

Kinesis is a platform for real-time data streaming. Data from multiple sources can be processed and streamed out to many different AWS services or to a live dashboardKinesis is a platform for real-time data streaming.

Go Back

Appendix



Account & Services Layer

Physical & Networking Layer

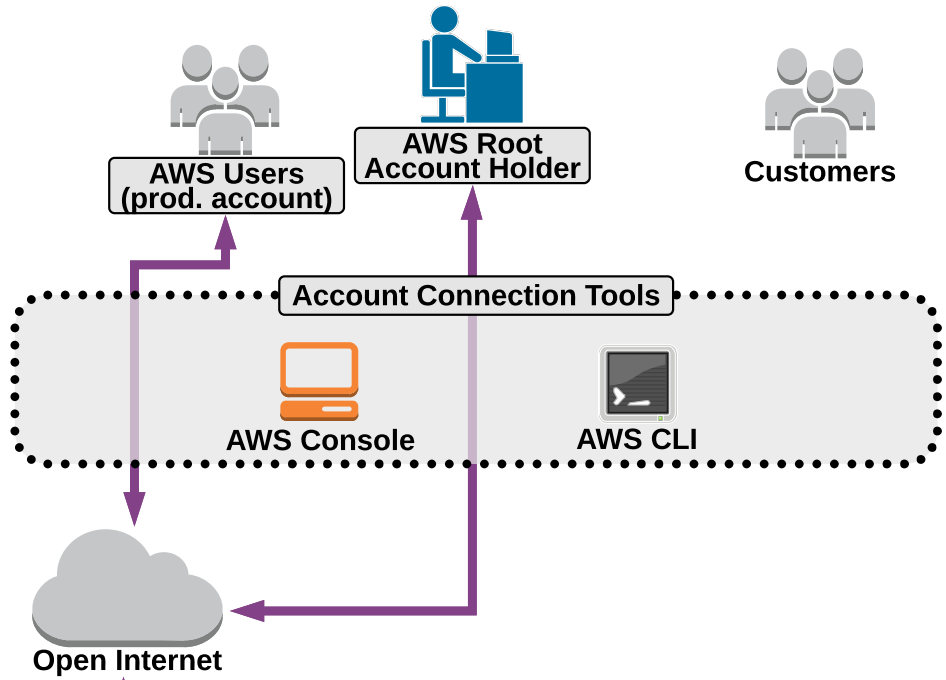
AWS Account & Services Layer

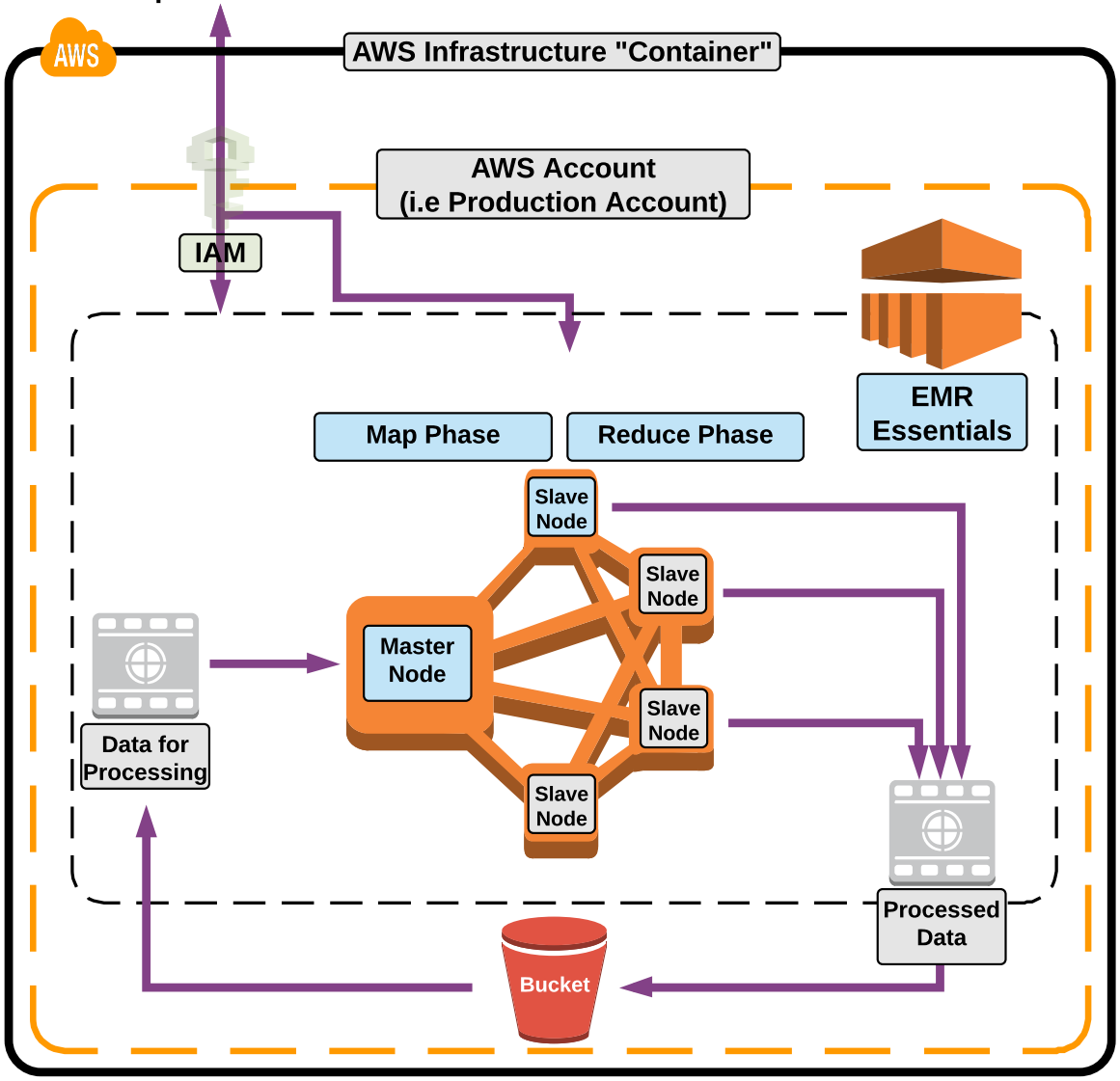
(Elastic Map Reduce)

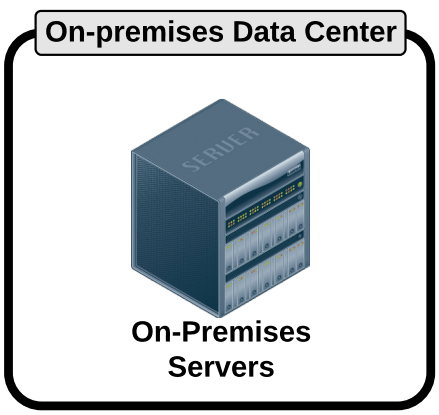
Elastic MapReduce provides EC2 instance clusters for Big Data processing based off the Hadoop framework.

Go Back

Appendix







Hybrid Environments

Analytic Services:

X

Kinesis Essentials | Benefits | When to Use

Analytic Services:

X

EMR Essentials | Benefits | When to Use

Analytic Services:

Kinesis Producers:

- Producers are devices that collect data for Kinesis processing.
- You build producers to continuously input data into a Kinesis stream.
- Producers can include (but not limited to):
  - IoT Sensors
  - Mobile devices (cell phones)
- You can have literally thousands of different producers and scale based on need.
  - The more data you want to process, the more "shards" you add to your Kinesis stream.
  - Each "shard" can process 2MB of read data per second, and 1MB of write data per second.

Analytic Services:

X

Kinesis Consumers:

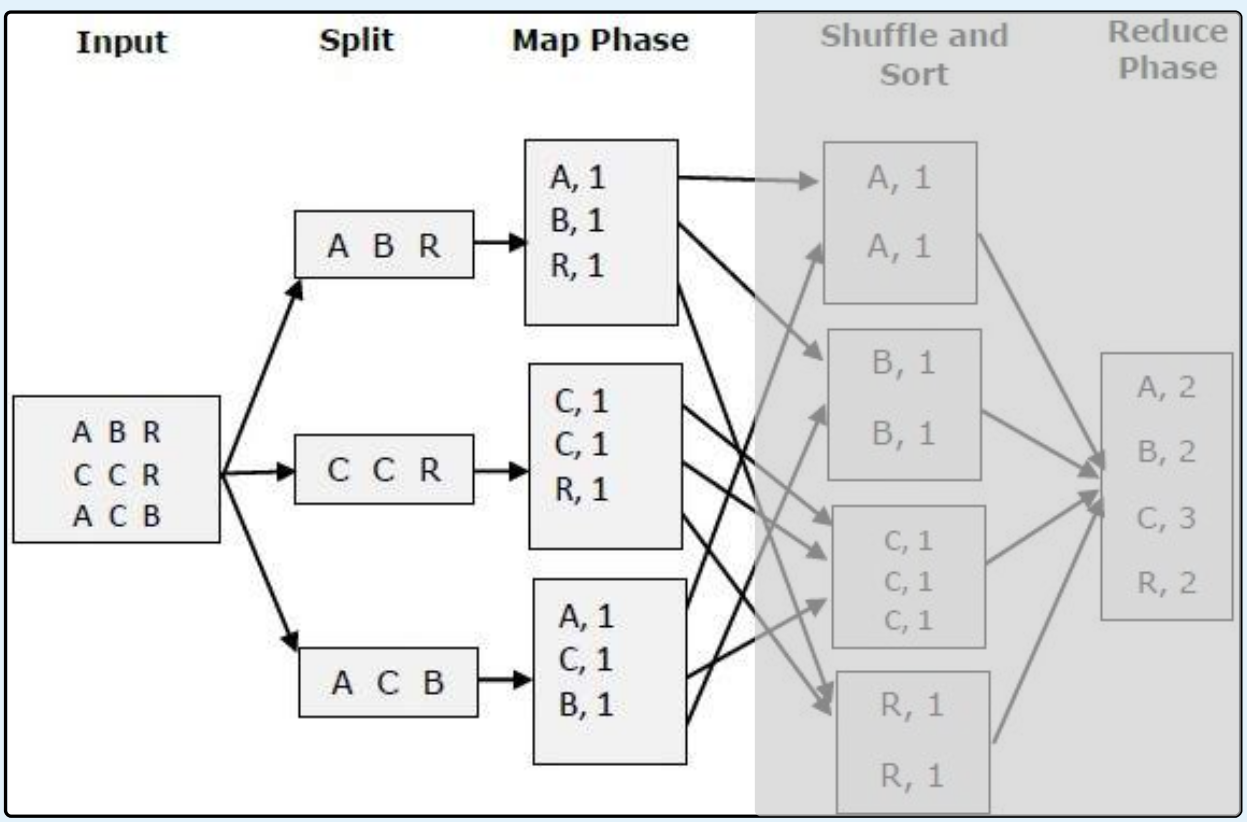
- Consumers consume the stream's data.
- This is done concurrently (multiple consumers can consume the same data at the same time).
- Consumers include (but are not limited to):
  - Real-time dashboards
  - S3
  - Redshift (data warehouse)
  - EMR
- Any application (one you create) can consume the streams data.

- Kinesis keeps 24 hours of streaming data stored by default, but can be configured to store up to 7 days.

Analytic Services:

EMR Map Phase:

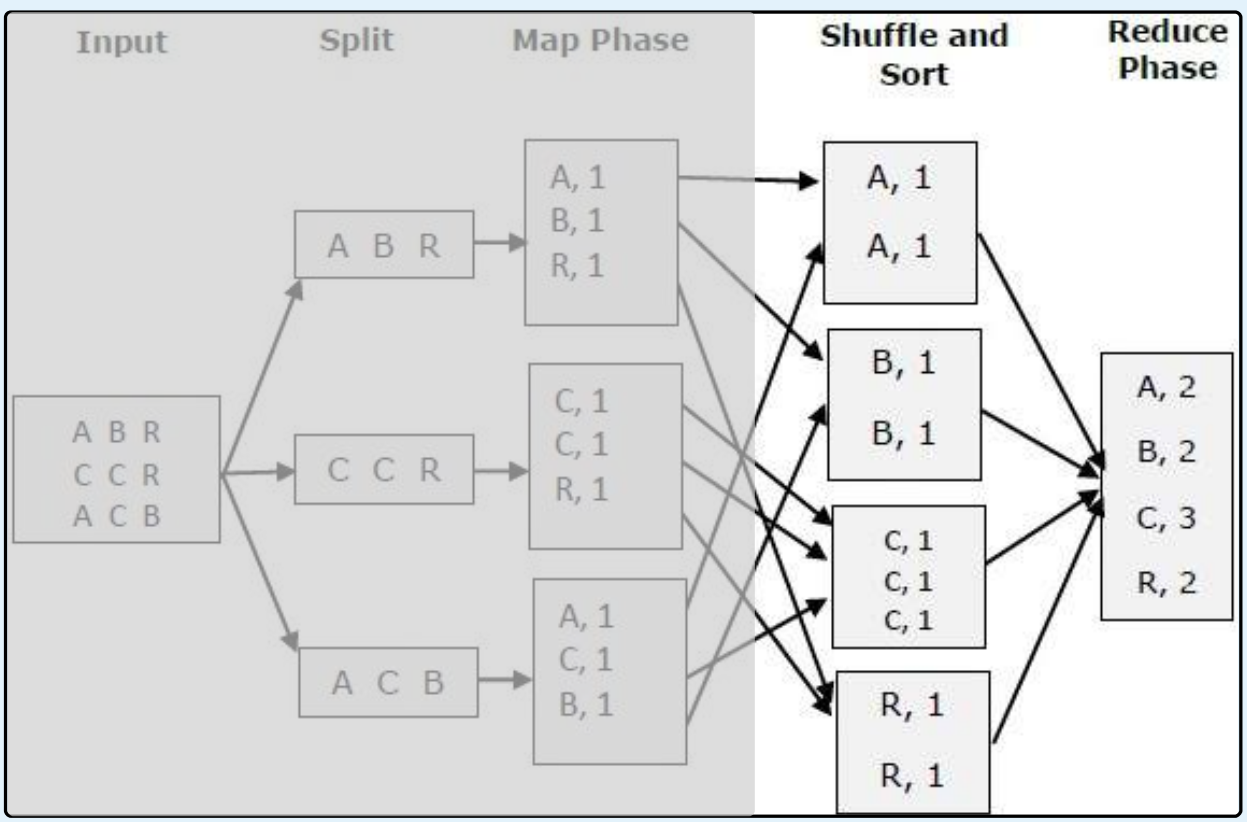
- Mapping is a function that defines the processes which splits the large data file for processing.
- During the mapping phase, the data is split into 128MB "chunks".
- The larger the instance size used in our EMR cluster, the more chucks you can map and process at the same time.
- If there are more chucks than nodes/mappers, the chunks will queue for processing.



Analytic Services:

EMR Reduce Phase:

- Reducing is a function that aggregates the split data back into one data source.
- Reduced data needs to be stored (in a service like S3) as data processed by the EMR cluster is not persistent.



Analytic Services:

X

Elastic MapReduce Essentials:

- Amazon EMR is a service which deploys out EC2 instances based off of the Hadoop big data framework.
- EMR is used to analyze and process vast amounts of data.
- EMR also supports other distributed frameworks, such as:
  - Apache Spark
  - HBase
  - Presto
  - Flink

General EMR Workflow

- Data stored in S3, DynamoDB, or Redshift is sent to EMR
- The data is ***mapped*** to a "cluster" of Hadoop ***Master/Slave nodes*** for processing.
- Computations (coded/created by the developer) are used to process the data.
- The processed data is then ***reduced*** to a single output set of return information.

Other Important EMR Facts

- You (the admin) has the ability to access the underlying operating system.
- You can add user data to EC2 instances launched into the cluster via bootstrapping.
- EMR takes advantage of ***parallel processing*** for faster processing of data.
- You can resize a running cluster at any time, and you can deploy multiple clusters.

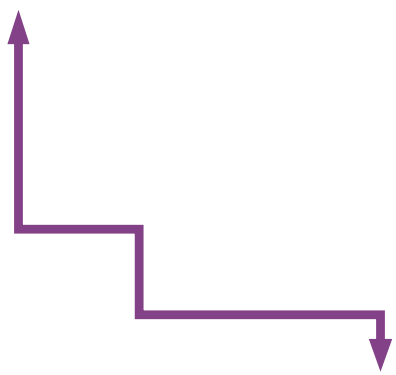
Analytic Services:

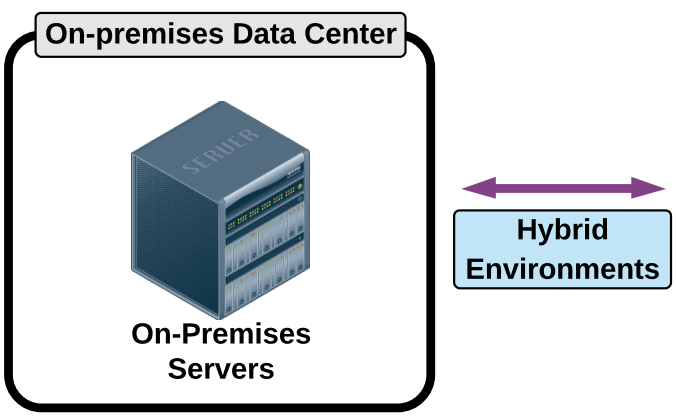
X

EMR Master node:

- A node that manages the cluster by running software components which coordinate the distribution of data and tasks among other (slave) nodes for processing.
- The master node tracks the status of tasks and monitors the health of the cluster.

Appendix



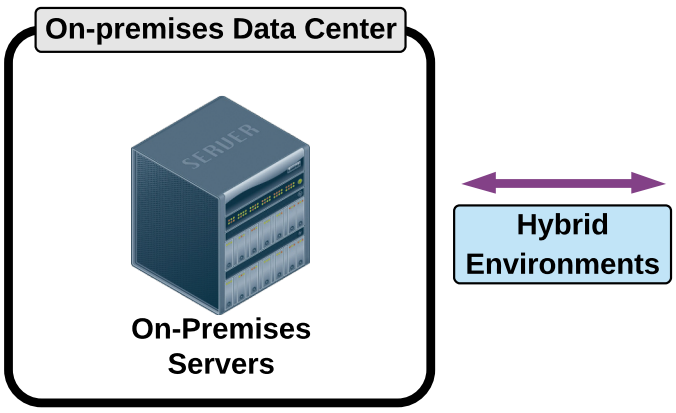


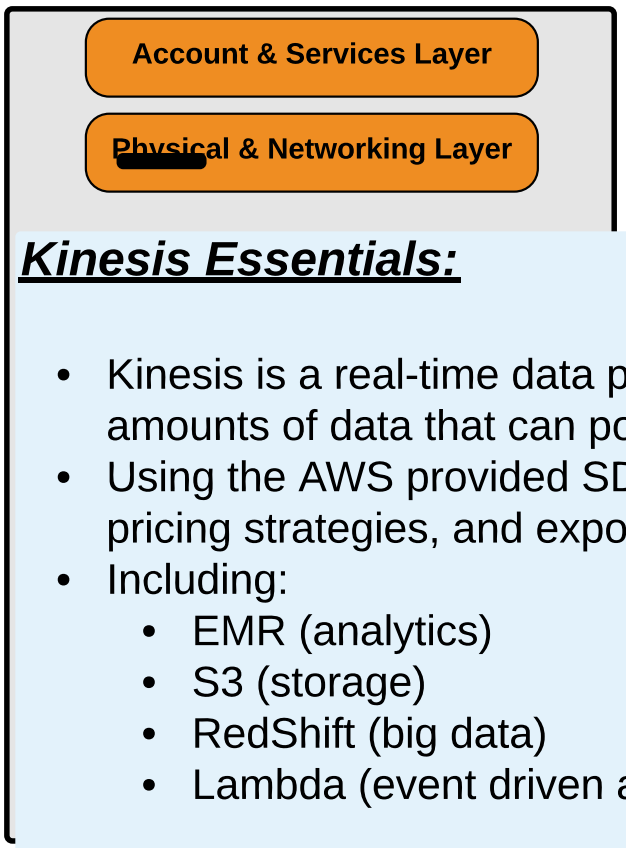
Analytic Services:

EMR Slave Nodes:

- There are two types of slave nodes:
  - **Core node**:
    - A slave node has software components which run tasks AND stores data in the Hadoop Distributed File System (HDFS) on your cluster.
    - The core nodes do the "heavy lifting" with the data.
  - **Task node**:
    - A slave node that has software components which only run tasks.
    - Task nodes are optional.

Appendix



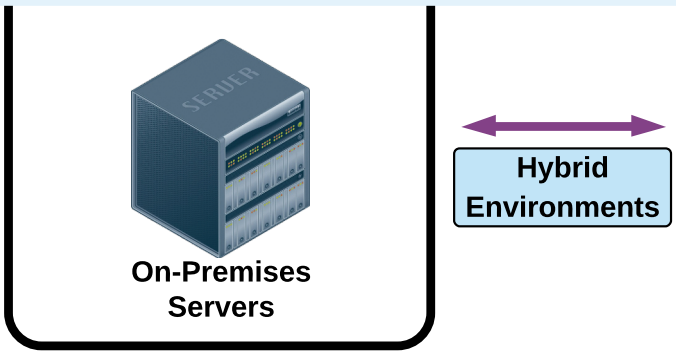


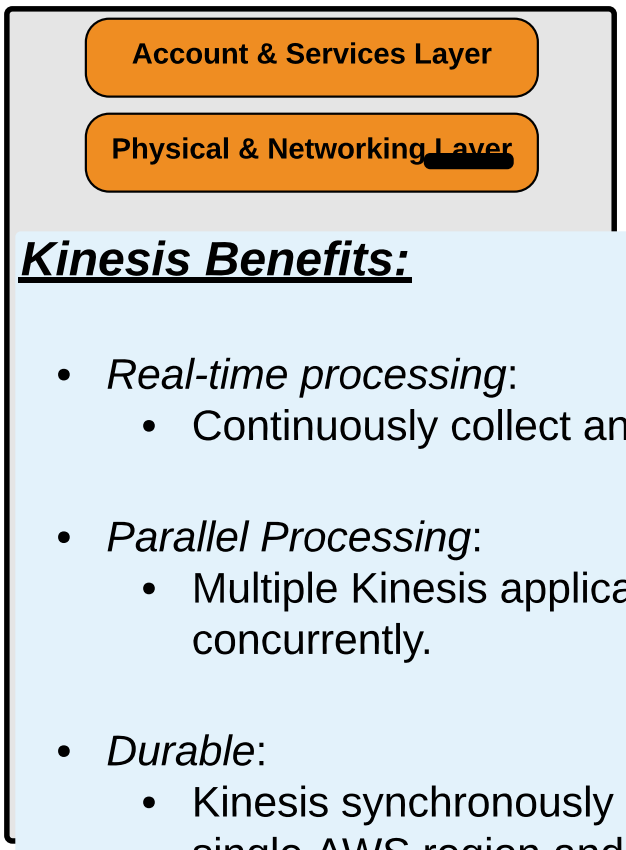
Kinesis Essentials:

- Kinesis is a real-time data processing service that continuously captures (and stores) large amounts of data that can power real-time streaming dash boards.
- Using the AWS provided SDKs, you can create real-time dashboards, integrate dynamic pricing strategies, and export data from Kinesis to other AWS services.
- Including:
  - EMR (analytics)
  - S3 (storage)
  - RedShift (big data)
  - Lambda (event driven actions)

Kinesis Components

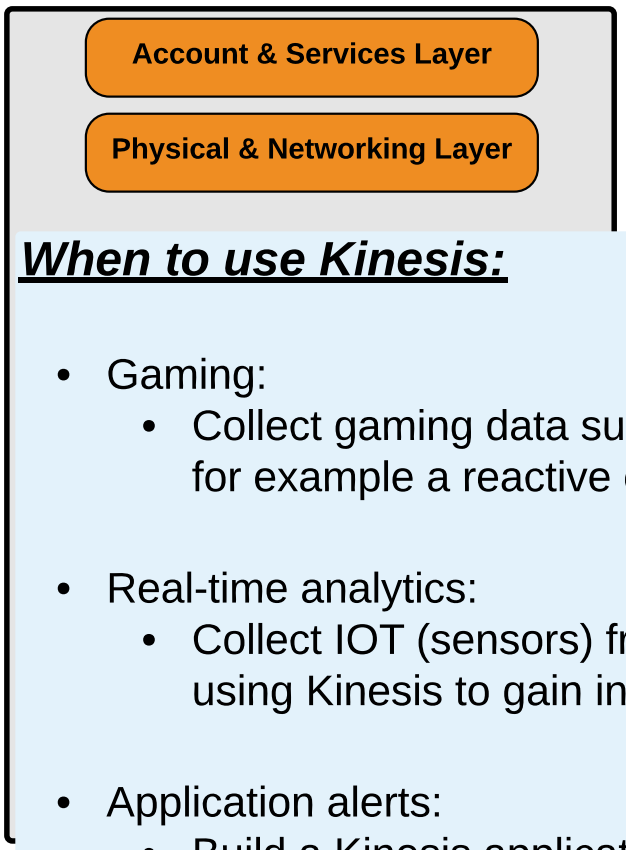
- Stream
- Producers (data creators)
- Consumers (data consumers)
- Shards (processing power)





Kinesis Benefits:

- *Real-time processing*:
  - Continuously collect and build applications that analyze the data as it's generated.
- *Parallel Processing*:
  - Multiple Kinesis applications can be processing the same incoming data streaming concurrently.
- *Durable*:
  - Kinesis synchronously replicates the streaming data across three data centers within a single AWS region and preserves the data for up to 24 hours.
- *Scales*:
  - Can stream from as little as a few megabytes to several terabytes per hour.



When to use Kinesis:

- Gaming:
  - Collect gaming data such as player actions and feed the data into the gaming platform, for example a reactive environment based off of real-time actions of the player.

- Real-time analytics:
  - Collect IOT (sensors) from many sources and high amounts of frequency and process it using Kinesis to gain insights as data arrives in your environment.

- Application alerts:
  - Build a Kinesis application that monitors incoming application logs in real-time and trigger events based off the data.

- Log / Event Data collection:
  - Log data from any number of devices and use Kinesis application to continuously process the incoming data, power real-time dashboards and store the data in S3 when completed.

- Mobile data capture:
  - Mobile applications can push data to Kinesis from countless number of devices which makes the data available as soon as it is produced.