AWS Certified Solution Architect – Professional

Domain	% of Examination
Domain 1: Design for Organizational Complexity	12.5%
Domain 2: Design for New Solutions	31%
Domain 3: Migration Planning	15%
Domain 4: Cost Control	12.5%
Domain 5: Continuous Improvement for Existing Solutions	29%
TOTAL	**100%**

Contents

Section 3: Designing Solutions for High Availability and Business Continuity

Business continuity and Disaster Recovery

- Resiliency and Fault Tolerant
- Redundancy and high availability
- Cost optimization
- Performance
- Security
- Monitoring
- Saleable and Elasticity
- Ease of deployment
- Migration and Hybrid architectures

Fault Tolerant: Ability of a system/application/infrastructure to be operational even if one or more system fails is call fault tolerant. A fault tolerant system should be the one which can recover/failover with minimal or no human interventions on a even of any failure.

AWS recommended to create a library of own AMI, with own standard & best practice. Ensure the AMI is up to date and stable for production usages.

Floating IP (elastic IP) = the IP that can be shift from the primary instance to secondary on an event of any failure.

Create regular EBS snapshots – EBS are region specific, to be used in another region we need to copy the snapshot to the desired region before creating the EBS volume from that snapshot. EBS are region specific and EBS volume are AZ specific.

Multiple site architecture: In this approach the goal is to have two or more independent copy of each application stack into two or more availability zone (site). Every application tier need to have redundant copy , in case of failure the traffic can be route to the alternative availability zone (site) – this can further be made more effective by introducing elastic load balance which can automatically route traffic between AZ, OR having a route53 health check created to route traffic between different region, alternatively one can also use elastic ip for routing traffic, so that when the instance fails the elastic ip can be remapped to another running instance in same/different AZ. Implement Autoscaling to address flatulating production load.

Disaster Recover – Recovery plan from a system failure to resume business continuity without impacting business operation.

Benefit of having AWS as alternative site	Counter Argument
Don't need to negotiate contract with other provide in another region.	Avoiding vendor locking – have a multi-vendor/multi-cloud landscape.
One can use the same underline AWS technology across region	Need to have trained staff in more than one cloud technology – will be cost intensive to maintain multi talent pool.
One can use same tool/AWS build artifacts/API across region	One need to maintain different tool/build artifacts/APIs for different cloud.

Recovery Time Objective (RTO) – Time taken after a disruption to restore a business process to its service level agreement as define by the operational level argument (OLA).

Recovery Point Objective (RPO) – Acceptable amount of data loss measure in time. For example, If the disruption occurs at 12 Noon and RPO is 15 minutes, the system should be successfully able to restore all its transactions which have occur till 11:45AM.

Note: RPO and RTO is defined by the financial impact to the business when systems are unavailable.

- Disaster Recovery Plan for Commute from AMI backup aspect:
  - AMIs can help in restoring failed instance quickly.
  - AMIs can be used within the context of the autoscaling to scale out DR site on an event of primary site failure.
  - AWS recommended to maintain a library of preconfigure AMIs, including the application stack, so that on an event of a disaster these can be used to create new (EC2) instances to replace the old/failed once.
  - Keep the AMIs available on DR sites.
- Disaster Recovery Plan for Storage from S3 Fault tolerance aspect [FOR OBJECT STORAGE]:
  - Can be used as primary object storage.
  - Highly redundant, with 99.99999999999 (11 9's) redundancy rate. Objects stored within S3 buckets are automatically backed up in multiple facilities within the region.
  - Additional security can be provided within S3 bucket for ensure high degree of data retention through MFA for delete operation, versioning, cross region replication – for auto backup in DR site, bucket level policies & object level policies.

- One can also use S3 Glacier storage class for archiving/storing data at very LOW COST, however RTO needs to be set to meet the recovery time as object archive in S3 Glacier takes longer time (usually 3-5 hours).

- Disaster Recovery Plan for Storage from EBS Fault tolerance aspect [FOR BLOCK STORAGE]:
  - Point-in-Time snapshot can be created to backed up EBS content into S3 bucket.
  - Snapshot created can be used to create a new EBS volume & then connect it the EC2 instance to replace failed EBS volume.
  - EBS volume data are stored in different system within the Availability Zone (AZ), this provide protection from single system failure within AZ.
- Disaster Recovery Plan for Storage from Storage Gateway prospective
  - Storage gateway provides an easy means to backup/store on-premises data on to AWS cloud. Customer can download VMware image from AWS console, and install a VM instance from the same on premises which will provide NFS/iSCSI interface for backup/storage.
  - Storage Gateway – File Gateway (NFS interfaces) [OBJECT STORE]: files are asynchronously backed up into S3 bucket, there is a one-to-one mapping between on premises files and S3 objects. User can ALSO access the S3 objects(files) from the S3 bucket. S3 sub resources like lifecycle policy, versioning etc. can be implemented on the uploaded file(objects).
  - Storage Gateway – Volume Gateway - Cached | Storage Mode (iSCSI interface) [BLOCK STORE]: Unlike File gateway, in case of the volume gateway, on premises files are stored into S3 bucket as BLOCK store. There are two possible operating modes – *Cached Mode* where frequently use files are cached on-premises while its asynchronously backed up on S3 bucket as volume store, in *Storage Mode* there is NO local file store everything is backed up on S3 as EBS snapshot. One can't access the files stored in S3 directly – to access the files one need to create EBS volume from form the snapshot (*in case of cache mode, first snapshot needs to create from the volume store prior to creating EBS volume*).
  - Storage Gateway – Tape Gateway: it provides an iSCSI VTL (Virtual Tape Library) interface to backed up on premises files into virtual tape data store within S3 bucket or can be achieve into S3 Glacier. It consists of virtual media changer, virtual tape drive, and virtual tapes.
- Disaster Recovery Plan for Storage from AWS Import/Export prospective:
  - If large amount of data needs to migrate (backed up) quickly into AWS one can use AWS import/export feature – solution like snowball, snowball edge and snowmobile can be leverage for the same.
  - Rule of thumb – "*if the data takes more than week to transfer over the available connection (VPN/Direct Connection/Open internet), then better use AWS import export instead*".
- Disaster Recovery Plan for Storage from VM Import/Export prospective:
  - Ease means to transfer VM image (hypervisor base virtual image) from/to AWS.
  - This can be helpful in creating DR site at AWS or on premises.
  - NOTE: VM Export is ONLY AVAILABLE to those instances which are initially brought in using VM import feature. AMI based EC2 instance cannot be exported using AWS Export feature.
- Disaster Recovery Plan for Storage from RDS Fault Tolerance aspect:
  - RDS Multi-AZ: Primary-to-Secondary synchronous data replication within a region. On an event of any failure the primary DB instance DNS entry will be automatically swapped with secondary DB instance. Help in archiving zero downtime during system patching/upgrade.
  - Read Replicas: For read intensive applications, a read-only DB instance can be provide with asynchronous data replication from the primary instance. On an event of any failure read replicas can be promoted
  - Automated Backup: Automated backup in conjunctions with transaction logs to help recover failed DB instance with RPO up to 5 min.
  - Manual Snapshot: this are necessary for backing up a DB instance OR restoring a DB instance in another region.
  - Can be synced from on premises database to RDS and vise-versa – failover from the on-premises to AWS OR AWS to on-premises is possible.
  - RDS instance size can be upgraded to a bigger size BUT can't be downgraded to a smaller size.
  - For DynamoDB one can copy the data into a S3 bucket and using data-pipeline it can be copied to another region DynamoDB. Also, one can use cross-region replication of the DynamoDB to replicate the data into

another region DynamoDB table. For Sync operation (preparation phase), the dynamo DB can be created with lower read-write capacity, during recovery phase the capacity can be altered to meet the production demand.

- For redshift database: During the preparation phase one can create snapshot of the database in to S3 within same/different region, during the recovery phase redshift cluster can be (re)created from the copied snapshot.

- Disaster Recovery Plan for Networking, on an event of a disaster the ability to quickly shift the network settings from the production site to the DR site is very much required.
  - Route53 AWS managed service helps in quickly change and restore network connectivity (routing) on an event of a failure.
    - Route53 is an AWS managed, highly available and highly scalable global service.
    - Route53 automatic Health-check configuration can route incoming traffic to healthy site.
  - Using Elastic IP (floating IP): these are the static IPs that can be configured to the system, on an event of any disaster these IPs can be swap with the DR instances to quickly resume the production workload.
  - Elastic Load balancing:
    - It can load balance the traffic within a given region
    - Elastic IP addresses can be allocated to the ELBs, which make the transition simple on an event of any disaster. (DNS can configure to pass the load to the elastic IPs, which can load balance the traffic among healthy instances).
  - Amazon VPC: On an event of a disaster Amazon VPC can help in extending the network topology to the cloud. This is mostly applicable in extending the on-premises application to the cloud during disaster.
  - Amazon Direct Connect: This helps in connecting the AWS cloud network to the on-premises network by providing a low latency, high bandwidth dedicated physical connection.
- Disaster preparation phase: Before disaster strikes, the stage where the disaster/backup plans are made are called disaster preparation phase.
- Disaster recovery phase: After disaster strikes, the steps that needs to be taken to recover from the disaster is call disaster recovery phase.
- Different Strategies for Disaster Recovery are: These strategies can be mix and match to find a right solution that meets the RPO and RTO expectations.
  - **Backup & Restore**
    - **Backup solution:** Snapshot | Object (file copy) | Volume copy.
    - **Storage solution:** S3|Glacier | Tiring solution (according to the usages storage class will be selected).
    - **Data transfer solution:** Over internet using custom solution| AWS File Gateway | AWS Volume Gateway – Stored/Cached mode | AWS Tape Gateway | AWS import/Export | AWS VPN Connect |AWS Direct Connect
    - **Key Consideration:**
      - ✓ Select appropriate means: tools/services for backup/storage/data transfer solutions.
      - ✓ Define retention policy – till when a backed-up objects/files/snapshots needs to be stored? How soon they need to be made available? Can they be archived?
      - ✓ Ensure adequate security measures are applied to secure backed up contents.
      - ✓ Plan regular game-day/dry run to ensure the data are correctly backed up and can be restore correctly when needed.
  - **Pilot Light:** DR strategy where a minimal version of the cloud is always running on the DR site. During preparation phase: The core component will be identified and a minimal version of the solution will be running on the DR site and will be in Sync with the production. During restore phase the other components on the solution can be quickly provisioned and the production traffic will be routed to the DR site. RDS site may be upgraded to meet the production traffic.
    - **Key Consideration**
      - ✓ Backed up AMI can be use to provision the remaining component.
      - ✓ Running Instance size should be upgraded to meet the production demand.
      - ✓ Add resilience to the DR Site: Plan for fallout instance failure.

- **Warm Standby:** Scale down version of the fully functional version of the application will be running on the DR site. On an event of any disaster, production traffic will be automatically route to DR site and the application will be automatically scaled up to meet production requirement.
    - **Key Consideration**
        - ✓ Increase the size of the EC2 instance fleet (horizontal scaling).
        - ✓ Increate the sized of the EC2 instance (vertical scaling).
        - ✓ Implement Route53 health checks to automatically route production traffic to DR site, on an event of a disaster. OR manually change the DNS entries of the Route53 to route production traffic to DR site on an event of a disaster.
- **Multisite:** In case of multi-site (active-active) DR strategy, a fully functional redundant site will be running in parallel to the production site. ON an event of any failure production traffic will be severed by the DR site.
    - **Key Consideration**
        - ✓ Configure weighted or equivalent routing mechanism to distribute production load between production site and DR site. (manually overwrite the Route53 DNS configuration to route all traffic to DR site)
        - ✓ Based on the RPO, synchronous or asynchronous replication need to be incorporate.
- Data Replication: For data replication following key things needs to be keep in mind.
    - Distance between the sites – larger the distance more latency & jitter.
    - Availability of the bandwidth – How two sites are connected? Type of connection VPN/Direct Connection etc.
    - Data rate required by the application – it should be always lesser than that of the available bandwidth.
    - Replication methodology: serial or parallel connection for replication.
    - Synchronous (*150 ms latency*) or Asynchronous replication: Synchronous replication data are automatically update; Asynchronous replications are eventual consistent.

- Self-Healing: The following services are self-healing services.
    - SQS (Simple Query Service) can help in decoupling the applications which can help in automatic reprocessing of the failed instances without any human interventions
    - CloudWatch with EC2 instances autoscaling, which can automatically detect un healthy instance using CloudWatch metrics and terminate unhealthy instances.
    - EC2 autoscaling instances – which can be use to replace the unhealthy instance with healthy once.
    - AWS Glacier/S3 perform integrity check and ensure that all objects are constantly availability across different S3 systems.

Section 4: Design for operational complexity and cost control

AWS Organization Overview

- Possible AWS Infrastructure configuration from account perspective are
    - One Account for the company, and individual account per employee. [Typically situated for small company.]
    - One Account for the company, and IAM user per employee. [Seen mostly on mid-size companies.]
    - One Account for the company, and account per company department. [Most common.]
- For above possible infrastructure configuration, one can use one of the following bill consolidation process:
    - Manage separate bill for each account [Maintain separate payer account}.
    - Have consolidated bill for each of the linked account, into a master account (payer account). [single consolidated billing.]
- Advantages of having a consolidated billing are:
    - Single bill – need not to deal with each account/department individually.
    - Ease tracking – per account level usage
    - Ease of user maintenance & standardization across all account(department) of the organization.

- In case of consolidated billing, AWS consider all linked account as one single account (master account), where one can take cost benefit as AWS services are priced in tire/usages with increasing usage unit price reduces.

AWS Cost Explorer

- Helps in viewing AWS cost as a graph
- Filter graphs as per – Region, Availability Zone, AWS service, API Operation, custom cost allocation tag, Amazon EC2 instance type, purchase options, usages type, usage group and more.
- Cost can be filtered per linked account, in case of consolidated billing.
- Cost forecasting base on historical data.

AWS Budget

- Two budgets free per account, up to 20,000 budgets can be configured. Fee will be applicable for budgets exceeding free budgets.
- AWS Budget helps in planning & managing service usages, service cost, and instance reservations.
- Following things can be viewed with ease in the AWS Budget
  - Account usage to date, including how much capacity of reserved instance has been utilized.
  - Current usages against the budgeted usage or free tier limit.
  - Predicted usage/bill at the end-of-the month.
  - SNS notification can be configure which can notify when the forecast exceeds the budget cost per account/service OR when the usages exceeded the budgeted cost.
    Different Types of budget
  - **Cost Budget:** Plan how much is plan to spend on a service
  - **Usage Budget:** how much is plan to spend on a single/multiple service together.
  - **RI (Reserved Instance) Utilization Budget:** Define utilization threshold and received alerts when the utilization is below the threshold limit, this protect from under-utilization of reserved instance (RI).
  - **RI (Reserved Instance) coverage Budget:** Define coverage threshold and received alert when the number of instance hours fall below the budgeted hours. This ensures that RI instance hours are in aligned with the budgeted hours.
- Linked account can create budgets of their, using IAM policy one of the member account (lined account) can be granted access to read/create/edit/delete budget in other linked account – this is typically done to provide full control to financial department (linked account) to manage other account/master account budget – using AWS organization one can provide/revoke access to linked account from access AWS budget.
- AWS cost explorer helps in visualizing the costing using different graphs - if cost explorer is NOT ENABLE, it gets automatically enabled when the budget is created for the very first time.
  Best Practice Recommendation from AWS: Its recommended NOT to run any AWS services in master account, with ONLY exception to host a single S3 bucket where all invoices/bills are store for future references.
  There are two ways to organize infrastructure from Billing prospective: a) using AWS organization & consolidate billing b) having a single account with multiple VPCs for each individual group.

Consolidated Billing	Single Account with multiple VPCs.
Easier form AWS architecture prospective	Simple billing, one bill for everything
Volume discount – all discount related to all linked account can be better mange when consolidated	Per account discount – easy governance.
Need IAM policy account accounts to view manage account bill/budget	NO IAM policy needed.
	Complex policies, to restrict/allow resource level access.
Tag is complex, as it needs to be done across different accounts.	Tagging is simple, as it needs to be managed within a single account.

	Setting VPCs get completed from costing prospective.

Note: In case of consolidate billing by default Reversed Instances are shared within all linked accounts. This can be disable if desired (Reserved instance sharing = turn on/turn off from the billing and preference page) by the master account (payee account).

- AWS CloudWatch can help in creating alert based on the usages and can send alert notifications for
  - When a set threshold of a service usages is reached as per the threshold defined in the billing alerts.
  - When the usage exceeds the amount that is defined in the billing alerts.
  - When signed up, it can also send alerts/notification when the pricing is changed.
- AWS Tags are the key value pairs which can be used for organizing and categorizing AWS resources. There are two try of tags a) system (AWS) generated and b) custom tags a tag cannot be deleted or merged. Logical grouping of the resources is useful in – projects, cost centers, development environment, application, department.
- Resource group: Group of AWS Resources that share one or more common tags ***within a same AWS region***. Resource group can be of two types – tag base and CloudFormation stack base. Resource group can have nested group from the same AWS region. They can be used for cost exploration or for other bulk AWS operations.

AWS Cost Optimization

- Right Sizing, selecting the right EC2 instance family and size. This needs to be done based the performance benchmarking (based on workload utilization, CPU, RAM, networking, storage size, and I/O – which storage classes to be selected , should be selected busting throughput or provision throughput).
- Continuous monitoring & tagging
- Leverage AWS service for cost optimizing by finding right (new) services.
- Cost explaining for cost optimizing
- EC2 instance purchasing option – OnDemand EC2 instance are most expensive. Don't use it forever, use it for a short span of time (during initial phases) before shifting to the Reserved instance. (One of the reasons for choosing OnDemand instances is when the instance type is new/high demand and availability cannot be guaranteed within a specific AZ). **[RI applicable for the Availability Zone OR a particular Region – base on the choice instance will be reserved]. [One can also have scheduled reserved instances – where instance will be reserved for a specific scheduled.]**
- **Dedicate Host:** Where a specific host is reserved for the client, NO other client instance will be running on that host. This helps in saving by bringing your existing per-socket licenses, per-core licenses or per-VM licenses.
- **Dedicated Instance**: Pay per hour for the instance that are running on single tannate hardware.
- **Capacity Reservation:** reserve capacity for your EC2 instance in a specific availability zone for any duration.
- **Reserved Instance Attribute:**
  - **Instance Type:** EC2 instance type for example m4. large – instance family/size.
  - **Scope:** The availability of the instance scope – within region or within zones (RI limits - **20 max zonal reserved instances and 20 MAX for each of the availability zone.**)
  - **Tenancy:** it's a single tenancy (dedicated) or multiple tenancy.
  - **Platform:** Windows or Lunix

  Offering Class: There are two offering classes for Reserved Instance, a) standard and b) convertible. Standard instance can ONLY be modified but CAN'NT be exchange for another Reserved Instance (RI) with different RI attributes. In case of convertible RI, they can be exchange for other RI instance with different RI attributes, like standard offering classes convertible offering classes can also be modified. Standard offing class instance can be sold in market place BUT convertible offering class instance can't be sold in AWS market-place.

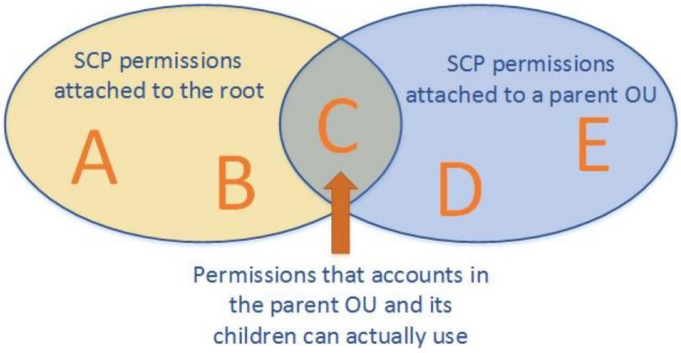
Once a reserved instance is purchase, it cannot be returned – it can only be modified, exchange, sell.

- **AWS Spot Block Instances:** Spot instance which can be blocked for a definite duration (1-6 hours). The pricing is dependent upon the request duration and the availability of the capacity. Instance get automatically terminated at the end the duration or can be manually terminated. NOTE: using RequestSpotFleet API function one can launch a large number of EC2 sport instance at same time. To run RequestSpotFleet one needs to provide Target Capacity, Maximum Bid price, Launch Specification(AMI Id, VPC, subnets or availability zones, security groups, block device mappings, user data, and so forth) , IAM Fleet Role
- **Elasticity**
  - Implement autoscaling – horizontal scaling of the instance when needed scale out and when NOT needed scale out.
  - Non-production workload – start when needed and shutdown (*deprovision)* when not needed.
  - Go Serverless where possible – pay ONLY for the used computation.
  - Use managed services instead of EC2 workloads.

- **AWS Organization:** AWS organization is a service for managing accounts at scale. **It's a global service, can be reached from US EAST url.**
  - **AWS OU are used for centrally managed accounts, automated creation of the AWS accounts, OUs, and Hierarchy AND can be used to enforce Service Control Policies.** (*AWS OU does not manage or control cross account permission; it's been done by the IAM policies*). AWS OU can override permission that are otherwise granted by the IAM policies.
  - The changes made in organization unit have eventual consistency.
  - **AWS Organization Unit is a container of other Organizational Unit (OU) or AWS accounts.**
  - **OU Policy** – are the rule/policy to control AWS services. If the **Service Control Policy** is attached to the root of the OU then the policy is applicable to all the OU. If the **Service Control Policy** is applied ONLY to the AWS account then that is applied only to that particular account.
  - **Feature of AWS OU**
    - **Create Account – OU can be used to create member accounts**
    - **Add new accounts – OU can invite new member account to it**
    - **Can apply service control policies to the member accounts**
    - **Consolidated single billing for all the member AWS accounts**
    - **Manage Hierarchical grouping of the AWS account as per – budgetary needs, security needs, compliance needs. [ONE AWS OU can have up to 5 nested OUs]**
    - **Manage AWS Service / individual API action for the member accounts, once applied this will override the IAM policy access provided by the member account administrator.**
  - **Mode of operation for AWS Organization**
    - **Consolidate billing** *(default feature).*
    - **All features** *(additional feature like service control policies are applied in addition to the consolidate billing) – Once a member account accepts to join a OU request sent by the Master Account by default all OU features will be enabled for the member account. This is primarily done to implement restriction applied from the master account using service control policies. Existing consolidate billing member, can be send fresh invitation to join for full feature mode.*
      Service Control Policies (SCP)– can prevent member account from accessing specific AWS services or even can block uses from leaving from the OU. Service Control Policy defines the services that can be used within the member account, its left to the IAM to provide required access to access AWS services that are granted by the Service Control Policy for that particular account.
      The master account of the organization is not affected by any SCPs that are attached either to it OR any root OR any OU master account is part of. [Master account is immunized by the SCP]
      AWS organization attaches an AWS managed SCP FullAWSAccess to all the root, OUs, and account which ensure building of the organization. In order to restrict access to an OU or an

account replace FullAWSAccess policy with a custom polices with limited permissions.

There are two possible ways to manage member account AWS services
- using SCP whitelisting – in this SCP defines the services that can be access within the member OUs/Accounts and rest other services are explicitly gets denies.
- using SCP blacklisting (default) - in this SCP defines the services that are denied within the member OUs/Accounts and rest other services are implicitly allowed.



- **AWS Organization Integration with other services**
  - **AWS CloudTrail:** CloudTrail from different AWS member account can be maintained in the master account, this will ensure that all trails are centrally logged.
  - **AWS CloudWatch:** CloudWatch can be configured to store all metrics from all member account into one single account. This will enable sharing of the event across organization.
  - **AWS Config:** Organization wide compliance status can be maintained, by sharing AWS Config across organization status.
  - **Artifacts:** All compliance report can be made available across organizations, anyone from the AWS organization can view the report when needed.
  - **AWS Firewall Manager:** Centrally all WAF configuration can be maintained from one single AWS account across AWS organization.
  - **AW Directory Services:** Integration of the AWS directory services, allows seamless sharing of the directory service across multiple accounts and VPCs within those account across AWS Regions.
  - **AWS License Manager:** The integration allows for cross-account discovery of computing resources throughout the AWS organizations to ensure licenses compliance of the resources in the organization.
  - **AWS RAM (resource access manager):** Service that enable sharing of the resources securely with different account OR with other organizational units (OUs). ONLY master account can enable sharing of the resources with other OUs, for resources to be share within OUs all feature needs to be enable – only with consolidated billing one can't share resources. IAM policies needs to be added to secure the shared resources ensuring that who can access the resource etc.

Service	Resource
Amazon Aurora	DB Cluster
AWS ClodeBuild	Project Report Group
Amazon EC2 instance	Reserve capacities, Dedicated Host, Subnets, Traffic mirror targets, Transit Gateway.
Amazon EC2 image builder	Component, Image (AMI), Image recipes.
AWS License Manager	License Configuration
AWS Resource Groups	Resource Group
Amazon Route 53	Forwarding Rules

This is not the full service list that can be shared using Resource Access Manager – for complete list visit : https://docs.aws.amazon.com/ram/latest/userguide/shareable.html

  - This helps in sharing AWS resources cross account. **(example sharing subnet, transit gateway etc).**
  - **AWS Catalogue:** Integrating AWS catalogue can help the organization to maintain a single portfolio of services hosted access the organization.

        - **AWS Single Sign** on across different account within an AWS organization.
    - **AWS Service Linked-Role (**`AWSServiceRoleForOrganizations`**):** *This is an AWS managed role (i.e. one can't edit or add policies to this role),* which helps the master account to enable trusted services in member accounts. When a new account is created in the organization OR a member account accepts an invitation to join the organization this role gets provision in the member's account. ONLY AWS organization can assume this role. This role is NOT NEEDED if the organization is operated ONLY for consolidate billing the role can be deleted, later if all feature (organization all feature mode) needs to be enable the role needs to be restored back. Once organization all feature mode is enabled, then the AWS Service linked role can't be deleted. AWS Service Linked role are excepted from AWS Organization Service Control Policies (AWS OU SCP). AWS OU SCP never effected AWS Service Linked Role.
- **AWS Organization best practices**
    - Use AWS cloud trail to monitor and log API Calls in the mater data S3 bucket.
    - Don't add AWS resources to AWS Master Account, except for those which are absolute necessary.
    - Follow Least Privilege Principle.
    - Provide AWS OU SCP at OU level then doing it at the account level. Policies applied at the OU level, automatically inherited by the accounts so one didn't have to add policies to each individual account within the OU.
    - Avoid have SCP at the root level, this will bar all the accounts to avail the service.
    - Use whitelisting OR blacklisting, DON'T mix whitelisting and blacklisting – as it creates confusion during debugging.

Section 5: AWS Services and Strategies for Deployment and Operation/Monitoring Management

CloudFormation

- Infrastructure as cloud – create /update/delete/version control "***stack***", collection of AWS resources combines together in a single template.
- Simplify Infrastructure management
- Quick replicated of an Infrastructure in different region
- CloudFormation will rollback stack creation, if any of the AWS failed to provision.
- "***Change Set***" will hold ONLY the modified delta of the CloudFormation template, one can review the change and decide upon applying it OR rejecting it.
- ***Deletion policy*** will help in retaining specific resource/information when a specific template is deleted. E.g. when deleted an EBS volume one might take a snapshot to retain its data, similarly for RDS instance before deleting an RDS instance one may like to take a snapshot.

- **Cloud Formation Template Anatomy**

CloudFormation Template Header	Description
AWSTemplateFormatVersion (Optional)	The version of the CloudFormation Template usually defined by the version date.
Description (Optional)	Short description of the template, that define the stack.
Metadata (Optional)	Metadata of the template which can be added to describe the template.

Parameters (Optional)	Value that are passed to the template are called Parameters. One can refer the Parameter section from the resource and the output section.
	Sample Parameter
	"Parameters" : {
	"InstanceType" : {
	"Description" : "Amazon EC2 instance type",
	"Type" : "String",
	"Default" : "m4.large",
	"AllowedValues" : ["t1.micro", "t2.micro", "t2.small", "hs1.8xlarge", "cr1.8xlarge", "cc2.8xlarge", "cg1.4xlarge"]
	}

Mappings (Optional)	Custom define Key/Value mapping which can be use to specific conditional parameters. It works like a lookup table

	One can easily map the value matching to the key using intrinsic function fn:FindInMap in both resource and output sections.
Condition (optional)	Condition that control whether certain resources are created OR whether certain resource properties are assigned a value during stack creation or update. For example: for test environment multi-AZ setup is not desired, this can be achieved through CloudFormation condition element.
Transform (optional)	This is to use for serverless applications (aka Lambda based application), to specify the version of the serverless application module (SAM). On specifying Transforms element, one can use AWS SAM syntaxs to declare resources in the template.
Resource (required)	This is required to specifies the stack resources and their properties for example – for specifying Amazon EC2 instance, s3 bucket.
Outputs (optional)	Describes the values that are returned whenever one views the stack properties. When one declares an output for an S3 bucket name, on calling `aws cloudformation describe-stack` CLI command it displays the name of the S3 bucket.

- **Intrinsic functions:** This are the built-in functions which helps in managing the stack. Intrinsic functions can be used following places within AWS cloud formation template – Metadata, Resource and Output sections. It can also be used in update policy attributes. One can also use intrinsic function in conditionally create stack resources.

Intrinsic Functions	Description
Fn::GetAtt	Fn::GetAtt returns the attribute of a resource in the template.`Fn::GetAtt:[logicalNameOfResource, attributeName]`
Fn::FindInMap	Fn::FindInMap returns the corresponding value mapped to the keys in a two level map that is defined under the AWS cloud formation mapping sections. `{"Fn::FindInMap" : [ "MapName", "TopLevelKey", "SecondLevelKey"]}`
Fn::GetAZs	Fn::GetAZs returns an array of listOFAvailabilityZone for a given region `{ "Fn::GetAZs" : "region" }`
Fn::ImportValue	Fn::ImportValue returns the value of an output exported by another stack. This is typically used to create a cross-stack reference. Things to keep in mind regarding the cross-stack referencing • Cross-stack reference cannot be created across region (within region ONLY). • For each AWS account Export Name must be unique within a region. • One can't delete a stack if it's been referred by another stack. • One can't modified/remove an output value that is reference by another stack.
Ref	The intrinsic function ref returns the value of the specific parameters or resources. `{"Ref": "logicalName"}`

- To further control resources within a stack one can add additional relationship and behavior of the resources specified in AWS CloudFormation template using following CloudFormation template attributes.
  - **CreationPollicy**: TO delay the resource creation until other resource is created successfully (unit CloudFormation Template received specific number of success signal) or based on set timeout. Currently ONLY ***AWS::AutoScaling::AutoScalingGroup***, ***AWS::EC2:Instance*** and ***AWS::CloudFormation:WaitCondition*** supports CreationPolicy. One can use cnf-signal helper script to send signal back, notifying about the progress of the stack creation. (cnf-signal are helped script available ***`/opt/aws/bin/cfn-signal`***)

- **DeletionPolicy**: If you want to retain (preserved) or create a snapshot (backed up) for resources before deleting the stack. This is also applicable to the stack update operation that leads to resource been deleted from the stack. (*Note: this is NOT APPLICABLE to the resources whose physical instance is replaced during stack update operation. For example, where AWS resource property is updated and cloudFormation template replace the resource with a new one.*)

  Deletion Policy Attributes

  - Delete: if delete attribute of the deletion policy is attached to any resource then resource will be deleted once the AWS CloudFormation stack is deleted. By default, if there is no deletion policy attached, then resource will be deleted from the stack. *However: Default deletion policy for RDS (AWS::RDS::DBCluster) is snapshot. For successfully deleting a S3 bucket, all the content within the S3 bucket needs to be deleted.*
  - Retain: if retain attribute of deletion policy is attached to a resources AWS will NOT delete the resource(s), when AWS cloudFormation template stack is deleted.
  - Snapshot snapshot attribute of the deletion policy can be attached to the AWS resources which supports snapshots like EBS volume, ElasticCache , RDS , Redshift etc. AWS cloudFormation creates a snapshot of the resource before deletion the resources.

- **DependOn Attribute**: Specifies the resources creation depends on creation of the other resources.
- **MetaData Attribute**: TO add more metadata to the resources.
- **UpdatePolicy Attribute**: How the resource updates need to be carried out.
- **UpdateReplacePolicy Attribute**:

Nested Stack: Moving the repeatedly using AWS resources within a main template into a separate stack within the main template and refer is where the common AWS resources are needed.

Cross-stack-reference: In order to improve the maintainability and ownership of the stack, standard infrastructure designs are grouped under single stack and been reference by other stack using cross-stack-references. For example, VPC networking related components can be placed under network-stack and the web-application stack can simply reference the resources from the output of the network stack which is maintained by the networking team.

Stack-Policy: The CloudFormation stack policy is a JSON document that defines what can be updated as part of a stack update operation. To set or update the policy, the IAM users or roles must first have the ability to call the `cloudformation:SetStackPolicy` action.

AWS CloudFormation Best Practice

Planning and organizing

1. Organize Your Stacks by Lifecycle and Ownership
2. Use Cross-Stack References to Export Shared Resources
3. Use IAM to Control Access
4. Reuse Templates to Replicate Stacks in Multiple Environments
5. Verify Quotas for All Resource Types
6. Use Nested Stacks to Reuse Common Template Patterns

Creating templates

1. Do Not Embed Credentials in Your Templates
2. Use AWS-Specific Parameter Types
3. Use Parameter Constraints
4. Use AWS::CloudFormation::Init to Deploy Software Applications on Amazon EC2 Instances – use CloudFront helper script cfn-init to install software and deploy application.

Cloud Developer Kit (CDK): is an open source framework to model and provision your cloud application resources through CloudFormation template. In CDK everything is a construct, developers can design, compose and share their custom resources that incorporate unique requirement. CDK can be model either in java or .net.

- **Cloud Formation Best Practice – Planning & Organization**
  - Organize Your Stacks By Lifecycle and Ownership
  - Use Cross-Stack References to Export Shared Resources
  - Use IAM to Control Access
  - Reuse Templates to Replicate Stacks in Multiple Environments
  - Verify Quotas for All Resource Types
  - Use Nested Stacks to Reuse Common Template Patterns
- **Cloud Formation Best Practice – Creating Template**
  - Do Not Embed Credentials in Your Templates
  - Use AWS-Specific Parameter Types
  - Use Parameter Constraints
  - Use AWS::CloudFormation::Init to Deploy Software Applications on Amazon EC2 Instances
  - Use the Latest Helper Scripts
  - Validate Templates Before Using Them
- **Cloud Formation Best Practice – Managing Stacks**
  - Manage All Stack Resources Through AWS CloudFormation
  - Create Change Sets Before Updating Your Stacks
  - Use Stack Policies (*Stack policy helps in preventing resources from unintentional update of the resources*)
  - Use AWS CloudTrail to Log AWS CloudFormation Calls
  - Use Code Reviews and Revision Controls to Manage Your Templates
  - Update Your Amazon EC2 Linux Instances Regularly
- **Cloud Formation Best Practice – Securing Stack**
  - Limiting Access to CloudFormation Stacks with IAM – Limit the IAM user access to the CloudFormation API. *Who can do what?*
  - Use IAM conditions for CloudFormation to limit the creation and update of the AWS resources using CloudFormation using
    - **cloudformation:TemplateURL** – using this policy one can ensure that all create/update/delete stack call are made by the user using a specific CloudFormation Template.
    - **cloudformation:ResourceTypes** – Using resourceType parameter one can enforce that what type of AWS resources are getting created/updated/deleted
    - **cloudformation:StackPolicyURL**– The CloudFormation stack policy is a JSON document that defines what can be updated as part of a stack update operation. To set or update the policy, your IAM users or roles must first have the ability to call the cloudformation:SetStackPolicy action. Using StackPolicyURL parameter one can enforce the stack policy that needs to be used

AWS OpsWorks

AWS CloudFormation deals with management of the cloud infrastructure, while OpsWorks deals with managing applications on a cloud infrastructure. OpsWorks deals in specific requirement of – creating instances , install necessary packages , distribute application to application servers, monitor stack performance , manage security, and permission.

Stack is the basic component of AWS OpsWork.

OpsWorks can work on the specified region alone. One need to mention the region before creating the stack.

AWS OpsWork uses **Chef** and **Puppet** as the management tool for deploying stacks.

Components of Chef

- **Cookbook:** it's a package file that contains configuration information including instructions called as recipes.
- **Recipes:** A recipe is set of one or more instruction written in ruby language syntax that specifies the resources to use and the order in which those resources are applied. Recipes can be run automatically or can be run manually.
- Recourses: In Chef term a resource is a statement in the configuration policy.

AWS OpsWork

- Stack: It's a container of AWS resources like EC2 instance, RDS instance, AWS DynamoDB etc. that are group together to meet a common purpose thus needs to be managed logically. Stack defines some default configuration settings – such as instance operating system: windows or Linux, AWS region etc.
- Layer: A stack may contains multiple layers – a AWS opsWork layer represents a set of EC2 instances that serves a specific purpose in the stack for example, EC2 instance serving as web servers , or EC2 instance serving as application server. Layer give complete control over the instances – having multiple layer add more control but also increase the cost as number of instances increases.
- All custom recipes, related file needs to be bundle into one or more cookbook and store it in a repository – a repository can be a git repository or S3 bucket.
- Once the stack is created some of the properties can be change but some of the properties like "region" are immutable.

Opswork recipe lifestyle event

- **Setup**: Occur on a new instance after it successfully boots.
- **Configure:** Occur on all the stacks instance when an instance enters and leaves the online state.
- **Deploy:** Occur when an application is deployed
- **Undeploy:** Occur when an application is deleted/ undeployed.
- **Shutdown:** Occurs when one stop an instance.

When a lifecycle event occurs on a layer instance, AWS OpsWork execute their corresponding recipes. A layer can have any number of recipes assigned to each lifecycle event.

AWS OpsWorks instance can be any compute resources – it can be an EC2 instance or a on premises instance.

AWS OpsWorks can have a single instance which can belong to multiple layers; however, this is NOT advisable. When an instance state changes corresponding lifestyle, event get triggered automatically which run the associated recipes on the instance.

AWS OpsWorks instance types are:

- **24/7 Instances**: Instance that run till someone manually stop them.
- **Time-base instances**: are those instances that active on specific intervals – setup at specific time and shutdown at specific interval.
- **Load base instance**: These are instance that AWS OpsWork automatically starts base on configure threshold values. [Currently load base instance type ONLY available for Linux instances.]

AWS OpsWorks autohealing : If instance stop communicating with AWS OpsWorks (*AWS OpsWorks agents becomes unresponsive*) then AWS OpsWorks restart (stop and start) the instance. But if the instance is an part of a layer (multiple layer) where auto heading is disable then the instance will NOT restart.

AWS OpsWorks Apps (application)

- AWS OpsWork can install the application automatically when an AWS OpsWork deploy stage gets triggered.
- AWS OpsWork can also install the application manually by running deploy recipes on a live instance.

AWS OpsWorks can be integrated with the IAM policies to control access for - Individual users (developer, administrator) can interact with a specific stack – who are allowed to create stack resources, layer and instances – if user have SSH or RDP access to the specific instances.

AWS OpsWorks Resources Management: One can incorporate other AWS resources, such as elastic ip addresses , into stack.

AWS OpsWorks network setting – one needs to configure two network settings – Public IP address and Elastic IP address. Choosing a public IP address will allow instance to communicate with the internet, along with public IP addresses if there are instances which has Elastic IP address option selected, then the instance will ALSO be assigned

with an elastic ip address. IF an instance with no public ip address and no elastic IP address then the instance needs to be connected to a NAT gateway/instance through which it can reach out to code repository (git) to fetch the recipes.

AWS OpsWork – elastic loadbalancing

AWS OpsWorks ONLY supports classical load balancer – once a CLB is added to the AWS OpsWork stack then it will de-register all the instances register prior and start refresh. ONLY ONE layer can be attached to a AWS OpsWorks elastic load balancer

AWS OpsWorks best practice

- Instance Type for predictable workload use time base instance, for other chose 12/7
- IAM Role for securing AWS OpsWorks resources – use minimum access policy to restrict access for the users.

AWS OpsWork Monitoring and Logging

- For Linux stacks AWS cloudWatch monitoring additional metrics that can be view on the monitoring page.
- For Windows stack AWS cloudWatch standard metric are provided that can be monitor from the CloudWatch console.

AWS CloudTrail can be integrated with the AWS Opsworks which will record the calls all the calls made to the OpsWorks API. AWS OpsWorks will also provide the events logs and Chef logs which one can refer to view the events.

If the AWS Opsworks instances are altered or modified outside the AWS OpsWork then the AWS OpsWorks stack will become un manageable.

`Aws::CharlieInstanceService::Errors::UnrecognizedClientException – security token included in the request is invalid.` This may occur when the AWS OpsWork instance is deleted outside the AWSOpsWorks knowledge.

ElasticBeanStalk

ElasticBeanStalk: AWS Elastic BeanStalk is an easy to use service for deploying scalable web application and services development with supported programming languages on familiar servers like Apache, Nginx, Passenger, Docker, tomcat and IIS.

CloudFormation	AWS Ops Work	Elastic BeanStalk
Where the target is the create a low-level infrastructure across different AWS region.	Where the target to deploy application in AWS infrastructure with more control on the deployment using chef /puppet base scripts.	Where the target is to deploy application into selected servers with minimum maintenance possible and visibility to application health monitoring.
	While onboarding docker image into Opswork, one need to define granular details to onboard the docker image to create Elastic Container Service.	While onboarding docker image into Elastic BeanStalk with minimal configuration required.
	There is NO direct integration of Ops Work with other AWS services like S3 RDS etc. However, one can use service role to access other services from Ops Work.	There is a direct integration of Elastic BeanStalk with other AWS services like S2, RDS etc .

- ElasticBeanStalk provides
  - Supporting of major programming language like java/go/ruby/.net/php/node.js/python.
  - Supporting different platform for running major programming language.
  - Supporting web containers like Tomcat / Passenger/ Docker etc.
  - Can also support custom build platform for hosting not supporting languages/platforms.

- It also provisions the resources needed to run the application like EC2, load balancer etc.

- How to deploy an application on elastic beanstalk.
  - Develop the application
  - Upload the application bundle, like .jar .war file etc.
  - Provide information about the application
  - Elastic Beanstalk automatically launch the application and provision the resources needed for the hosting of the application.
  - Once the environment is provision and the application is successfully launched on the provision resource, elastic bean stalk help in monitoring the application and also deploy a new version of the application if needed. Application monitoring metric and environment status can be monitored through AWS management console or through AWS command line interfaces.

Note: Elastic Bean Stalk does not provide data persistence, within EBS. Once the EBS is stopped the data persistence within its resources will be lost. It's ADVISABLE NOT TO host RDS instance within EBS instance as the RDS data as well as the snapshot will be lost once the environment is stop. One can host the RDS instance outside of the EBS and refer the same using the CName within the application or store the files within S3 buckets.

- For Dev environment, one can consider using the elastic beanstalk as an alternative for test database, but recommended for PROD environment.
- AWS Elastic BeanStalk resources can be accessible directly, however it's advisable not to alter their configuration directly as it may become unusable.
- One can rebuild an environment which has been terminated with 42 days, once terminated environment is rebuild it creates a new resource with the same name, ID and configuration – however data stored earlier will be lost.

Elastic BeanStalk Application: Application server as a container for the environment that run the web-application and the version of the source code, saved configuration, logs and other artifacts that one creates for using Elastic Bean Stalk. (*In case of Elastic BeanStalk, application is conceptually compared to a folder.*)

- Deleting an application is Elastic BeanStalk Application, results in deleting the application source code, logs, configuration and all the version of that application.
- **Application Version**: Application Version is a part of an application which refers to a labeled iteration of deployable code for the web application. Application version points to a unique labeled deployable package of the source code store under S3 location. One can deploy an application version to a running environment or create a fresh application version of the code and deploy the same to the environment.
- **Environment**: With respect to Elastic BeanStalk an environment is a version that deployed onto AWS resources. An environment can deploy ONLY ONE version at a time, if multiple different versions of the code need to deployed side-by-side it needs multiple different environments.
- **Environment Tier**: there are two types of environment tier – *webserver environment* & *worker environment*. Webserver environment are the once that servers HTTP request, while worker environment are the once that servers pulling task from the Amazon SNS (Simple Notification Queue) for processing.
- **Environment Configuration:** Collection of setting for the underline AWS resource for hosting EBS Application, if the configuration changes – new settings are applied on the underline AWS resources or deploys (change) the underline AWS resources with the new settings.
- **Configuration Template:** Template for creating environment configuration for (EBS) elastic beanstalk.
- **Elastic BeanStlck web container** is a component of the webserver that runs Java Servlets / JSP. Apache webserver is HTTP server that manages static pages, caching, redirection while tomcats supports web application features.
- **Elastic BeanStalk Host Manager (HM) –** agent deployed to EC2 instance by the Elastic BeanStalk for deployment and monitoring of the application. Following are the task that are performed by an EBS HM (host manager).
  - Deploy the application into EBS – EC2 resources.
  - Aggregate Event and Metrics for retrieval via console /API/ Command Line Interface.
  - Generate Instance level events.
  - Monitor Application Level Log file for critical errors. (this feature is not available on OpsWorks).

- Monitor Application Server.
- Patching Instance component.
- Rotating Application log and publishing it directly to configure S3 bucket.

- **Elastic BeanStalk Custom Platform or Custom AMI** – If supported default pre-build platform does not supports the customer requirement customer can opt for using Elastic BeanStack Custom Platform feature where one can build their desired platform from scratch and use it within EBS for hosting custom application.
  In order to start with the building custom platform, one needs to starts with building AMI from a supported operation system, then create an Elastic Bean Stalk platform using Packer an open source tool for creating machine images for many platform including AMI which can be used within EC2 instances.

  When to use custom platform?
  - for hosting legacy application which are NOT supported by the Elastic BeanStalk platforms.
  - to reduce the startup time of the application, as Elastic BeanStalk needs to run multiple scripts to build the desired environment for deploying the application, ALL these configuration (scripts) can be move to the AMI instead of running it during the installation process they by reducing the startup time of the application.

- **Elastic BeanStalk – for docker container base applications:** A docker is a standardized unit of software development, containing everything needed to run including library binary files, system tools, code and runtime. The main benefit of using docker container is the ability to quickly and easily spin lightweight repeatable environments using minimal codding.
- One of the possible ways to onboard a custom application into elastic beanstalk is to create a docker image and use the docker image to build the Elastic Beanstalk application.
- Environment Variable define within the Elastic BeanStalk are passed directly to the container image.
- Elastic BeanStalk infrastructure automatically takes care of the capacity provisioning, load balancing, auto scaling and application health monitoring.
- If the docker container with the Elastic BeanStalk crashed / killed, Elastic BeanStalk restart the docker container automatically.
- Docker Platform configuration available within Elastic BeanStalk
  - **Single container Configuration**: It can be used to deploy Docker image describe in docker file or Dockerrun.aws.json (optional) definition and source code EC2 instances running in an Elastic Beanstalk environment.
  - **Multiple container Configuration**: Use Elastic Container Service (ECS) to coordinate deployment of multiple docker containers to Amazon ECS cluster in Elastic BeanStalk environment. Instance in the environment each run the same set of containers, which is define in dockerrun.aws.json file (in this case dockerrun.aws.json is mandatory).
  - **Pre-Configure docker platform configuration with supported language**: There are many pre-configure docker platform that can be used to run popular combination software stacks like java with glassfish etc. This option does not provide means to configure the software that application runs on. However, to customized the preconfigure docker platform to install additional software that the application needs, one can add a docker file to the application root folder.
- Elastic BeanStalk integration with other AWS Service
  - By default, Elastic BeanStalk creates s3 bucket in the same region where EBS is running, this bucket will be used by EBS for storing object required for proper operation of the application. The buckets are NOT encrypted, but the same time secure as it does not have public access to it.
  - Elastic BeanStalk can be integrated with Elastic File System (EFS), it can be mount to instance across availability zone, which can use it like local storage drive without changing the code.
  - Elastic BeanStalk can be integrated with RDS, EBS instance can refer the RDS using CNAME value so that in case of multi-AZ RDS instance once the primary DB instance fails and it get switch over the secondary instance without changing the endpoint.

NOTE: Don't use the EBS generated security group to all access to the RDS instance, as it will create a dependency on the RDS security group, which will prevent it from deleting when the EBS instance is terminated. (All other EBS resources will be deleted on termination, but NOT the security group). Solution to this problem is to create a new security group (outside EBS) allowing DB communication with the security group and attach it to both EBS EC2 instance and to the RDS instance, this will help in removing the dependency).

- Elastic BeanStalk can access dynamo DB (created within or outside the Elastic BeanStalk).

- Monitoring in ElasticBeanStalk – default ElasticBeanStalk is monitored by Cloud Watch. Elastic BeanStalk can also be customized to send custom metrics to CloudWatch. CloudWatch then can be used for monitoring OR creating alerts to implement decisions or automatic notification based on the define rules.
- Elastic BeanStalk application logs are automatically sent to CloudWatch, which can be configured to also forward to an S3 location or archived at S3 location. Each of the EC2 instance of the Elastic BeanStalk has a CloudWatch Log agent installed which helps in publishing metrics datapoints [custom datapoints] to CloudWatch to each configure Log Groups. [Each Log Group applies its own filter pattern to determine what log streams events to send CloudWatch as data points – each logstreams can be configured to have their own retention, monitoring, access control settings].
- Additionally, enhance monitoring can be enable for Elastic BeanStalk EC2 instance. Each of the AMI for Elastic BeanStalk comes with health agent installed on it which helps in supporting monitoring of the enhance health.
- Elastic BeanStalk can export log data from different log group to be exported to S3 location, a same S3 location be used to store logs from multiple different log group (source). In order to separate the log data, different prefix can be attached to each of the log groups. [*additionally, lifecycle policies can be created on the S3 bucket to archive the data into S3 Glacier Storage class.*]
- Logging for docker container within Elastic BeanStalk: One needs to configure application to write logs into a specific folder and then those logs can be configured to send to S3 bucket.
- CloudTrail can be enabled to Elastic BeanStalk API calls – every call made to Elastic BeanStalk can be made to record in cloudTrails logs which can then be send to S3 bucket.

Best-Fit to use case for Elastic BeanStalk	**When NOT to use Elastic BeanStalk**
○ Quickly Deploy application for prototyping or for testing. ○ To deploy application to AWS with no or minimal knowledge of AWS infrastructure. ○ Migrating custom web application to AWS infrastructure. • Create a docker file including all application dependencies. • Create a docker image from the docker file • Upload the docker image to public or private docker image repository. • Deploy the docker image using Elastic BeanStalk as single docker image or multiple docker image. ○ To be used in software development project by the software developers.	○ When team has expertise on build secure/scalable AWS infrastructure, and needs more control on the AWS infrastructure. ○ Team needs to recreate the AWS infrastructure quickly across different AWS regions.

CloudWatch

CloudWatch: Provide required visibility to the AWS Infrastructure.

- It provides system wide visibility into resource utilization, application performance and operational health.
- Monitor AWS resources and Applications in real time.
- CloudWatch can be used to collect and tract metrics – it can be AWS generated and custom metrics.

- CloudWatch metrics can be viewed from Amazon CloudWatch console, AWS CLI, CloudWatch APIs, Or using AWS SDKs.
- CloudWatch can collect data point securely from VPC resources using CloudWatch VPC interface endpoint.
- CloudWatch can be configure to send Alarms (SNS Notification) OR to take actions (like EC2 recovery/start/stop/autoscaling) based on the configure condition on collected metric data.

- **CloudWatch Concepts**
  - **Namespaces:** Namespace helps in isolating cloud watch metric data from mixing up. Each collected metric data has its own namespaces. One can also define custom namespaces for custom metrics. For AWS services AWS/<service-name> namespace is reserved.
  - **Metrics:** time-ordered set of data-points that are published by AWS services/resources to CloudWatch. For example, for EC2 instance CPU utilization data point sends for every 5 min intervals (basic monitoring) OR for EC2 instance CPU utilization data point sends for every 1 min intervals (detailed monitoring). [**CloudWatch are metrics are regional, however the CloudWatch dashboard can global.**]
  - **Timestamp:** each metrics data-point has to be had a specific time stamp attached to it. It can be 2 weeks in past and up to 2 hours in future. If no timestamp is attached to data-points, then the time it been received is set as its timestamp.

    Metrics cannot be deleted – if now new data point are published on metrics more than 15 months AWS automatically deletes the metrics.

    Data-Point Retention

    - Data-Point less than 60 seconds (high resolution data points) are available for 3 hours.
    - Data-Point equal to 60 seconds (1 minute) are available for 15 days.
    - Data-Points equal to 300 seconds (5 minute) are available for 63 days.
    - Data-Point equal to 3600 seconds (1 hour) are available for 455 days (15 months).
    - Data points older than 15 months automatically rollover by the new data-points.
    - Shorter Data points can be aggregated to longer data-points to keep it available for long duration.

    > After 3 hours all less than 60 sec data-point are aggregated to 1 min data points.

    > After 15 days all 1 min data points are aggregate to 5 min data points

    > After 63 days all 5 min data points are aggregate to 1 hr data points and retain for 15 months.
  - **Dimensions:** Dimension is an embedded property of the AWS::CloudWatch::Alarm type. Dimensions are name/value pairs that can be associated with a CloudWatch metric. You can specify a maximum of 10 dimensions for a given metric.
  - **CloudWatch Alarm:**

    - It can be created on a single metrics or on math expression based on CloudWatch metrics.
    - CloudWatch Alarm can be configure to perform action base on the configure threshold value over a time period (duration of time period where the alarm is evaluated).
    - Possible state of a CloudWatch alarm are – OK / ALARM / INSUFFICIENT DATA
    - For a CloudWatch ALRM to trigger three things are required

    a) Data-Point: length of time to evaluate the metrics expression to create individual data point for alarm.

    b) Evaluation Period: time period for which the data-points needs to be evaluated.

    c) Datapoints to Alarm: Number of data points that need to exceeded the threshold value to trigger the alarm.

    [CloudWatch Alarm and CloudWatch Event are two separate things – CloudWatch alarms can trigger EC2 instance recovery/start/stop/autoscaling action/send SNS notifications WHERE CloudWatch Events can trigger many other services then what alarm can].

- **CloudWatch Logs: CloudWatch logs can be used for monitor, store, and access log files from AWS and non-AWS sources. CloudWatch logs can be monitored, store, access logs from**
  - Amazon EC2 instances
  - Containers running on ECS (these required installing of the cloud watch agents).
  - AWS Cloud Trail
  - Route53 DNS Queries logs

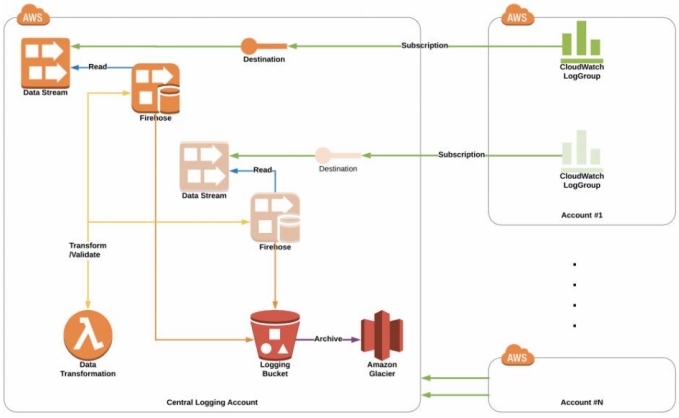
- RDS Aurora, MYSQL and MariaDB
- Amazon Neptune
- VPC Flow Logs
- Elastic BeanStalk for EC2 instance in the EB environment.
- API Gateway execution logging
- Lambda function logs and other sources
- Logs from on-premises instances (The instance needs to install cloud watch agents to sends the logs to cloud watch). **Note: Logging in the central log repository, help in query filtering and tracing the logs end-to-end.**

- **Log Event** – is the record of some activity recorded by the application or resources been monitored. It consists of two properties timestamp and raw log message.
- **Log Stream** – sequence of log events that share a same source. Empty log steams older than 2 months old, are automatically deleted. One can delete a log stream manually. Log streams need to belong to a Log Group.
- **Log Group** – Group of log streams that shares same retention, monitoring and access control settings.
- **Metric Filter** – metric filter can be used to extract information from the log event and transform them to a data point in a cloudWatch metrics. Metric filters are applied at the log group level and gets applies to all logs stream within that log group. CloudWatch logs send log metric to cloudWatch every minute.
- **Encryption of log data in cloudWatch logs**
  - **At Transit:** CloudWatch service encrypts the logs during transit using end-to-end encryption of data during transit.
  - **At Rest: CloudWatch** service can be configure to encrypt the log data at rest using server-side encryption, same can also be achieve using server-side-encryption using Amazon KMS – CMK (customer manage key) [*this can be done ONLY from CLI not from the AWS console*]. Encryption can be enabled at the Log Group level, while creating the group or later.
- **CloudWatch Log Insight**: CloudWatch Log Insight enables the interactive search and analysis of the log data in Amazon CloudWatch logs. It consists of few simple but powerful commands to search & perform analysis on the log data stored in the CloudWatch. CloudWatch Log Insight automatically discovers fields in AWS services such as EC2, Route53, AWS Lambda, AWS Cloud Trails, Amazon VPC OR any application/ custom logs that ingest into CloudWatch in JSON format. Up to 20 logs groups can be search a single request. **[CloudWorks insight works for logs that are logged after Nov 5th 2018 – logs sent before this data can't be search.]**
- Real time monitor of EC2 instances: CloudWatch logs can monitor application logs from the Amazon EC2 instance in real time.
- CloudWatch Logs and CloudTrail:
- Exporting CloudWatch Logs to S3: Depending upon the size of the data, exporting of CloudWatch Logs to S3 bucket can take significant time (up to 12 hour), in case the logs need to be process in real time then instead of exporting the data into S3 use subscription.
- Streaming cloudWatch log data to elastic search.

- **Unified CloudWatch Agent:** The newer version of the CloudWatch agent which can be installed on both Lunix and Windows base systems. It can collect metrics and logs from (EC2 or on-premises) systems. Its capable to send both rotating and non-rotating logs to cloudWatch which can be then viewedS on CloudWatch. [*New CloudWatch agent (unified cloudWatch agent) has better performance as compare to old one, and helps to capture advance metrics in addition to the logs for in-guest visibility.*]
- **CloudWatch - Real time processing of log data with subscription:** Use subscription to get access of real time feed of log event from CloudWatch logs to be deliver it to other services for custom processing, analysis, loading of logs through services like - Amazon Kinesis, Amazon Kinesis Data Firehose streams, AWS Lambda etc. [Can have subscription filter, to enable selective subscription of the logs.]
- **CloudWatch logs to be shared across AWS accounts for processing** [*cross account log data sharing*]. (Linked account (member account) can send the log to a designated member account for processing of the logs).
  NOTE: Currently ONLY kinesis data streams is allowed to collect the log from different accounts. In order to create cross account log data sharing the following steps needs to be performed.

- Create a log group to collect logs streams from different destination – log group and the destination should be in the same region.
- Create a CloudWatch log subscription filter in case one needs to send ONLY the matching parameters to the target Kinesis data stream for processing.
- CloudWatch log destination name, logical name that point to the target (Kinesis data stream).
- Target ARN: ARN of the destination, situated on another AWS account.
- Role ARN: IAM role that grant required permission to enqueue data to the destination Kinesis data stream.
- Access Policy

- Use case: For centrally processing of log across multiple account.



- CloudWatch Event: CloudWatch event delivers near real time streams of system events that describes changes in AWS resources. It can be used for:
  - Sending messages to response to the environment.
  - Activate functions
  - Making changes
  - Capturing state information
- CloudWatch **Event**:
  - Indicate change in the AWS environment
  - Custom application that can generate CloudWatch events
  - Schedule CloudWatch event to perform specific actions
- CloudWatch **Rules**:
  - Rule match the incoming CloudWatch event, perform action on them and forward it to the target. Rule can also be used to customized the events, before forwarding it to the target – it can override the JSON content of event with constants OR filter-out certain parts of the event.
- CloudWatch **Target**:
  - CloudWatch Target are the components that receives CloudWatch events in json format and process then. Many AWS services can be configured as AWS CloudWatch targets.

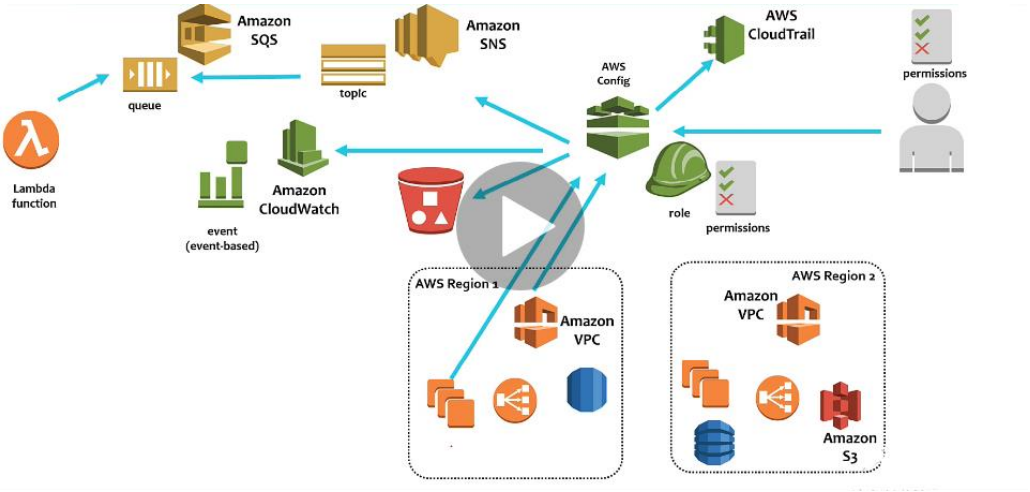
Note: *CloudWatch event can be shared across accounts, however the both the sender account and the receiving account needs to be in the same region. Sender account will be charge for sending CloudWatch event to the receiving accounts.*

- **CloudWatch Synthetics** – A fully managed services that enables canaries creation to monitor endpoints and APIs in target environment from outside in. Canaries are configurable scripts that follows the same route and perform the same actions as a customer – which enables outside-in view of the customers experience and also the service availability from customer view point. It can also check the availability and the latency of the endpoint and can store load time data and screenshots of the UI. It can be schedule to run once or on a regular schedule interval. CloudWatch alarms can be configure to notify if *canaries* fail.
- **CloudWatch ServiceLenses** – Enhance the observability of a service/application by integrating, traces, metrics, alarms and logs in one single place. AWS ServiceLense integrates CloudWatch with X-Rays, to provide end-to-end view of the application to help more effectively to pin-point the bottle neck of the service/application.

Service Map displays the service endpoints and resources as nodes and highlight the traffics latency and error for each node and its connections. One can be drill down on the nodes to see the detailed insights of the correlated metrics, logs, and traces associated with that part of the service.

AWS Config

- AWS Config: How the resources are connected to other resources & it also helps in know the past configuration (configuration and change history). It's possible to create a private link for AWS config to lookup on premises resources.



  - Benefit of AWS Config
    - Evaluate AWS resources for desired settings
    - Get a snapshot of the current configuration of the supported resources that are associated with the AWS account.
    - Retrieve configuration of one or more resources that exist in the account.
    - Retrieve historical configuration of one or more resources that exist in the account.
    - Received configuration when a resource is created/modified/deleted.
    - View relationship between resources.
    - AWS Config can help in achieving administrator, auditing, and compliance
      - Achieving governance over the configuration status of your involved resources.
      - Get notified when a change/delete/modification of a resources happens.
      - Evaluate auditing compliance of your AWS resources.
      - Perform auditing using the historical configuration data from AWS config.
    - Managing and Troubleshooting Configuration Changes
      - Using AWS config, related resource can be identified – minimizing the risk of cascading failure due to change in one resource.
      - Using historical configuration of the resources provided by AWS Config to troubleshoot issues and access the last know working configuration and the delta between the current configuration and the working configuration.
    - Security Analysis
      - To expose the potential security exposer historical configuration can be used.
      - Trace the history of IAM permission granted to user or group.
      - Evaluation the configuration security groups when troubleshooting connectivity or packet dropping issues.
  - AWS Config – Concepts
    - **Configuration Items** – a point-in-time view of the various attribute of supported AWS resources that exist in the account. It includes – metadata, attributes relationship, current configuration, and related events. AWS Config create a configuration item whenever it detects a change to a resources type that it is resource.

- **Configuration History** – is a collection of configuration items for a given resources over given period of time. It helps in finding answers when the resource was created and when it was modified over the due course of time. It automatically delivers to S3 bucket, which can be retain for minimum of 30 days and maximum for 7 years
- **Configuration Recorder** – When create its stores the configuration of the supported resources in an account as configuration items. By default, a configuration recorder records all the configurations of the supported resources of its region when AWS Config is running. (this can be change).
- **Configuration Snapshot** – A collection of the configuration items for the supported resources that exist in an account. It's useful for validating configuration. Configuration Snapshot can be delivered to an S3 bucket that you specify.
- **Configuration Stream** – A configuration stream is an automatically updated list of all configuration items of the resource that AWS Config is recording. Every time a resource is created/deleted/modified AWS config create a configuration item and adds to the configuration stream. It works using AWS SNS topic. The configuration stream is helpful to monitor configuration changes in real-time as they occur. SNS topic can be used to notify the change in near-real time.
- **Resource Relationship** – AWS config discovers AWS resources in your account and then create a map of relationship between AWS resources.

- **AWS Config – Delivering Configuration Items**: AWS Config can deliver the configuration items to following channels:
  - **TO Amazon S3 bucket**: AWS Config can track changes in the configuration of the supported AWS resources and sends updated configuration details to a specified Amazon S3 bucket. For each resource type that AWS config records in sends a configuration history file in every 6 hours. Each file consists of configuration details that has change is last 6 hours – if no configuration change AWS config does not manage the S3 bucket lifecycle. One need to create their own lifecycle policy as per their convenience.
  - **TO Amazon SNS topic**: AWS config can be configured to deliver configuration item into SNS topic. It's advisable to create an SQS to the SNS topic and process the delivered configuration items programmatically. Under following conditions AWS Config will send notification to SNS topic
    - On change of the configuration item for an AWS resources
    - On successful delivery of a configuration history of the account.
    - On completion of the configuration snapshot of the recoded resources of the account.
    - On execution of the compliant state of the AWS resources of the account.
    - On starting of the evaluation rule for resources of the AWS resources of the account.
    - On failure of the AWS config to send notification.
    - On identification of a non-compliant flag for any AWS resources of the account.
- AWS Config Rule – Evaluation Resources with AWS Config Rules
  - AWS Config rule represent a desired configuration, if the resource configuration changes from the desired configuration then AWS Config flags the resources and mark the rule as non-compliant. The AWS Config rules can be configured to run at pre-define time or when the configuration changes. [There are predefine managed rules for AWS Config and also custom rules can be created.] AWS Config rule are associate with lambda functions which contains the completing it evaluation.
- Permission for AWS Config: AWS Config needs IAM roles to access/evaluate configuration of different AWS resources.
- AWS Config is allowed to aggregated configuration and compliance data from multiple account & different region or organization into a single account – this helps in maintaining a central IT administrator to monitor compliance for multiple account withing an enterprise.

- **Source Account** – is the AWS account for which configuration and compliance data needs to be collected – it can be a single account or multiple organization within a single account. AND the region where its hosted is known as source region.
- **The Aggregator** – is a new resource type within AWS Config that collects AWS Config collection and compliance data from the multiple *Source Account* and region AND region where aggregator is created is call Aggregator region. Permission the Source Account grants to Aggregator account for collecting configuration and compliance data is called as "Authorization", this is NOT REQURIED if source account and aggregator account are part of the same organization.

- AWS Config allows querying of the current configuration state of AWS resources based on configuration properties. It supports Ad-hoc, properties-based queries against current AWS resource state metadata across all supported AWS Config resources. AWS Config supports strong query feature provides a single query endpoint and power query language for obtaining current resource state metadata – this query can be used for:
  - **Cost Optimization**: For example, identifying list of EBS volume that are NOT attached to any EC2 instance.
  - **Security and Operational Intelligence**: For example, retrieving a list of all AWS resources within an account / enterprise that have a specific configuration property enabled/ disabled.
  - **Inventory Management**: for example, retrieving a list of all EC2 instance of a specific size.
- AWS Config monitoring
  - Using SQS queue: Send AWS Config SNS notification to one OR more SQS queue, then programmatically process the message – looking for non-compliant data, any undesired changes, maintain an up-to-date list of AWS resource inventory etc.
  - Using AWS CloudTrail: list out all the action performed on AWS Config by users, role, o by any AWS services.
  - Using AWS CloudWatch Event: Using CloudWatch event to detect and react on any change in the status of AWS Config events.

AWS Service Catalogue

AWS Service Catalogue: Key concepts

- **Product** – A product is a service OR an application for the end-user. A Product can comprise of one-or-more AWS resources such as EC2 instance, Storage Volume, DB instance, monitoring configuration OR package AWS marketplace product or network components. It can be a single EC2 instance or a fully configured web-application or anything in between. When a user launches a product that has an IAM role attached to it, AWS Service Catalogue uses that IAM role to launch the products (AWS Resources/Application) using AWS CloudFormation template. By assigning IAM role to the products, one can avoid giving access to user to launch required AWS resources.
- **Catalogue** – A catalogue is collection of products that the administrator creates, add to the portfolios and provided updates using AWS Service Catalogue. Administrator, creates catalogue of products by importing CloudFormation templates and also defines who can used those products and how they can use them.
- **Portfolio** – Administrator distribute products to end-user in Portfolio. Administrator can create a customized portfolio for each type of users within the organization and selectively grant access to the users to access those portfolios. Granting access to user to access a portfolio, allows user to browse and launch product from the portfolio.
- **Publishing Portfolio** – Add products to the portfolios, add IAM users, roles, groups to access the added products.
- **AWS Catalogue Provision Stack** – The AWS CloudFormation stack that gets provision when an AWS Catalogue get executed is called "*AWS Catalogue Provision Stack*".
- **AWS Catalogue Version** – AWS Catalogue allows to manage multiple versions of the same product in the Catalogue. When a new product version is added to the Catalogue its available to all the users to select from. Use can select the new version when creating a product from the catalogue OR can chose to update

their running stalk to the newer version. **[AWS Catalogue DOES NOT automatically upgrade the running stalk to the new version – users need to act on it to upgrade the version.]**

- **AWS Service Catalogue Scope and VPC Access** – Products and Portfolios are regional constructs; they are ONLY available on the region in which they are created – one can restrict in which region the data will be stored. It's possible for AWS Catalogue Provision Stalk resources to privately access VPC resources using VPC endpoints.
- **AWS Service Catalogue constrains** – rules that limit the users from selecting parameter values from a specific set of values while launching a product from their service catalogue. It's a means to apply cost control and governance. Constrains can be applied at each product level OR at the portfolio level. When there are multiple constrains at the product level and at the portfolios level – one which is more resistive will be applied. Constrains are applied while launching a new product OR while updating a running stalk with the newer version. Following are the different types of AWS Constrains
- **Launch Constrains:** IAM Role can be attached to a Product in a portfolio. This role is use for provisioning product at launch, there by user permission can be resisted without impacting the user's ability to provision product from their portfolio.
- **Notification Constrains:** Notification constrain enable users to get notification about the stack event using an Amazon SNS topic.
- **Template Constrains:** Template parameter rules are called as *template constrains* because this constrain are applied to the AWS CloudFormation template for selecting the parameter values while launching a product. One can reuse the same CloudFormation template and apply different template restriction per product basis OR per-portfolio basis.

Sharing of the Portfolios between AWS accounts. It's possible to share portfolios between AWS two accounts, however the control remains with the account who has shared the portfolios with

- The owner account (the account that shares the portfolios) can un-share the portfolio at any time.
- The owner account can ONLY make changes to the portfolios like – adding new product(s), updating existing products.
- Any product or stack that are currently in use will continue to RUN till stack owner decides to terminates those products/stacks. [Once the product/stalk is provisioned – portfolio owner account cannot delete them, ONLY the stack owner (account) which has provisioned the product can terminate those products]
- On top of owner account constrains – provision account can add more constrains to it.

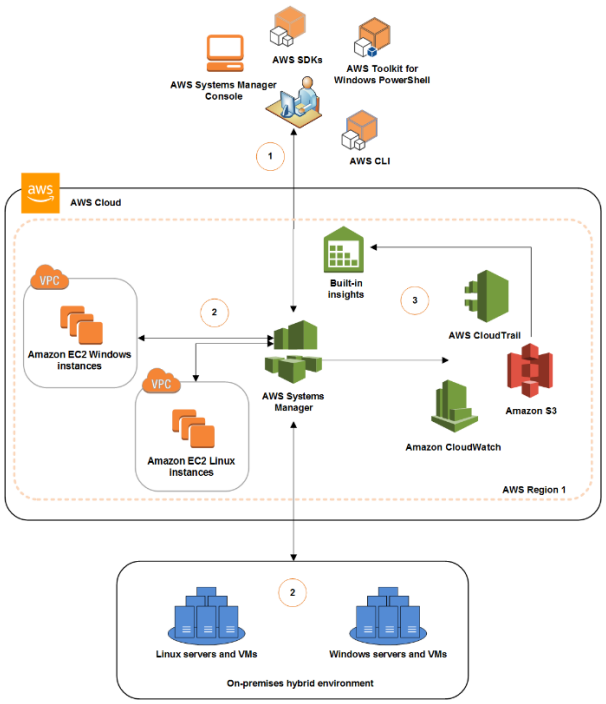
AWS System Manager (AWS SSM)

AWS System Manager (aka AWS Simple System Manager - AWS SSM): These are the collection of capabilities to configure and manage Amazon EC2 instances, on-premises server, virtual machines, and other AWS resources at scale.

- AWS SSM includes a unified interface that allows to easily centralized operational data and automate task across an AWS client resources across region.
- AWS SSM helps in effectively detect and resolve operational problem in AWS infrastructure – like finding out instance for which anti-virus are not running/not updated, out-of-sync configuration for instances etc.
- AWS SSM gives a complete view of the client's infrastructure performance, and configuration and simplified resources and application management.
- AWS SSM works with – Windows, Linux , RaspBian instance and virtual machine.
- AWS SSM can manage remotely and securely – EC2 instances, on premises servers, virtual machines in cloud or on-premises. IT can also mange virtual machine running on other cloud providers.
- AWS SSM can remotely and securely manage hybrid workload using same tools and scripts.
- AWS SSM can be used to use IAM for centralized access control for actions that can be performed on the servers and VMs.
    - AWS SSM can be used to collect auditing information in single location by using AWS CloudTrail.

- AWS SSM can be configured to use CloudWatch events and SNS notifications for centrally monitoring to send notifications about service execution success & failure.

- How AWS SSM Works?
  The below diagram shows how AWS SSM works in details – it has three distinct phase **Configuration**, **Verification**, and **Reporting**.
  Configuration: use AWS SSM console, SDK, AWS CLI, or AWS Toolkit for Windows PowerShell to configure, schedule, automate, execute SSM actions on AWS or On-premises resources. Install SSM agents and configure permissions.
  Validation: AWS SSM verifies the configuration including permissions and send request to SSM agents running on AWS instance or in servers in hybrid environments. SSM agent perform the specific configuration changes.
  Reporting: SSM Agent reports the status of the configuration changes and actions to AWS SSM. AWS SSM sends (forward) the status to the user and various AWS server if configured.



For connecting SSM agent to SSM service hosted on AWS infrastructure.

- For instance, in public subnet, SSM agent can connect to the SSM service directly OR using AWS SSM VPC endpoints.
- For instance, in private subnets one need to configure NAT gateway to route the SSM agent request to SSM Service through internet OR through SSM VPC endpoints.
- NOTE: VPC endpoints are more preferred approach.

Required permission for AWS SSM

- IAM User permission – Needed for those users without administrator access.
- IAM Instance Profile – Needed for the EC2 instance that will be managed by SSM. If the instance profile change, till the change credentials are reflected SSM agents will NOT work. SO, it's advisable to restart the SSM agent when instance profile change to reflect the credentials change immediately.
- IAM Service Role – Needed for on-premises servers/VMs

SSM Capabilities

- **Resource Group** makes it easy to manage and automate task on large number of resources at one time. Resource group is a collection of AWS resources that are all in the same AWS Region and that matches criteria provided in the query. Queries can be built in the resource group

console OR pass as arguments to Resource Group command in the AWS CLI. Resource group can be used to create custom console that organizes and consolidates information based on criteria that are specifies tags.

- **Insights**
  - **Built-In insights:** Built-in Insight shows detailed information about a resource in the respective AWS Resources Groups such as
    - CloudTrail Logs
    - Result of evaluation against Config Rules
    - AWS Trusted Advisor reports

    Single, or multiple resource groups can be selected at a same time.
  - **AWS CloudWatch Dashboards**: Helps in viewing data of the manage instances.
  - **Inventory Management:** It's an automated process of collecting software inventory from manage instances – it can be used to gather metadata about an application, files, component, patches, and many more on managed instance. If there are multiple Account spanning across multiple region – then Inventory manager need to use Resource data-sync to send all the inventory related data to a single S3 bucket from were data can be queried using amazon Athena. The data can send further to AWS Quick Sight or third-party tool to visualized the data.

    Inventory Manager store system data for 30 days – once the data is sent to s3 bucket one can configure required Lifecycle policy to persist the data in a cost optimized way for a longer duration of time.
  - Configuration Compliance: System Configuration Compliance can scan the fleet of instances of managed instance for patching compliance and configuration inconsistency. One can collect data from multiple AWS Account and AWS region and then drill down on the data of those instances which are found non-compliant.

    Other benefit of using System Configuration Compliance
    - View compliance history and change tracking of Patch Manager patching data and State Manager association by AWS Config.
    - Remediate issues by using system manger run command, state manager or Amazon CloudWatch event.
    - Port data to Amazon Athena and Amazon Quick Sight to generate fleet wide reports.
- **Actions**
  - Automation: SSM can perform automated task on mange instance at scales. SSM simplifies the common maintenance and deployment tasks on managed instance like
    - Build automation workflow to configure and manage instances and AWS resources by using custom workflows OR using predefine workflows maintained by AWS.
    - Received notification of the automation workflow tasks using AWS CloudWatch Events.
    - Monitor Automation process and execution details by using the Amazon EC2 instance OR System Manage console.

    Some of the example of task that can be automated by SSM Automation capabilities are:
    - Create or update Amazon Machine Images.
    - Apply Driver or Agent updates.
    - Reset passwords on windows instances.
    - Reset SSH key on Linux instances.
    - Apply OS patching OR application updates.

    Automation Use case using SSM Automation capabilities
    - Request that one or more IAM user approves the instance stop action.

- Automatically stop and start instance on a schedule by using CloudWatch events or by using Maintenance Window task.
- Update resources that are deployed by CloudFormation templates – additionally the workflow can be configure to have one or more IAM users to approve the task before updating the resources.
- Tag large/small groups of EC2 instances using Amazon EC2 tags that helps to roll out changes according to the limit defined OR target an AWS resources group that includes multiple instances.
- Create Golden AMI from the source AMIs
- Included or exclude specific package from getting deployed.
- Recover Impaired instances.

- **RUN Command:** RUN Command can be used to remotely and securely manage the configuration of the manage instances at scale – it can be used to run administrative task OR ad hoc configuration at scale. RUN Command can be executed from AWS Console, AWS CLI, AWS Toolkit for windows powershell or using AWS SDKs. RUN Command can be configure to send its logs to CloudWatch events, if one desired to maintain the complete log then RUN Command can be configure to send its logs to S3 bucket. RUN Command can be configure to use CloudWatch event – to notify command execution status OR to it can be specify as target for CloudWatch event to execute specific documented commands on receiving a CloudWatch event.
  RUN Command can be used for the following use cases to run on-demand changes at scale.
  - Update Application
  - Running Linux shell scripts OR windows PowerShell
    - For Linux instance use RUN Command with AWS-RunShellScript document
    - For Windows instance, use RUN Command with AWS-RunPowerScript document
  - Install OR bootstrap applications
  - Build a deployment Pipeline
  - Capture the log files when the instance is terminated from the autoscaling group.
  - Join instance to a windows domain.

Troubleshooting RUN Command:

- *Can't find the desired instances, to run RUN Command.*
  Possibilities
  - SSM agent on the instance are NOT running/installed
  - EC2 instance are attached with IAM role that allows it communicate with the System Manager APIs.
  - Verify the user account has IAM trust policies that enable account to communicate with the System Manager API.
  - Ensure that SSM document supports the instance type. (Some of the SSM documents are specific to windows or Linux)

- **Session Manager:** AWS SSM State manger is a secure and scalable configuration management service that automate the process of keeping the manage instance in a desired state as define by the administrator.SSM Session Manger helps in complying with the stick enterprise policies and security practices with fully auditable logs while providing an easy means to the end-users with simple one-click access.

  SSM Session Manager – How it works?
  If administrator need to connect to an (private) EC2 instance – instead of creating and configuring baston host for Ad-hoc login, they can quickly and securely connect

to the EC instance using SSM Session Manager console or through AWS SSM Session Manger CLI behind the scene – AWS SSM Session Manager on receiving the first command to start a new session performs

1. Authenticate the login ID
2. Verify the action granted by the IAM policies
3. Check the configuration settings (such as verify the allowed limit of the sessions etc.)
4. Send a request to SSM Agent deployed on the EC2 instance to open a 2-way communication connection for the use to interact with the EC2 instance.

NOTE: As SSM Session manager used SSM Agent to open 2-way communication connection, it does NOT require inbound ports to be open on the managed instances.

AWS SSM Session Manager can be integrated with

- *AWS CloudTrail*
- *Amazon S3*
- *Amazon CloudWatch Logs*
- *Amazon CloudWatch events and Amazon SNS.*

- **Distributor:** SSM Distributor capability can be used for deploying/publishing software packages to SSM Managed instances. Package can be custom packages OR AWS provided agent software packages.

  Package can be delivered to specified mange instance using – RUN Command/State Manger based on - AWS Account Number/instanceID/tags/AWS Region

  Package deployment can be automated using State Manger to schedule package for automatic deployment on target instances when those instances are first launched.

- **Patch Manager:** SSM Patch Manage can be used for automating the process of scanning managed instances for missing patches and to apply the patches. It can be used for scheduling recurring patching activities using System Manager Maintenance Window
  - **For Windows based instance:** ONLY security patches can be delivered using SSM patch manager.
  - **For Linux based instance:** Along with security patch, it can also be used to install/update package on the managed instance.
- **Maintenance Window:** Use to setup recurring schedule for managed instances to execute administrative tasks like installing packages and updates without interrupting business critical operations.
- **State Manager:** SSM State manage is a fully managed service by AWS, that allows maintaining fleet of EC2 instance through an interactive one-click browser base shell OR through AWS CLI – it allows accessing of the EC2 instance without opening the inbound ports OR manage baston host OR maintain SSH keys.

  AWS SSM State Manager can be used to do
  - *Instance are patched with specific software update.*
  - *Bootstrap instances with specific software at the start-up*
  - *Download OR update agents at define schedules*
  - *Configure Network settings*
  - *Join instance in windows domain*
  - *Patch instance with software update throughout their lifecycle*

- *Run scripts on windows / Linux instances throughout their lifecycle [if one desired to run script in recurring basis thought the lifecycle of the instance then – State Manager is the right approach NOT RUN Command].*
- *AWS SSM - State Manager can be integrated with the AWS CloudTrail to provide all the logs of all execution for auditing purpose.*
- *AWS SSM – State Manage can be configure to save detailed command output into S3 bucket for viewing later*

SSM State Manager Association with manage instances

- *State manage uses SSM document to create an association*
- *Configuration defines the state that needs to be maintained on the instance. Example of an SSM State Manage State – antivirus is installed & running on the instance sand specific ports are close.*
- *State Manage association can also specify a schedule for when the configuration need to be (re)applied (Example: like every Thursday 3 PM)*
- *State Manage association can specify action to be taken when applying a configuration. Example – if the antivirus is NOT installed then state manage installed the antivirus – if the antivirus is installed but not in running state then it can turn-on the antivirus.*

- Share Resources
  - **Managed Instance:** Instances that are managed by the AWS SSM – there are two types of managed instance – Standard and Advance Instances. Advance Instance enable AWS SSM to connect to hybrid machine by using AWS SSM Session Manger – Session manager also provides interactive (2-way-communication) shell access to the instances.
  - **Activation**:
  - **System Manager (SSM) Document**:
    - These are list if actions that are performed by SSM on managed instances.
    - It includes steps and parameters that one can specify for SSM to act upon.
    - SSM documents are written in JSON or in YML.
    - There are two types of SSM Documents
      - Command Documents: These documents are used by SSM and RUN Command.
      - Automation Document: These documents are used by SSM Automation.

    SSM offers large sets of pre-configured documents that one can use by specifying parameters at runtime.
  - **Parameter Store**: Parameter store secure credentials, passwords, security strings, software licenses that one didn't want to add in the code (document) level, which can be shared securely with SSM components. There are two types of parameter store – Standard and Advance Parameter store. Standard parameter can store up to 4K and can be use free of cost. Advance Parameter can store up to 8K of data and come with additional cost. Advance parameter is chargeable based on the number of advance parameters store each month AND per API interaction.

    AWS SSM Parameter Store provides a serverless, secure, hierarchical storage for configuration data management and secret management. Parameter store can be used for manage instance as well as for on premises instances. Parameter stored in Parameter store can be referenced using UNIQUE string value – the parameter store values can be stored in plain-text as well as encrypted text using KMS encryption (in case the parameter values is encrypted – instance accessing those values should have access to the encryption key.

    Benefits of using Parameter store

- A secure, scalable, AWS managed secret management service.
- It stores configuration and secure string in hierarchical and trackable version with control and audit access at granular level.
- It can refer Secret Manager.
- Individual parameters can be tag and its access can be secure at different levels – operational, parameter, Amazon EC2 tags, or at path level.
- It can be integrated with other SSM capabilities and AWS services to retrieve secret and configuration data from a central store.
- Parameter store is supported by – Amazon EC2, Amazon ECS, AWS Lambda, AWS CloudFormation, AWS CodeBuild , AWS CodeDeploy, AWS KMS, AWS SNS, AWS CloudWatch, AWS CloudTrail.

Parameters Store can be referred from

- SSM Scripts, commands, SSM documents, configuration and automation workflow.
- SSM RUN Command, State Manager and Automation.
- Form AWS ECS and AWS Lambda.
- AWS CloudWatch Agent can store agent configuration in Parameter Store.

AWS Parameter store secret using secret manager, can be leverage to store & organized secret data like credentials, passwords, security strings, software licenses etc. AWS Service access Parameter Store – Parameter Store can refer Secret Manager for accessing secret. (Parameter store acts like a proxy for secret manager, and it doesn't retain secret data or metadata.)

- **AWS SSM Inventory and other AWS Service**s: AWS SSM inventory provides a snapshot of the current inventory which can be used to manage software policies and improve the security of the entire fleet. AWS SSM Inventory management and migration capability can be enhance using following AWS services
  - **AWS Config:** It can provide historical record of changes to the inventory, along with the ability to create rules to generate notification when a configuration item is changed.
  - **AWS Application Discovery Services**: It is designed to collect inventory on OS type , application inventory, processes , connections and server performance metrics from on premises VMs for an successful migration to AWS infrastructure.
  - **AWS System Manager Resource Data Sync** can be used to send data from different AWS Service Manger gathering information from all managed instances into a single S3 bucket. Additional services like KMS can be used to encrypt the data store in the S3 bucket , which then can be used by AWS Services like AWS Athena and AWS Quick Sight to query and analyze the aggregated data.
- **AWS SSM Pricing**
  - **Parameter Store Charges**: For using advance Parameter storages there is additional charges.
  - **Distributer Charges:** For AWS Software Packages there is NO CHARGES (FREE) for distribution and update. For Custom packages its CHARGEABLE.
  - **Mange instance Charges:** For standard instance up to 1000 on premises instances per account / per region are FREE. For more than 1000 instances – one need to covert all the instances in to "Advance" instance which will be CHARGEABLE
  - **Automation Charges**: Based on its usages, number of steps & types of steps its chargeable.

Section 6: Designing Network on AWS for Complex Organization and Hybrid Architecture

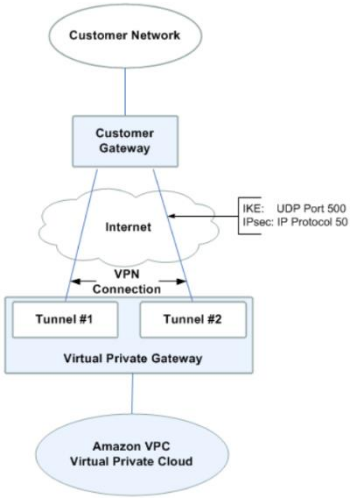
Network refresher

- For a VPC CIDR once selected can't be edited once created – possible CIDR block size are /28 (2^{32-28} - 5 = 11 IP addresses) to /16 (2^{32-16} - 5=65531 IP addresses). However, one can add a non-overlapping secondary CIDR block to the existing VPC to expand the size of the VPC.
- VPC Wizard options
    - VPC with single public subnet
    - VPC with public and private subnet
        - Custom route table will be associate with the public subnet with a route entry with IGW
        - Default route table will be associate with the private subnet with a route to NAT Gateway.
    - VPC with public and private subnet and Hardware VPN access
        - Custom route table will be associate with the public subnet with a route entry with IGW
        - Default route table will be associate with the private subnet with a route virtual gateway, which will have a connection to VPN or a Direct Connect (Direct Connection) to HQ.
        - Route propagation will be set to the default route table – so that it need not to be updated manually for the know route from the virtual gate to the default route table.
    - VPC with Private Subnet ONLY and Hardware VPN access
        - Create a Virtual Private Gateway with NO Elastic IP configured
        - Create a default route table with the route to virtual private gateway.
        - It will create a VPN Connection (user can configure Direct Connection if needed).

Direct Connect

link : https://docs.aws.amazon.com/vpc/latest/adminguide/Introduction.html

VPN Based connection

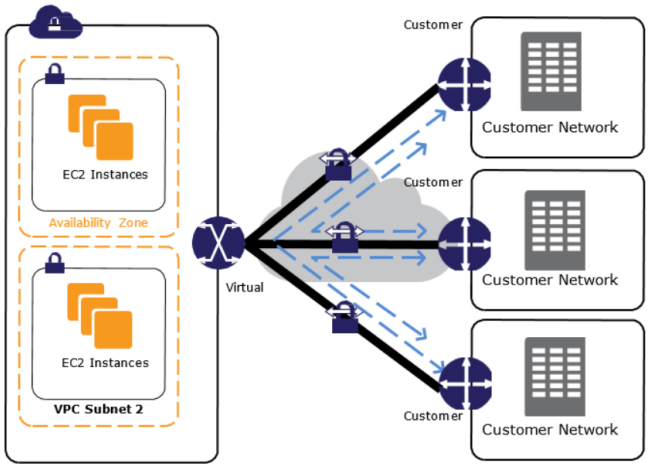


- Mostly desirable as it cost effective and takes less time to provision as no dedicated line needs to be created.
- On customer side Customer Gateway needs to be configured – on VPC side Virtual Private Gateway (VGW) needs to be configured.
- Customer Gateway can be physical devise or software application with internet routable IP address.
- The connection between Customer Gateway and Virtual Private Gateway (VGW) is established over IPsec Tunnels. Two connection are configured for each VPN Connection, to increase availability. Multiple customer gateway can be connected to a to single VGW
- NAT gateway can be reached through the VPN Connection. Connection initiated from the customer gateway cannot be route through NAT gateway.

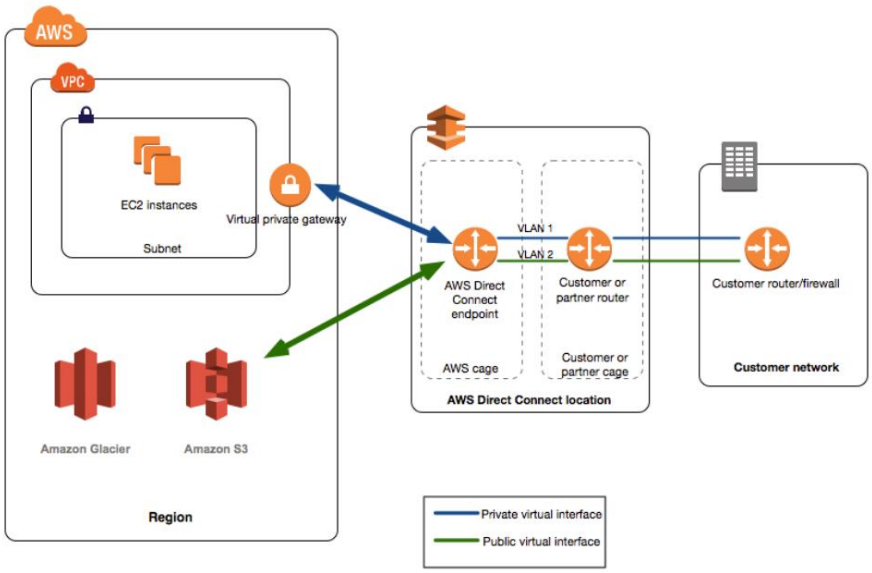
- *How route table connected to the Virtual Gateway knows about the subnets behind the customer gateway?*

  In order to connect VPN only subnets to the subnets from the customer datacentre one needs to updates the vpn-only subnet route tables to point it to the customer subnets and in customer side route table need to updated with the VPN only subnets information. This can be done by Manually updating the Route Tables Or by enable dynamic route propagation.
- ***AWS VPN CloudHub**:* AWS VPN CloudHub operates as simple hub-spoke model that can be use with or without VPC. *It's a cost-effective option to connecting remote sites with AWS network.* It also be used as a backup connection to connect remote sites. AWS VPN CloudHub leverage AWS Virtual Private Gateway with multiple gateways with each using unique BGP autonomous system number (ASN). Customer gateways advertise appropriate routes over their VPN Connections, these routes are received and re-

advertised to each BGP peer connected to the AWS Virtual Private Gateway. This allows connected remote sites to communicate with each other through VPN connection and also send & received data from the VPC. This option can be combined with AWS Direct Connect or other VPN options.
NOTE: The remote network prefixes for each spoke must have unique ASNs, and the sites must not have overlapping IP ranges.



- ***CloudHub Pricing:*** *For the VPN uses hourly rate are applicable – and the data charges are applicable for the data send from the AWS network to the VPN connection.*
- For IP prefix to send and received traffic through VPN Connection –IP behind the customer gateway need to advertise the IPs by the customer gateway & the VPC IPs that allow to send/received traffic need to be advertised at the Virtual Private Gateway. ONLY IP prefix that are known to the virtual private gateway are allowed to communicate. VGW learn about the IP prefixes through or BGP routing.
- Direct Connect (AWS DX): AWS Direct Connect links internal network to a Direct Connect location over a standard Ethernet fibre-optic cable. One end of the cable is connected to the customer route and the other end of the cable is connect to the AWS Direct Connect Route. With this connection one can create virtual interface for Public AWS Services like S3 and bypass the internet while communicating with these services – make it fast and more secure. An AWS Direct Connect location provides access to AWS in the region with which its associated. The same connection can be used to connect to other AWS services in another public region.



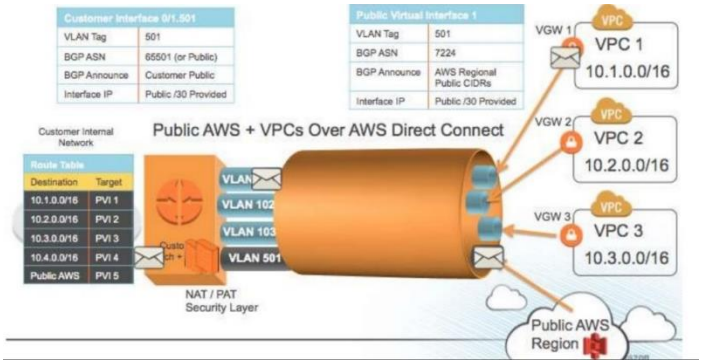
- Benefits of Direct Connect
    - Low latency and jitter (using direct connect one can bypass internet to connect to AWS services or VPCs).
    - Consistent performance and dedicated bandwidth.
    - Direct Connect enable customer to lower their Internet service provided bandwidth commitment.

- AWS Charges lower data rate on Direct Connect then VPN connections.
- Single Direct Connect Connection to build multi-region services - AWS Direct Connect to a public Regions or AWS GovCloud (US) can access public services in any other public Region, as wells as can access VPC in any public region.

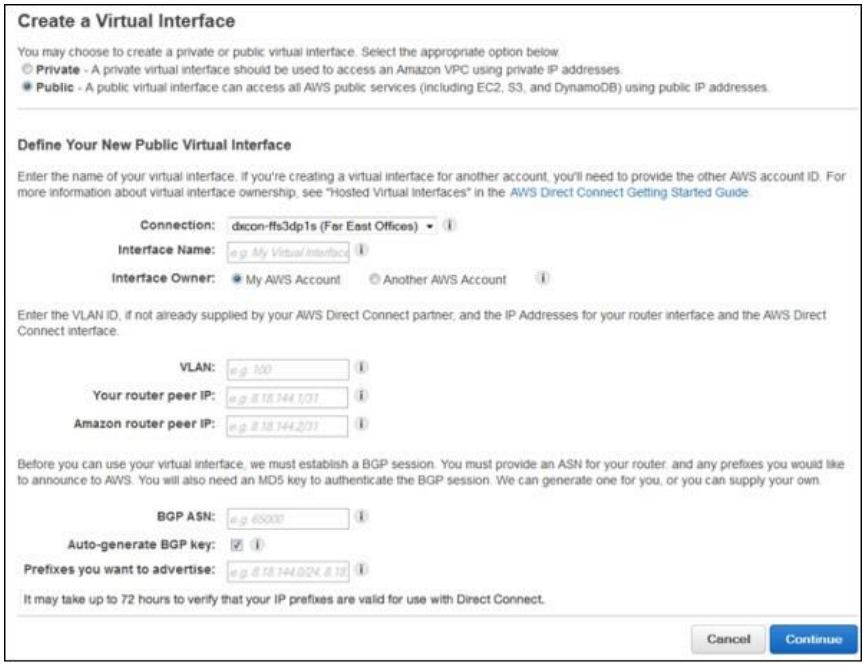
- Requirement for establishing Direct Connect
  - Connections to AWS Direct Connect requires single mode fibre, 1000BASE-LX (1310nm) for 1 gigabit Ethernet, or 10GBASE-LR (1310nm) for 10 gigabit Ethernet. **Auto Negotiation for the port must be disabled.** Support for 802.1Q VLANs across these connections should be available.
  - Network must support Border Gateway Protocol (BGP) and BGP MD5 authentication. Optionally, we may configure Bidirectional Forwarding Detection (BFD). Border Gateway Routing (BGP) Connection must be used to connect between the two ends of the Direct Connect.
  - Customer Routing must be capable of 802.1Q and BGP Routing (in case of VPN connection Static routing is allowed along with BGP routing – in case of DirectX ONLY BGP is allowed).
  - Direct Connection can be used to established Private Virtual Interface (VIF) to connect to VPC in the same region (via the VPC's VGW).
  - One can establish multiple private VIF to connect to multiple VPC within the same region.
  - One can also establish multiple public VIF to connect to any public AWS Service in any Region.
  - Direct Connect support both IPV4 and IPV6, but IPV6 needs public VIF.
  - All network traffic remains on the AWS global network backbone, whether one access public AWS service OR any VPC in another region. NO TRAFFIC LEAVES AWS NETWORK.
  - Any data transfer out of the remote region is billed at the remote region data transfer rate.

- 802.1q Link and Virtual Interface: *802.1q is industry standard Layer 2 protocol to segregate and isolate chunks of packets into separate VLAN channels. Each of the VLAN is configure to a separate Virtual Interface (VIF). This allows users to use the same connection to access public resources such as objects stored in Amazon S3 using public IP address space, and private resources such as Amazon EC2 instances running within an Amazon Virtual Private Cloud (VPC) using private IP space, while maintaining network separation between the public and private environments.*

  Up to 4019 channels (VLAN) can be possible in a standard connection. Virtual interfaces (VIF) mapped to each VLAN can be reconfigured at any time to meet your changing needs.



- **Static Vs Dynamic Routing**: Routing is needed to correctly forward the traffic correctly. Static Routing - For simple network (with few routers) one can manually define the routes that is called static routing like in the case of AWS Route table. As the network grows (large number of routers gets added) maintaining static routing can be cumbersome, to overcome such administrative challenges Dynamic Routing (for example enabling route propagation between customer gateway router and AWS Virtual Gateway (VGW) when connecting to AWS network using VPN or Direct Connect.
- Direct Connect configuration page:



- BGP (Border Gateway Protocol):
  - Currently, BGP is the main internet routing protocol.
  - It can carry a large number of routes with granular route control.it has many route/path attributes to control routing.
  - BGP it connects different Autonomous Systems (AS) together.
  - Each of the Autonomous System (AS) should have an Autonomous System Number (ASN).
  - BGP creates a TCP based session between two peers on port 179.
  - BGP has two flavours – IBGP (Internal BGP – within a same Autonomous System) and EBGP (External BGP – within a different Autonomous Systems).
  - BGP can also have Transit AS capability, where peers connecting to a same AS can also communicate without establishing a peering connection between them.
- BGP Communities & Attributes:
  TO be added later.

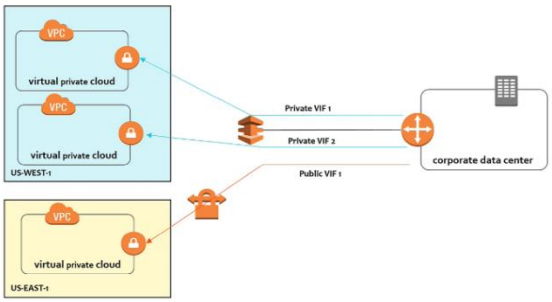
- AWS Direct Connect Inbound routing policies
  - Client should own the prefix that its advertising to AWS VGW.
  - Transitive routing through AWS network is NOT allowed – Traffic must be destined to the AWS prefix ONLY.
  - AWS Direct Connect perform inbound packet filtering to validate that the source of the traffic originated from clients advertise prefix. (Client should own those prefix).
- AWS Direct Connect Outbound routing policies
  - AWS Direct Connect advertise all local and remote AWS prefixes where available, including on-net prefix from other AWS non-region Point-Of-Presence (PoP) where available – example route53 / CloudFront etc. (NO ACCESS TO INTERNET OVER DIRECT CONNECT).
  - Inbound traffic Load sharing between multiple AWS Direct Connect can be achieved by advertising prefix with similar attribute (*AS_PATH attribute should have equal path length for all the available connections* – AWS Direct Connect advertise prefix with minimum path length of 3).
  - To create a failover connection – secondary connection advertise prefix with higher path length then that of the primary connection.
  - AWS advertise all its public prefixes with well know NO_EXPORT community. This prevent client routers to share AWS prefixes with other connected AS (Autonomous System) beyond the network boundaries of the client connection.

AWS ROUTE TABLE NEEDS TO BE MAPPED TO VGW NOT TO THE DIRECT CONNECT OR THE VPN CONNECTION.

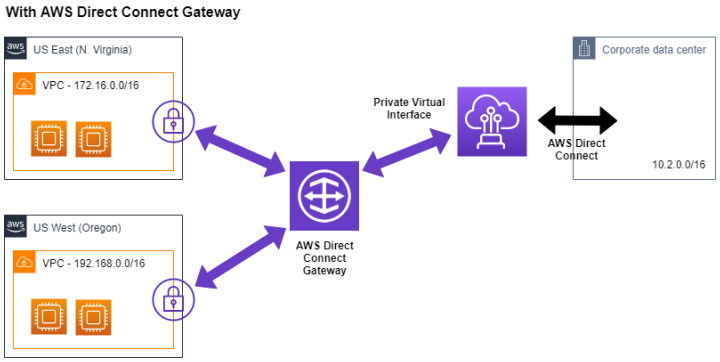
- Link Aggregation Group (LAG)
  - Combining multiple direct connect links into a logical group using Link Aggregated Control Protocol (LACP) is called LAG.
  - Link Aggregation group is a logic interface that used Link Aggregation Control Protocol to aggregate multiple connection at a single AWS Direct Connect endpoint allowing it connect as single managed connection. All connection in the LAG are in active/active mode.
  - LAG can be created from the existing connection or can be provisioned from new connections – standalone connections can be added to the LAG at later point of time.
  - Rule to add connection to a LAG
    - The connection should be of same bandwidth.
    - Up to 4 (four) connections can be added to a LAG.
    - All connection LAG should be terminating at the same AWS Direct Connect Endpoint.
  - One can define number of active connections for a LAG to be operational - default value is set as '0' however one can override the default value – in doing so, if the minimum number of connections is NOT available the LAG will become NON operational. This LAG attribute can be used to prevent overutilization of the remaining connection.
- AWS Direct Connect HA
  - By having a secondary connection build on VPN – which will allow traffic to reach to/from AWS cloud to the data-centre using VPN connection over the internet – this option can only be use as Failover scenario as VPN connectivity over internet is non-reliable.
  - By having a secondary connection build on Direct Connect – this option can be used with both Failover and Active/Active scenario.
- ***Longest prefix match*** – there are two route table entries 10.0.0.0/16 and 10.0.0.0/24. Second cidr block is part of the first cidr block in that case longest prefix match is applied. ONE with longest prefix in this case /24 will be selected. It will go for the specific.
- **Route Table priorities – From AWS to On-Premises**
  - Route table on AWS side determines where the network traffic is directed.
  - For AWS to route traffic from VPC to On-Premises CIDR block - it needs a route entry for On Premises CIDR block destined to VGW (Virtual Gateway). [THIS IS TRUE FOR BOTH DIRECT CONNECT AND FOR VPN CONNECTIONS.] This can be achieved using Static Routing OR by enabling Route propagation where CIDR block entries are auto populated based on the route received from the BGP.
  - In case of the overlapping route entry, and Longest Prefix match can't determine the preferred route, the following rules gets applied – from most preferred to least preferred
    1. Local route will be preferred most.
    2. Static route to Internal Gateway, Virtual Private Gateway, Network Interface of an instance, VPC peering Connection, A NAT gateway, VPC Endpoint will be the 2nd (second) preference.
    3. BGP propagated route from the AWS Direct Connection is always preferred.
    4. Manually add static routes for site-to-site VPN Connection.
    5. BGP propagated routes from site-to-site VPN Connection.

  [VIRTUAL PRIVATE GATEWAY ONLY ROUTES TRAFFIC TO KNOW PREFIXES THROUGH BGP ROUTE PROPAGATION OR THROUGH STATIC ROUTE ENTRY.]
- **Direct Connect Gateway**

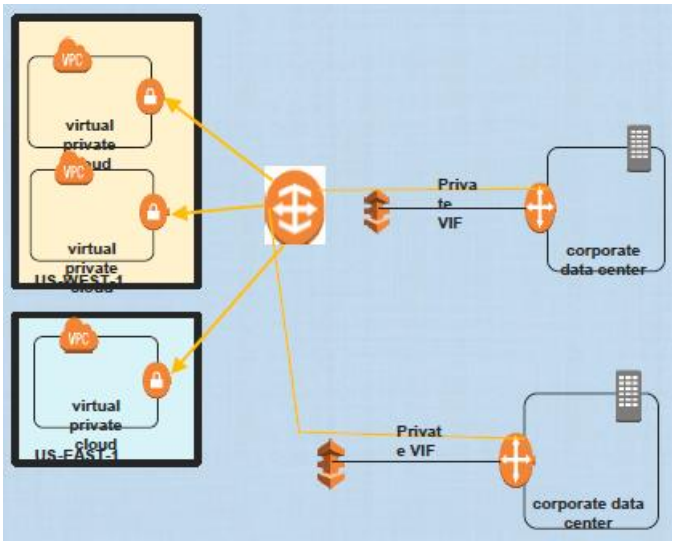
  Prior to using Direct Connect Gateway - In case one needs to connect to different VPC (same or different region) using Direct Connect then – for the region where Direct Connection is connected (associated) one can create Private VIF (Virtual Interface) per VPC and connect. For other region one need to create a Public VIF (Virtual Interface) and connect it over VPN Connection (IPSec tunnels) OR work with AWS Direct Connect Partners to establish a new connection to AWS region of interest. This solution is expensive and cannot be scaled – as it required to maintain large number of VPN Connection (IPSec Tunnel).



When using Direct Connect Gateway: The above mention problem can be overcome after the introduction of the Direct Connect Gateway. A Direct Connect gateway is a globally available resource. One can create the Direct Connect gateway in any public Region and access it from all other public Regions (except China). It also possible to connect multiple direct connection to Direct Connect Gateway.



Even multiple client HQ can be connected using direct connect gateway.



The limitation of Direct Connect Gateway:

- The VPCs that are connected through a Direct Connect Gateway can't have overlapping CIDR block.
- Public VIF can't connect to a Direct Connect Gateway.
- A Direct Connect gateway supports communication between attached private virtual interfaces and associated virtual private gateways only.
- Following connections are NOT SUPPORTED
    - Direct communication between the VPCs that are associated with the Direct Connect gateway
    - Direct communication between the virtual interfaces that are attached to the Direct Connect gateway
    - Direct communication between a virtual interface attached to a Direct Connect gateway and a VPN connection on a virtual private gateway that's associated with the same Direct Connection.
- One cannot associate a virtual private gateway with more than one Direct Connect gateway. Similarly, cannot attach a private virtual interface to more than one Direct Connect gateway.
- A virtual private gateway associated with a Direct Connect gateway must be attached to a VPC.

- **Direct Connect Limits**

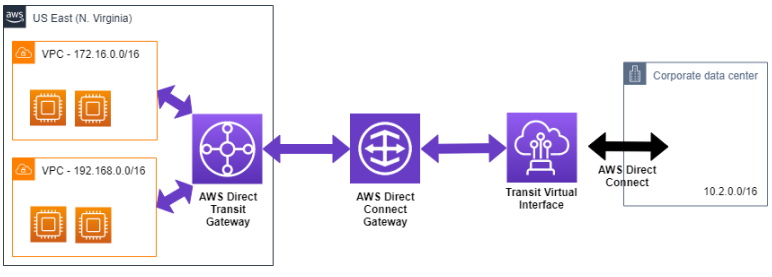
Component	Limit	Comments
Private or public virtual interfaces per AWS Direct Connect dedicated connection	50	This limit cannot be increased.
Transit virtual interfaces per AWS Direct Connect dedicated connection	1	This limit cannot be increased.
Private, public, or transit virtual interfaces per AWS Direct Connect hosted connection[1]	1	This limit cannot be increased.
Active AWS Direct Connect dedicated connections per Region per account	10	
Routes per Border Gateway Protocol (BGP) session on a private virtual interface	100	This limit cannot be increased.
Routes per Border Gateway Protocol (BGP) session on a public virtual interface	1,000	This limit cannot be increased.
Dedicated connections per link aggregation group (LAG)	4	
Link aggregation groups (LAGs) per Region	10	
AWS Direct Connect gateways per account	200	
Virtual private gateways per AWS Direct Connect gateway	10	This limit cannot be increased.
Transit gateways per AWS Direct Connect gateway	3	This limit cannot be increased.
Virtual interfaces per AWS Direct Connect gateway	30	
Number of prefixes from on-premise to AWS on a transit virtual interface	100	This limit cannot be increased.
Number of prefixes from AWS to on-premise on a transit virtual interface	20	This limit cannot be increased.

VPC Peering

- Inter region VPC peering is allowed – traffic between different VPC from different region travels over AWS backbone.
- There is NO SINGLE POINT OF FAILURE in case of VPC peering.
- Edge-to-Edge to peering is NOT allowed. For example, VPC A is peered with VPC B - VPC A is connected to corporate HQ using direct connect, the same connection cannot be used for connecting to VPC B.
  Note: for connecting multiple VPC over VPC peering – one need to create a full mesh. Number of peering connection need to connect all the VPCs are n (n-1)/2 for example for 3 VPCs to connected 3 * ((3-1)/2) = 3 peering connection.

AWS Transit Gateway

- Transit Gateway is a network transit hub that can be used to interconnect VPCs and Corporate HQ.
- It's a regional construct – only able to connect VPCs within the same region.
- Though Transit Gateway, connected VPCs can communicate with each other and also with the corporate HQ – however this behaviour can be altered by creating multiple route tables, to create a control access within the connected VPCs and Corporate Data Centre.
- Transit gateway can be used to connect to VPCs within across accounts.
  Ref Link : https://docs.aws.amazon.com/vpc/latest/tgw/tgw-transit-gateways.html
  Transit Gateway important concepts
  - VPCs / Direct Connect / VPN Connection – all these can be attached to a transit gateway.
  - Default – transit gateway is connected to a single route table. Additional route tables can be attached to segregate / control the route.
  - By default, VPN and Direct Connect are attached to the default table.
  - Each association are attached to single route table – route define in the route table are the ONLY association where communication is possible.
  - Route Propagation can be enabled for the Transit Gateway and the VPN / Direct Connect Connections allowing corporate data centre to connect to all connected VPCs.
    NOTE: Transit Gateway cannot be connected to a Direct Connect – it can ONLY be connected to the through a Direct Connect Gateway.



Connecting Transit gateway with direct connect gateway

[Revision Topics:]

[EC2: SR I/OV – Enhance Networking – Spread / Cluster Placement Group.]

[Difference between NAT Gateway and NAT instance.]

[Host – HA: Create Baston Host behind an autoscaling group – set desired size as 2 (to ensure always 1 Baston Host availability) & select multiple AZ, for cost optimization set desired size as 1 and select multiple AZ.]

[EC2 – ENI - eth0/eth1 – Attaching and De-attaching eth1 from/to EC2 instance retains its MAC Address/Elastic IP/IPv4-IPv6 addresses, this feature can be helpful in installing software that tied to MAC/IP address. (ENI can be re-attached to another instance within the same AZ but can be in a different subnet. OTHER usages of having a secondary ENI attached to the EC2 instance – to have two distinct IP for client and for management request connected to two different subnet – public for client and private for management request.)]

VPC Endpoint

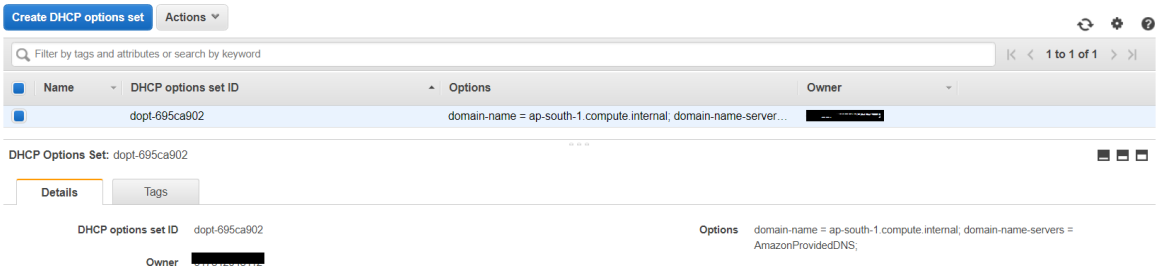
- For accessing the AWS Public Services from public subnet, one need to add a NAT Gateway to access all these services OR they can be accessed using VPC Endpoints.
- When communicating over the VPC endpoint the traffic remains on the AWS network backbone and it does not travel through open internet.
- There are two types of VPC Endpoints – Gateway Endpoint and Interface Endpoint.
- VPC Endpoint can be ONLY used within the region.
- ONLY one is required per VPC.
- VPC Endpoint Gateway – is ONLY for S3 and Dynamo services
- VPC Endpoint Interface – For most of the services we have VPC Endpoint Interfaces.
- VPC Endpoint Private DNS Name -

VPC Endpoint Flow Logs

- VPC Flow logs helps to capture the IP ingress/egress traffic to the VPC.
- The VPC Flow logs can be created at the VPC Level, at subnet level and at the ENI level. Depending upon the desired granularity one can decide the where to enable the VPC Flow log. Additional one can select the type of traffic it needs to be logged – rejected traffic, accepted traffic or both.
- Flow logs can be sent to Amazon CloudWatch Logs OR to AWS S3 bucket.

VPC Endpoint DHCP Option Sets

- One can use on-premises DNS host for AWS instances; however, AWS DNS host are NOT allowed to use for on-premises instances.
- Dynamic Host Configuration protocol provide a standard for passing configuration information to host on TCP/IP network. It's a standard to share information which DNS servers to be used for resolving the DNS names.
- BY default, DHCP option set is created within the VPC pointing to the Amazon DNS server, with domain-name and domain-name-servers option populated.



- One can create custom DHCP (cannot edit the default DHCP option) connect and attached to the VPC, at a given time ONLY one DHCP option can be attached to the VPC.

- Once a new DHCP Option is connected to the VPC, instance need not to be restarted, after some time (4-5 hours) the news DNS will be automatically picked up.

Section 7: Introduction to Digital Certificate and Public and Private Share Encryption

Digital Certificate

- X.509 certificate, SSL, TSL are synonymous.
- X.509 certificates are based on – ITU-T-X.509 format.
- Different Types of Digital Certificates are
  - **Personal**: Used by individuals requiring secure email, digital signature, and web base transactions.
  - **Organizations**: Used by corporate to identify employee for secure email and web base transactions.
  - **Server**: To prove domain name ownership and establishing SSL/TSL base encrypted session between their webserver and a client.
  - **Developer**: To prove code authorship and retain integrity of distributed software.
- CA Authority actions
  - Accept Certificate Request (issuing) application from the entity.
  - Authenticate issuing application to validate its correctness.
  - Issues Certificate.
  - Maintain Details and status about the issuing certificate.
- CA Public key are wide share – even share within web browser to evaluate other certificates without any delay.
- Asymmetric Key / Symmetric Key encryption:
  - Asymmetric Key are less preferred as compare to Symmetric Keys.
  - In case of Asymmetric Keys there are pairs of keys available – Private/Public Key. Client encrypts the data (plain text) using Public Key and Server decrypts the cypher text using the Private Key.
  - Symmetric Key encryption is more efficient then Asymmetric Key encryption which is also knows as Shared Secret key.
  - **AWS uses Symmetric Key encryption for KMS – for sharing the keys its used Asymmetric key.**
  - In case of Symmetric Key encryption there is ONLY ONE key – Client uses the key to encrypt the plaintext – server use the same algorithm (encryption-decryption) algorithm to decrypt the cypher text message.
  - CA and Intermediators for the certificate to be authenticate its chain of trust needs to be authenticated.

Section 8: AWS Elastic Load balancer – Network / Application / Network

Load balancer

There are three types of Elastic Load balancer available – Application Load Balancer (Layer 7 Load balancer)/ Network Load Balancer (Layer 4), AND Classic Load Balancer (Operates at both layers 7 & 4 but now delicate by AWS).

Enabling Proxy Protocol for NLB (TSL/SSL connection – layer 4) – As NLB terminates the incoming request and start a new request to the backend services, for backend services it shows source as NLB instead of the actual client – Enabling proxy protocol ensure that original source is maintain however if the inbound request has pass through an proxy there will be a double proxy header added to the original request.

Enabling Proxy Protocol for ALB (HTTP/HTTPS connection – layer 7) – As ALB terminates the inbound request on receiving, and initiates a new request to the backend services. For backend services to know the actual client address it creates and add original client IP into x-forward-for header custom header. Backend services that desires to filter request based on the actual originator IP addresses, then it needs to process the information from x-forwarded-for header.

[IMPORTANT POINT] Classic Load Balancer DOES NOT support loading of multiple SSL Certificates into a single CLB. ONLY Application Loadbalancer can support multiple SSL certificates and allows to choose the right certificates using SNI.

TCP/SSL load-balancing

Use Case	Front-End Protocol	Front-End Options	Back-End Protocol	Back-End Options	
Basic TCP Loadbalancer	TCP	NA	TCP	NA	Supports Proxy Protocol header
Secure website or application offloading SSL Decryption	SSL	SSL Negotiation	SSL	NA	Required SSL certificate to be deployed to Elastic Loadbalancer Support Proxy Protocol header
Secure website or application using end-to-end encryption with elastic Loadbalancer	SSL	SSL Negotiation	SSL	Backend validation	Required SSL certificate to be deployed to Elastic Loadbalancer. Doesn't support insertion of SNI header on backend SSL connection Does not support the proxy protocol header.

HTTP/HTTPS Load Balancing					
Use Case	**Front-End Protocol**	**Front-End Options**	**Back-End Protocol**	**Back-End Options**	
Basic HTTP Loadbalancer	HTTP	NA	HTTP	NA	Support x-forward-for header.
Secure website / Application using Elastic Loadbalancer to offload SSL decryption	HTTPS	SSL Negotiation	HTTP	NA	Required loading of the certificates to the Elastic Loadbalancer. Supports x-forward-for header.
Secure website or application using end-to-end encryption	HTTPS	SSL Negotiation	HTTPS	Backend validation.	Required loading of the certificates to the Elastic Loadbalancer. Supports x-forward-for header.

[Revision topic]

- ELB – Cross Zone Load Balancing

- ELB health check – there are two type of health checks – TCP (layer 4 response) and HTTP (layer 7 response) checks. For the legacy application, if it doesn't provide standard HTTP response 200 then ELB will mark the instance as unhealthy, in-order to overcome the same configure the check to TPC port. For nonstandard/custom/legacy application which host its application in other port apart from standard HTTP port 80 then it's advisable to use TCP based health check instead of HTTP based health check.
- Sticky Session (server affinity) Best practice to store the session information in persistence store. Server Affinity needs SSL certificate offloading at the Loadbalancer side.
- loading X.509 certificate to ELB or to IAM or managed by ACM. Certificate should be in the same region as that of the ELB.
- ELB can managed cookies if application is not managing the cookies.
- ELB Doesn't supports multiple SSL certificates.
- ELB Cannot be used for two-way mutual certification.
- ELB request can be log using

Section 9: Domain Name Service (Route 53) and Content Distribution Network (CloudFront)

[Revision Topic]

- Different between CNAM and alias name.
- Routing policy – simple routing policy /complex (nested) routing policy.
- DNS Resolver – for hybrid cloud

Section 10: AWS Compute Options

- AWS Support both docker licensing model – docker community edition (one-sourced) and docker enterprise edition (subscription based).
- Docker runs on docker container as well as on the virtual machine.
- Container Orchestration and management System – Container Security System.
    - Container Orchestration System – helps to deploy containers into different virtual machines.
    - Helps in archiving elasticity – when workload increase /decreased – container orchestration increased or decreased number of docker instances to the change in the demand.
    - Container Orchestration System – create virtual networking between different deployed container instance allowing them to communicate among themselves by assigning IP addresses to each deployed docker container.
- Kubernetes – is widely accepted open sourced container management system introduced by google.
- Docker Enterprise Edition – best known commercial container management solution that provides integrated, tested and certified platform for apps running on enterprise Linux, windows operating system and cloud provider.
- Other commercial container management solution is – CoreOS's, Tectonics, Red Hat open shift container platform, Rancher Lab's Rancher.
- IAM role for ECS – EC2 launch type.
    - IAM Container level role – to be kept as minimal as possible.
    - IAM task level role should be defined if any specific role is desired by the task to perform specific actions. (IAM task role). NOTE: ECS container will inherited the IAM task roles by default.

Section 11: Data storage option in AWS and on-Premises

- S3
- EFS
- Amazon FSx
- Amazon FSx Lustre

Section 12: SQL Data Base (DB) options on AWS

- RDS
- Aurora DB
- Aurora Serverless DB
- Amazon Elastic Cache

Section 13: No-SQL Data Base (DB) options on AWS

- DynamoDB
- DynamoDB – Partition Keys – Primary Key, Primary & Sort Key, secondary indexes
- DynamoDB indexes – Local Secondary Indexes and Global Secondary indexes
- DynamoDB Streams – Streams for DynamoDB events
- DAX (DynamoDB accelerator)
- Dynamo DB transactions
- Dynamo DB pricing – Read Capacity Unit and Write Capacity units.

Section 14: Analytics, Data streaming, and Internet of Things (IOT)

Amazon Athena

- Amazon Athena is a serverless interactive query service to query data from S3 using standard SQL.
- Athena can be used to query data directly on – Structured, semi-structured and unstructured data store in S3 in .**CVS** or. JSON or **apache Parquet** or **ORC (Optimize Row Columnar)** format. One can define schema in Athena and can apply it directly on the data stored in S3. Data does is not needed to be loaded into Athena for executing the SQL queries in Athena. Athena charges based on the query it executes.
- Athena can also be integrated with business intelligence tool or SQL clients connecting to Athena through JDBC and ODBC drivers, it can also be connected with AWS Quick Sight for data visualization.
- For enhancing the performance of the Amazon Athena quires, non-columnar data store in s3 can be converted into open source columnar data format such as apache Parquet or ORC format by creating an EMR cluster and converting the data using Hive.
- When Amazon Athena run a query against any S3 data, it also stores the query output in S3 bucket – which comprises of two files output data file in csv format (*.csv) and (*.csv.metadata) format. [*metadata file can be deleted without impacting the results however metadata about the query will be lost*]. Athena retains the output data in s3 bucket for 45 days before deleting it.
- Athena can be used to query logs from different sources stored into a S3 bucket(s), and the result of the query can be visualized using Amazon QuickSight. In order to query the logs using Athena one needs to create Athena schema for such data.

AWS Kinesis

Streaming data - A large amount of data that is generated and sent over a 100-1000's of data source and sent concurrently are called data-streams. In case of streaming data, the size of the data is generally small – in KB or few MBs.

AWS Kinesis has three main services – Kinesis Streams, Kinesis Firehose, and Kinesis Analytics.

AWS Kinesis Firehose

Read data from the AWS Kinesis Streams and process it and delivers it to the destination. AWS Kinesis Firehose can also transform the incoming data as per the destination needs.

The difference between the AWS Kinesis Stream and Kinesis firehose is – One need to write AWS Kinesis Stream custom app to read data from AWS Kinesis Stream and process to the end destination. While AWS Kinesis Firehose can read AWS Kinesis Stream data in real time and process to the end destination.

AWS Kinesis Analytics

To be added later

Elastic Map Reduce (EMR)

- AWS ERM host web scale Hadoop Framework running on EC2 instance and S3 instances.
- It helps users to spend more time on the analysis then having to do time consuming set for Hadoop components.
- ERM launch all node in a same availability zone, for leveraging benefits due to co-location of the resources.
- ERM charges are on hourly basis, as soon as the ERM cluster is started it starts billing.
- The preferred use case of EMR are
    - Web indexing
    - Data Mining
    - Click Stream Analysis
    - Log File analysis
    - Machine learning
    - Financial analysis
    - Scientific simulation
    - Bioinformatics research
- EMR uses Hadoop as its distributed processing engine.
- Hadoop is an open source java software framework that supports data-intensive distributed application running on large commodity hardware. Hadoop uses a processing/programming model known as "*MapReduce*" which is design to process large amount of data parallelly. Its primary purpose is to process the data for Analytics and Business Intelligence.
- Hadoop recap
    - Hadoop is not a data storage solution, it's a framework for massive parallel processing.
    - In map-reduce the data are divided into small fragments of work, each of which may be executed in any node within the cluster.
    - Hadoop is a preferred platform to process Petabytes of data in thousands of commodity machine called as Hadoop nodes.
    - Hadoop is an enabler for certain distributed NO SQL type of distributed databases such as H-base that can allow for data to be spread access multiple servers without impacting the performance.
    - Hadoop is an opensource framework – it has many other related open source frameworks like Apache Hive, Apache Pig, Apache H-Base, which one can use to process data for analytics and business intelligence.
    - Apache Hadoop is an opensource software for distributed computing.
    - Apache Spark is a fast and general computing engine for Hadoop data.
    - H-Base is a scalable distributed database that supports structure data storage for large tables.
    - Hive is a data warehouse and analytics package that provides summarized and ad-hoc querying, its usages a SQL like query call as Hive QL.
    - Pig is an open source analytics package that run on top of Hadoop, it uses a SQL like language to perform analytical queries called Pig Latin.
- EMR architecture
    - Master Node: Every EMR cluster has one node, which coordinate tasks among other processing nodes. Master node also track the status of the task and also the overall health of the cluster. It's possible to create EMR cluster with only a single Master Node.
    - Core Node: The core nodes are the one that run the task and also store the results in Hadoop distributed file system (HDFS). In a single EMR cluster there can be more than one Core Nodes.
    - Task Node: Task nodes are the one that ONLY run distributed task but doesn't store any data. Even if a task node is lost, there is no data loss.
- Amazon EMR integrates with multiple other AWS service to provide capabilities like – compute, network, storage etc.
    - Amazon EC2 instance – EMR uses Amazon EC2 instances for providing compute capabilities, it comprises of all the node of an EMR cluster.
    - Amazon VPC – EMR uses Amazon VPC to create the virtual datacenter where Amazon EC2 instances will be securely hosted and operated.
    - Amazon S3 for storing input and output data.

- Amazon CloudWatch for monitoring and deploying alerts for the EMR cluster.
- AWS IAM for providing required permission for accessing the EMR cluster and its components.
- AWS CloudTrail for logging all the API calls made to the EMR cluster.
- AWS Data pipeline for automating creation of the start-to-end processing of EMR cluster. (When a new data is loaded in S3 bucket, data pipeline will automatically trigger the EMR cluster creation script – morning the processing of the data within the EMR cluster and delete the EMR cluster once the data processing is completed.)

Compute instance for EMR

- EMR provide flexibility for choosing the compute instance type, one can also mix-and-match the compute for better performance – on demand instance for guaranteed processing power while spot instances for additional processing power to complete the job faster.
- One can also mix different types of instance from different instance family to create the EMR cluster.
- Also, add more instance when the workload peaks and remove instances form the EMR cluster during non-peak hours.

File system to be used for EMR

- Choose Hadoop Distributed File System (HDFS) for the storing the data run on master and core modes within the EMR cluster and which is NOT need to be stored outside the EMR cluster.
- Choose Elastic Map Reduce File System (EMRFS) for storing the data in S3 bucket, in order to separate compute and storage – this helps in autoscaling the storage need for the EMR cluster. One need to scale the EMR cluster based on the compute needs, while S3 automatically scales as the storage grows or shrinks.

EMR launches EC2 instance within a single availability zone – as it improves performance of the cluster by keeping the compute instance in close proximity. EMR choose the AZ which has highest number of instances available to run the cluster, use can also change the AZ.

EMR is not a real time ingestion especially for large datasets. Data needs to store/ingested into S3 bucket for EMR to process the data from the S3 bucket. EMR pull the data from the S3 bucket and launch compute instance (EC2 instance) to process the load in parallel, Once processed, EMR will load the data into destination S3 bucket for storage OR for another EMR cluster to pull and process the data.

EMR Hive – data analytics package run on top of Hadoop. Hive uses SQL like query language call Hive QL for processing analysing the data, it can process complex, structure and unstructured data – hive can also query data from DynamoDB table.

Amazon EMR clusters can read and process Amazon Kinesis streams directly, using familiar tools in the Hadoop ecosystem such as Hive, Pig, MapReduce, the Hadoop Streaming API, and Cascading. You can also join real-time data from Amazon Kinesis with existing data on Amazon S3, Amazon DynamoDB, and HDFS in a running cluster. You can directly load the data from Amazon EMR to Amazon S3 or DynamoDB for post-processing activities.

EMR cannot directly writes the data processing results back to Kinesis streams – it needs to store the output into S3 bucket.

AWS Data Pipeline

AWS Data Pipeline is a webservice that makes it easy to schedule regular data movement and data transform / processing activity in AWS cloud environment.

AWS Data Pipeline is a highly scalable and fully managed service – which can be used to integrating on premises and AWS cloud data sources, it can also help in creating distributed data copy, SQL transformation, map reduce application, custom script against destination such as S3, DynamoDB, AWS RDS etc.

AWS Data Pipeline components

Task Runner – Task runner are the application, when assigned any task perform the action on the task and report back to the AWS Data Pipeline with its final status – AWS Data Pipeline uses Amazon EC2 instance, AWS EMR nodes or on-premises compute nodes as compute to perform its tasks.

Data Node – Data node defines the location and the type of data that the AWS Data Pipeline uses as input or output node. Example of data nodes are – SQLDataNodes ReadShiftDataNodes, S3DataNodes, DynamoDBDatanodes etc.

Databases – Supported databases are – RDS, Redshift, and JDBC connected databases.

Resources – EC2, EMR nodes OR on premises compute nodes.

Activities – activity are the action that AWS Data pipeline performs on behalf of the user, example of activities are – EMR or Hive job, copy activity, SQL quires, command line scripts. There are two types of activities – ***custom activities*** and ***standard activities***. Examples of standard (pre-configure) activities are:

- ***CopyActivity***: This activity can copy data between S3 and JDBC datasources OR run a SQL query and store the output of the query in an S3 bucket.
- ***HiveActivity***: The activity that allows to execute a Hive query.
- ***EMRActivity***: This allow to run arbitrary EMR jobs.
- ***ShellCOmmandActivity***: this allows to run custom shell commands as part of the AWS Data pipeline

Pre-Conditions – is a pipeline component that contain a conditional statement that must be true before an activity can be executed. Example of a pre-conditions is, check source data exists on the S3 bucket before triggering the AWS data pipeline. There are many pre-build AWS data pipeline quires already exists, for example

- ***DynamoDBtableExists*** – this precondition checks the existence of the DynamoDB table before triggering the activity.
- ***DynamoDataExists*** – this precondition check if the if the data exists in the DynamoDB before triggering the activity.
- ***S3KeyExists*** – this precondition checks the existence of the specific path before triggering the activity.
- ***S3PrefixExists*** – this precondition checks if at least one file exists in the S3 bucket before triggering the activity.
- ***ShellCommandPreconditon*** – this precondition check before execution of the shell command.

AWS provided task runner packages that can be installed on premises, which continuously pool AWS hosted data-pipeline. Once any task is available that needs to be performed by the on-premises task runner AWS data pipeline will forward the task to the designated on-premises task runner to execute the task. It advisable to have multiple tasks running instances running on premises to improve availability.

AWS Data Pipeline use cases

- Moving to the cloud: on-premises data can be copied to S3, making it available to various AWS services to consume.
- ETL data to Amazon Redshift: AWS Data pipeline can, can copy the data into S3 from on premises data source and run analytics using SQL queries before loading it into AWS RDS, DynamoDB or Redshift.
- ETL Unstructured data: Unstructured data like clickstream can be processed using EMR hive or pig task and later can be uploaded into Amazon Redshift combining other data loaded from relational data sources.
- Data Load and Extract: AWS Data pipeline can be used to load AWS RDS or Amazon Redshift data to/from S3
- Loading AWS logs into redshift
- Amazon DynamoDB backup and recovery – one can export the DynamoDB table data into S3 for backup and recovery. It can also be used to copy the DynamoDB data into another region.

Quick Sight

Data visualization tool for AWS cloud data source, as wells as on premises data source.

AWS Quick sight is a business analytics services that can be used to build data visualization and perform ad hoc analysis and get business insights from the intended data.

It can auto discover AWS data sources and also works with on premises data sources. For on premises data source, one needs to configure the on-premises data sources in order for quick sight to create the visualization on the data.

Quick sight can to scale up to hundreds and thousands of organization users and deliver responsive performance by using **robust in memory engine (SPICE)**. SPICE is Amazon QuickSight's Super-fast, Parallel, In-memory Calculation Engine and it's engineered to rapidly perform advanced calculations and serve data. After you import your data into SPICE, its storage and processing capacity speed up your analytical queries. By using SPICE rather than running direct queries on your data, you save time. You save time because you don't need to retrieve the data every time that you change an analysis or update a visual. SPICE stores your data until you choose to delete it. You can improve performance by importing the data into SPICE instead of using a direct query to the database. Default SPICE capacity is allocated to your home AWS Region. The other AWS Regions have no SPICE capacity unless it purchased some. Up to 1 TB of additional SPCIE capacity can be purchase.

AWS quicksight has two versions – standard and enterprise version.

AWS quicksight can create data visualization on own account datasources OR shared account datasources.

One can query the data as is or can import the data into SPICE and perform visualization on the imported data. As part of data preparation – transforming the raw data for use in analysis one can perform the following:

- Filtering out the data so that one can focus on what is important, leaving the rest.
- Renaming the data that makes the data easy to read.
- Changing the data type so that they become more useful.
- Adding calculated fields to enhance analysis
- Creating SQL queries to refine data.

Data preparation can be done on the data with/without importing the data into SPICE engine.

Filtering out the data so one can focus on what is important – filter data are called datasets. Dataset specifies the fields and rows that needs to be used for analysis, in addition to the raw data, datasets also stores the change made to the raw data – so that next time one uses the same data sets does not need to make the same changes.

Amazon quick sight can be connected to following data sources:

- Amazon Athena
- Amazon Aurora
- Amazon Redshift
- Amazon Redshift spectrum.
- Amazon S3
- Amazon S3 Analytics
- Apache Spark
- MariaDB
- Microsoft SQL
- Postgres SQL
- Presto
- Snowflakes
- Teradata

Amazon Glue

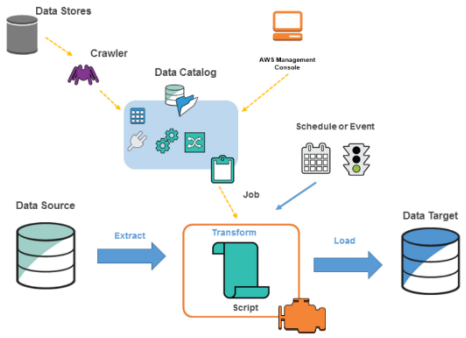
Serverless managed ETL service from AWS, it can automate the time-consuming data preparation steps for analytics. It makes it simple and cost effective to categorise, clean, move data reliably. AWS Glue protect the data at rest and at transit.

AWS Glue uses other AWS services to orchestrate your ETL (extract, transform, and load) jobs to build data warehouses and data lakes and generate output streams.

AWS Glue takes care of provisioning and managing the resources that are required to run your workload. You don't need to create the infrastructure for an ETL tool. To reduce startup time, AWS Glue uses an instance from its warm pool of instances to run your workload.

With AWS Glue, you create jobs using table definitions in your Data Catalog. Jobs consist of scripts that contain the programming logic that performs the transformation. You use triggers to initiate jobs either on a schedule or as a result of a specified event.

With your input, AWS also Glue generates the code that's required to transform your data from source to target. One can also provide scripts in the AWS Glue console or API to process your data.



AWS Glue consist of following feature:

Automatic Clawing: based on the provided dataset, it populates a central metadata repository known as AWS Glue data catalogue. It creates data table definition (schema), physical location, add business relevant attributes, as wells as tracks the changes on the data over the time.

Job Authoring: an ETL engine that automatically recommends and generates Python or Scala code, to transform the source data into target schema.

- It run the ETL job on a fully manged scale-out spark environment to load data into its destination.
- It also allows the setup, orchestration and monitoring of complex data flow.

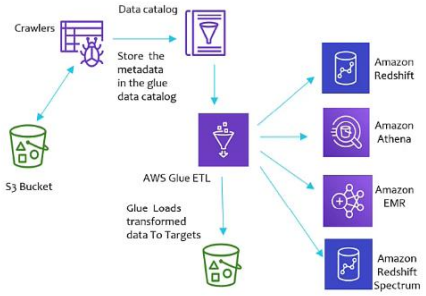
Job Execution, a flexible scheduler that handles dependency resolution, job monitoring and retries.

AWS Glue Data Catalogue: The AWS Glue Data Catalogue is a central repository to store structural and operational metadata for all relevant data assets.

AWS Glue automatically discovers and profile the data via the Glue data **Catalogue**.

The AWS Glue Catalogue is an Apache hive metastore compatible – it can be used a drop-in replacement for Apache Hive Megastore for Big Data application running on EMR.

AWS Glue Catalogue also provide out-of-the-box integration for Amazon Athena, EMR and Amazon Redshift spectrum.



AWS Glue Usages

AWS Glue can be used to run serverless queries against S3 data lakes. Instead of using Athena and creating schema manually, one can use AWS Glue to query S3 data lake directly from the AWS Athena or AWS Redshift spectrum. AWS Glue catalogue will internally manage to sync with meta data and schema based on the S3 data – and keep itself up to date with changes. AWS Athena or AWS Redshift spectrum, can directly use the AWS Glue catalogue schema definition for querying the data in the S3. One can use the AWS Glue's single unified interface to query the data.

Aws Glue can be used to cleanse, format, organized data from disparate data sources and load into Enterprise data warehouse.

Aws Glue can be used to automated ELT pipeline – once a new data is uploaded into S3 bucket, a Lambda function will be triggered, which will trigger the AWS Glue Crawler. Once crawler creates the data schema – ETL pipeline can be triggered to extract/transform/load data into different data source for analytics or further processing.

How to get Apache Hive Metastore into Glue – one need to create an EMR job to transform Apache Hive Metastore data into S3 bucket, from the S3 bucket the data can be loaded into AWS Glue.

AWS Data Pipeline	AWS Glue

It provides more control and flexibility on the execution environment, compute resources, as well as the code that transforms the data.	AWS Glue provides a serverless environment to run Apache Spark environment.
If one needs more control over the environment / compute and the transformation logic can't be written on Scala or Phyton then prefer AWS Data Pipeline	If the solution needs to be built on apache sparks, and the code should be written on Scala or Python then AWS Glue is a better option.
It mostly used for data migration	Its mostly used to ETL jobs.

<Pending – compare Glue with other AWS data tools>

Section 14: Amazon Media Services

Amazon Kinesis Video Streams

Amazon Kinesis Video Streams is a managed service from Amazon which can be used for ingesting large amount of video/audio feed in real time or near real time from customers (IoT Devices) and store the feed durable on the cloud storage processed them in real time or in batches for video analytics.

Producer – are the device(es) that generates videos or non-video data like – audio feeds, images, RADAR data in real time or near real time to a single stream or different streams (producer can produce video data to one stream and audio data to another streams).

Kinesis Video Streams producer libraries – This are the libraries that can be installed on the producers that can help to securely connect and reliably stream video content in real time or send in batches for processing.

Kinesis Video Streams – it helps in transporting Kinesis video data to Kinesis consumers for processing in real time or it can also store the data for some before consumers consumes data. Typically, there a producer that produces kinesis video data and one ore more consumes data in parallel.

Kinesis Parser Libraries – these libraries help in parsing the uploaded video in real time.

- Benefit of Kinesis video streaming
- Connect and stream from millions of devices
- Durably store, encrypts and index data
- Customer can focus on managing application instead of the infrastructure.
- It can be used to build application to process data in real time or in batches.
- It encrypts the content during transit and in rest.

Kinesis Video Streams – putmedia API

Kinesis Video Streams – getmedia API – from the requested index, the media content will be fetched in increasing sequence. The media file is in MKV format.

Playing video using HLS (HTTP live streaming) protocol or can even uses getmedia API and play it offline.

Monitoring and Logging – AWS CloudWatch can be used to monitor Kinesis media streams with real time metrics. Kinesis media streams also gets integrated with cloudTrail for auditing the API calls made to the Kinesis media streams.

Section 15: Architecting a Continuous Integration and Continuous Deployment Process on AWS

Amazon X-Ray: The challenging faced in centralized logging

1. Single component with distributed component.
2. Tracking request can be challenging.
3. Different log system – different format.
4. Finding bottle neck on a distributed application

5. Logging and tracking request in a microservice application hosted on multiple containers.
6. Creating an end-to-end tracking of a request.
7. Detecting latency in each component of the application.
8. Identifying the root-cause of the failure/bug in a distributed application.

Concepts of AWS – X-Ray

Trace: When a new request is arrived, a unique trace id is generated and throughout the flow the same id will be maintained as the request move from one component to another. An end-to-end tracking of single unique traceID is called as a trace.

Segments: is a hop in the complete end-to-end trace. It's a encapsulation of all the datapoints for a single components of a distributed application.

Sampling: AWS X-Ray does not take all the request received for a particular segment, instead its use sampling mechanism to evaluate the lag and other parameters. Thus AWS-X-Ray cannot be used for audit and compliance prospective as it does not guaranteed completeness of data.

Annotation: are the tags/meta data ended to a particular segment of a distributed application. It can be system generated or manually added once – for example developers can add region specific data in the annotation to draw a better picture.

X-Ray Error: these are the system annotation associated with segment for a call that results in error response.

X-Ray Agent: For some AWS service, one needs to install X-Ray agent that can collect and sends information from application logs to AWS -X-Ray service for storing, sampling and visualizing. These agents are available for Lunix , redhat and windows servers.

For ECS and EC2 on needs to include X-Agent in the image , if its in beanstalk then one need to include specific language libraries for A-Ray to work.

X-Ray Groups: Groups are collection of traces that are defined by a filter expression, which decides which traces join the group. Groups can be used to generates additional service graphs and supply Amazon Cloud Watch metrics.

X-Ray can visualize data from different regions, X-ray stores data locally in the region share just enough data to the client application to create visual traces of data combining all the region information. Data sent to X-Ray service are available in real time within 30 sec or less for retrieval or filtering and it stores the data for 30 days. One can query the X-Ray trace data for 30 days in past.

X-Ray can be used to create visualization of an application request trace across AWS account – X-Ray can assume role to publish its data into a centralized account.

AWS-X Ray can be integrated with AWS-Config and AWS Cloud Trails for any auditing and compliance purpose.

Section 17: Designing Secure Solution – AWS Security, Identity and compliance service.

IAM Role, is a permission which can be assume be any other principle to get access to a specific service or resources momentarily. IAM Role uses STS for providing access instead of relying on long term credentials (password or access key). STS creates temporary credentials dynamically and provide to the user who is assuming that role.

IAM Role can be assumed by

- An IAM user from the same AWS Account
- An IAM user from the different AWS Account
- Any web services offer by AWS like EC2, CodeDeploy , CodePipeline etc
- An external user authenticated By an external Identity provider (IdP) service that is compatible with SAM2.0 or OpenID Connect (OIDC) or a custom build identity provider.

When to use AWS IAM Role and when to use Resource base Policy : When a user assumes an IAM Role it temporarily lose its other access part from the what is mention in the role, while in case of the resource base policies the user does not need to lose any access – they can have both access at the same time.

Cross account access with resource base policy has an advantage over Role – as the use can have both the access of the trustee account as well as the trusted account at the same time. This is required at time when, user need to perform account with both access like copying a file from one Aws account to another.

Resource base policies are applicable to the following

- Amazon S3
- Amazon Glacier Values
- Amazon SNS
- Amazon SQS

AWS Service Role: These are the service role which AWS services can assume and perform account on behalf of the user(s).

Note : For EC2 Service don't need to request for temporary credentials for a service role, it would made available to EC2 service by AWS and will be rotating the credentials (credentials will be refresh 5 min before expiry), while for the other services they need to request for a temporary credentials for a service role.

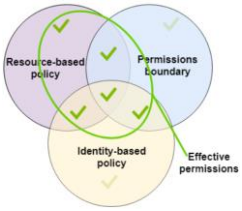
Instance Profile: One need to create and assign instance profile for EC2 to attach Service Role. **Only one role can be assigned to EC2 instance at a given time, all applications needs to share the same role and permission.**

IAM Cross account access – IAM user/resources from one account to have access to another AWS account service/resources.

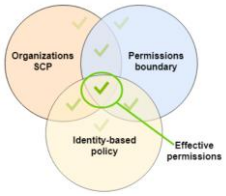
Identity-based policies – Attach managed and inline policies to IAM identities (users, groups to which users belong, or roles). Identity-based policies grant permissions to an identity.

Permission Boundaries: Can be attached to IAM entities – this is a json base police that DOES NOT provide access to the IAM entries – however it creates a boundary of maximum permissions that an identity-based policy can grant to an IAM entity.

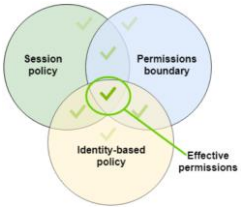
Resource-based policies control how the specified principal can access the resource to which the policy is attached. Within an account, an implicit deny in a permissions boundary does not limit the permissions granted by a resource-based policy. Permissions boundaries reduce permissions that are granted to an entity by identity-based policies, and then resource-based policies provide additional permissions to the entity. In this case, the effective permissions are everything that is allowed by the resource-based policy and the intersection of the permissions boundary and the identity-based policy. An explicit deny in any of these policies overrides the allow.



Organizations SCPs – SCPs are applied to an entire AWS account. They limit permissions for every request made by a principal within the account. An IAM entity (user or role) can make a request that is affected by an SCP, a permissions boundary, and an identity-based policy. In this case, the request is allowed only if all three policy types allow it. The effective permissions are the intersection of all three policy types. An explicit deny in any of these policies overrides the allow.



Session policies – Session policies are advanced policies that you pass as a parameter when you programmatically create a temporary session for a role or federated user. The permissions for a session come from the IAM entity (user or role) used to create the session and from the session policy. The entity's identity-based policy permissions are limited by the session policy and the permissions boundary. The effective permissions for this set of policy types are the intersection of all three policy types. An explicit deny in any of these policies overrides the allow



Access control lists (ACLs) – Use ACLs to control which principals in other accounts can access the resource to which the ACL is attached. ACLs are similar to resource-based policies, although they are the only policy type that does not use the JSON policy document structure. ACLs are cross-account permissions policies that grant permissions to the specified principal. ACLs cannot grant permissions to entities within the same account.

IAM Polices deep dive

There are 6 types of IAM polices

1. Identity base Policy
2. Resource Base Policy
3. Permission boundaries
4. Organization SCP (Service Control Policies)
5. ACL (Access Control List)
6. Session Policies

Identity Base Policies are attached to Role, User or a Groups. There are two type of identity base policies – Managed Policies (AWS managed or Customer managed) and Inline Policy.

Inline polices are the one that embedded directly to a user, group or roles. IT's not recommended to use inline polices.

Resource Base Policies are added directly to the resource – like S3 bucket etc. These policies are used to provide cross account access – in cross account access scenario adding a resource base policy to grant access to a specific resource is half part of establishing trust, one also need to use identity base policies to add access to the identity of the AWS account. This is not needed when the principle and the resources are in the same account.

STS (Security Token Service): The STS is a web service that enables one to access temporary token (limited privilege credentials for IAM users or for users that are authenticated by trusted IdP (federated users).

Temporary security credentials – cannot be extended after expiry however one can request for a new temporary credentials.

STS API

- **AssumeRole** – evaluate and return temporary security credentials that includes – KeyID,secrectAccessKey and a securityToken this can be used by IAM users or users who want to extend the expired security token.

Inline policy can be attached to STS AssumeRole operation which can have upto 10 managed policies to manage resulting session permission which is the intersection of what Assume role can access & what manage polices allowing to access. (intersection of the role's identity-based policy and the session policies.)

To assumeRole across account , AWS trust policy should be added to trust the account to which access is shared. For same account users one can

a. Attach a policy to the user (identical to the previous user in a different account).
b. Add the user as a principal directly in the role's trust policy.

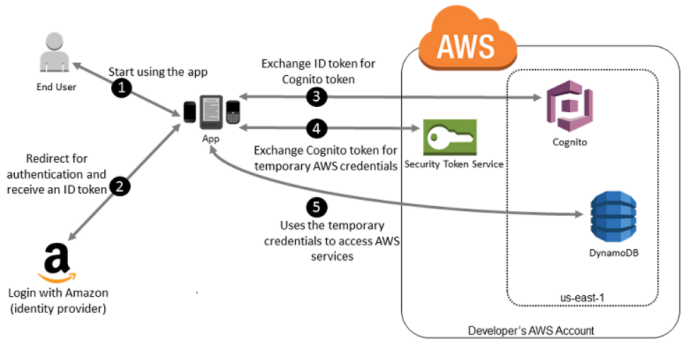
In assumeRole operation session tags and MAF authentication can be added. Administrator can have granular policy stating which session tags are allowed. In trusted policies one can add additional condition to make MAF authentication allowed.

Input Parameter for AssummeRole (duration, externalID, JSON policy (session policy)/ session policy ARN, MFA serialNumber, MFA token Code, tag name)

Response from AssumeRole (*AssumeRoleTemporaryUser*, *Credentials*)

- **AssumeRoleWithSAML** -any user can request for STS token using AssumeRoleWithSAML API with SAML response from known IdP (identity provider).
- **AssumeRoleWithWebIdentity** - any user can request for STS token using AssumeRoleWithWebIdentity API with web identity token from known IdP (identity provider).
- **GetSessionToken** – IAM user or root user can call this API to get SessionToken.
- **GetFederatedToken** – IAM user or root user can call this API to get federatedToken.

One can pass the IAM Session policy as a parameter to most of the AWS STS API, to be used in conjuction with the other policies affecting the user to determine what the user is allowed to do with the temporary credentials that result from the API call. *User are granted the resulting session's permissions that are granted in both role's identity-based policy and the session policies.*This is not supported for GetSessionToken API.

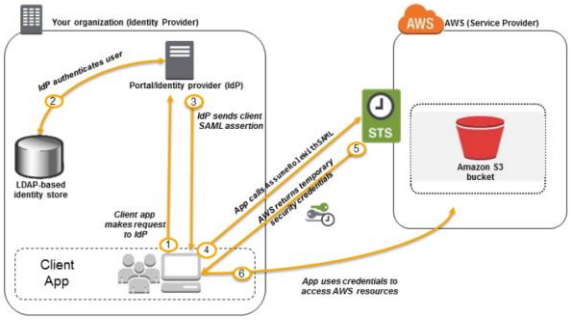


For each of the client application needs to be register with the IdP (Identity Provider) and have their assigned applicationID.

1. Customer, login into the application needs to forwarded to IdP for authentication along with the register applicationID of the application.
2. On successful authentication with the IdP, client will received a OpenConnect ID token, which then can be forwarded to cognito in return to get KeyID , SecretKey and SessionToken.
3. Use can use the SecretKey to access the service.

LDAP On premises / Identity Broker should support SMAL 2.0 (XML base Security Markup Assertion Language). AWS can be authenticated by on premises Active Directory using Active Directory Federation Service (ADFS) 2.0 and Security Assertion Markup Language 2.0.

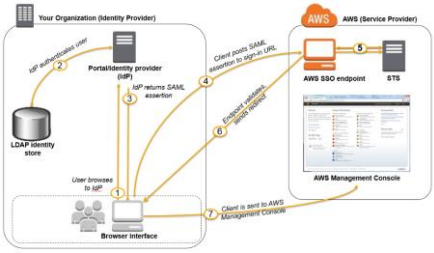
Invoking AssumeRolewithSAML for accessing AWS services/resources based on the SAML token provided by the on premises SAML Identity broker.



1. A user in your organization uses a client app to request authentication from your organization's IdP.
2. The IdP authenticates the user against your organization's identity store.
3. The IdP constructs a SAML assertion with information about the user and sends the assertion to the client app.
4. The client app calls the AWS STS AssumeRoleWithSAML API, passing the ARN of the SAML provider, the ARN of the role to assume, and the SAML assertion from IdP.
5. The API response to the client app includes temporary security credentials.
6. The client app uses the temporary security credentials to call Amazon S3 API operations.

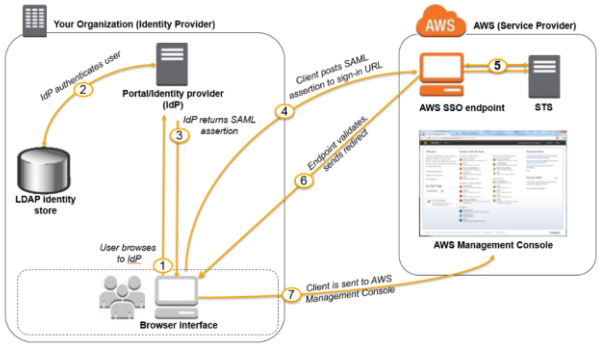
For on premises IdP to work with AWS, one need to create trust between on premises IdP and the AWS account this can be done by importing AWS SAML metadata xml file into IdP and configuring on premises IdP SAML metadata into AWS account, then add a new role which can be assume by the IdP authenticated user, in the role add a trust policy to and set SAML provider arn as principle to establish trust.

SSO configuration from on premises to AWS console.



1. The user browses to your organization's portal and selects the option to go to the AWS Management Console. In your organization, the portal is typically a function of your IdP that handles the exchange of trust between your organization and AWS. For example, in Active Directory Federation Services, the portal URL is: https://ADFSServiceName/adfs/ls/IdpInitiatedSignOn.aspx
2. The portal verifies the user's identity in your organization.
3. The portal generates a SAML authentication response that includes assertions that identify the user and include attributes about the user. You can also configure your IdP to include a SAML assertion attribute called SessionDuration that specifies how long the console session is valid. You can also configure the IdP to pass attributes as session tags. The portal sends this response to the client browser.
4. The client browser is redirected to the AWS single sign-on endpoint and posts the SAML assertion.
5. The endpoint requests temporary security credentials on behalf of the user and creates a console sign-in URL that uses those credentials.
6. AWS sends the sign-in URL back to the client as a redirect.
7. The client browser is redirected to the AWS Management Console. If the SAML authentication response includes attributes that map to multiple IAM roles, the user is first prompted to select the role for accessing the console.
   These steps are transparent to the user, user will NOT be aware of the background steps perform on their behalf to connect to AWS and provide a seamless login experience.

Enabling federated users to use SAML token, if the IpD is NOT compatible to the SAML2.0 standard. In this case one need to write & run an additional code achieving the same.



To be added from - https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_providers_enable-console-custom-url.html

1. Verify that the user is authenticated by your local identity system.
2. Call the AWS Security Token Service (AWS STS) AssumeRole (recommended) or GetFederationToken API operations to obtain temporary security credentials for the user.
3. Call the AWS federation endpoint and supply the temporary security credentials to request a sign-in token.

When to use IAM user and when to use federated user

Use IAM user	Use IAM role	User Federate user
If there are one or very limited set of users who need AWS access	When AWS resource needs access to another service or resources. DON'T embedded IAM user credentials within an AWS resource.	When on primies has full functional SAML 2.0 supported IdP then create a federated user.
If there are NOT IdP provider on premise which can be configure to provide access on AWS environment.		

Active Directory Services available on the AWS

- Active Directory Service for Microsoft Active Directory:
- Simple AD
- AD Connector
- Amazon Cloud Directory
- Cognito

	When to use?	How much load it can support?	Snapshot supported	Supports Connecting to MS SQL RDS	Supports Trust Relationship
Active Directory Service for Microsoft Active Directory	Full-fledged Microsoft Active Directory feature.	> 5,000 users (for standard) > 30,000 users (enterprise edition)	YES	YES	YES
AD Connector	For connecting to on premises AD	Proxy to be used to connect to on premises AD 500 (SMALL)	NA	NO	Proxy

		5000 (LARGE)			
Simple AD	No Advance Microsoft Active Directory feature required. Samba 4 active directory compatible server. Supports basic AD feature – joining Lunix domain, windows base EC2 instance, Kerberos SSO, group policies.	< 5000 users (LARGE) < 2000 user (SMALL)	YES	NO	NO

AWS managed Microsoft Active Directory, can be configure to have share trust with the on-premises Microsoft active directory service – for these one need to establish VPN connection or direct connect to configure the Trust.

AWS Managed Microsoft AD support – AWS application and services like Amazon workspace, Amazon wordDocs , Amazon QuickSight, Amazon Chime, Amazon MS SQL RDS.

AWS Managed Microsoft AD provides

- Fine Grained password policy management
- LDAP encryption through Secure Socket layer
- Approved for AWS cloud for application that are HIPPA or PCIDS compliant.
- This can be configured to support MFA authentication, by integrating it with standard RADIUS base physical or virtual MFA device.
- Can be integrated with CloudTrail and automated snapshot and automated recovery.
- Can be scaled by adding additional domain controller.
- AWS Managed Microsoft AD – Standard edition for 5000 users
- AWS Managed Microsoft AD – Enterprise edition for up to 30,000 users.

EC2 based Microsoft Active Directory in AWS cloud. In order to create a replication with the on-premises AD one need to install Microsoft Active Directory on AWS and join on premises Microsoft Active Directory in replication mode. (Same cannot be done using AWS managed Microsoft Active Directory service – one can share trust between on premises Microsoft Active Directory, but cannot join in replication mode.) in order to connect on-premises AD, from EC2 instance Microsoft Active Directory one need to create VPN connection between client location and AWS site. (Note: this cannot be communicated over direct connect as its non-encrypted channel).

NOTE: Microsoft Active Directory in replication mode is less secure as compare to Microsoft Active Directory in trust sharing mode as in replication mode its more chattier.

Microsoft AD and AD Connector supports MFA with RADIUS compatible devices.

Option	Fully Managed by AWS	SSO support	RDS MS SQL support	MFA Support	Best Use case	Snapshots for Backup and DR	Same Policies on premise and in AWS	Can authenticate to AWS console w/out SAML2.0	Cloud Trail and SNS integration	Supports Multi AZ
AWS Active Directory Service for MS AD	Yes - Standard < 5000 users, and Enterprise >5000 users	Yes, SAML 2.0 , No replication option to On Premise AD (VPN)	Yes	Yes	> 5000 users, or when Trust with on-premise MS AD	Yes (Automated and Manual)	Can be achieved with Trust	Yes	Yes	Yes (default)
AD Connector	Small <=500 users and Large <=5000 Users	Yes, however, No LDAP DB in the cloud, (Requires VPN or DX)	No	Yes	When you want to use your On-premise Existing AD directory with your AWS application	N/A	No policies in AWS, everything is on premise	Yes, allows users to log in to AWS using their AD credentials		N/A
Simple AD	Yes	Yes (Kerberos based), No trust relations or SAML based federation	No	No	<= 5000 users, Best for a low cost Active Directory compatible service in AWS	Yes (Automated and Manual)		Yes		Deployed on 2 EC2 instances in 2 different subnets in a VPC
You own MS AD on EC2 instances (unmanaged)	No , Small <=500 users and Large <=5000 Users	MS ADs in AWS can join On PremiseAD **(Replication) Not trust (VPN) But will need trust with AS for**		Yes	Low scale, Low cost, basic features AD like requirement, and for LDAP aware applications	Not automated, you need to do your own snapshots of the EBS volumes	Yes	You need SAML2.0 for SSO	N/A	Your own Design

Encryption / Hardware Security Module

There are two types of encryption – symmetric key (same key used for encryption and decryption) and asymmetric keys (different keys are used for encryption and decryption).

KMI (key management infrastructure) – Hardware Security Module is a KMI offered by AWS – its FIPS 140-2 certified platform.

- HSM supports
- Digital Right Management (DRM)
- Public Key Infrastructure
- Document Signing
- Cryptographic Functions.

KMS – are confine to a region, within a region its highly durable (99.999999999 %) as its store in multiple availability zone within the region.

CMS (Customer Master Key) created by AWS or imported by customer cannot be exported. This are generally used to encrypt data keys max of 4KB of data can be encrypted by CMK.

Typically, encryption of data key with a Customer Master Key is called ***Envelope Encryption***. There are two type of CMK –

- **Customer managed CMKs** (customer create manage and rotates the key content on their own – still they need to use AWS KMS as they need KMI Key Management Infrastructure.
- **aws managed CMKs** – aws service generated CMK, for each of the services with naming format of a CMK is `aws/<service-name>`

  `NOTE: if aws managed CMK are used, then AWS will also manage the policies on your behalf. However, if customer managed CMK are used, user also needs to manage the policies AWS will not maintain the polices.`

Default Master Key are the CMK that gets create when AWS service first uses encryption for a particular service.

Data Key – can be used to manage any amount of data.

Envelop encryption – is a process of encrypting data using a data key and then encrypting the data key with a customer master key.

Key policies – these are the polices that determines who can access the key. One can add, remove and modify access to a customer manage key; however, one cannot edit key polices of AWS generate customer master keys.

Cloud Trail logs all calls to KMS API usages coming from AWS console, AWS API, AWS CLI, AWS SDK.

Key rotation - AWS does not encourage to use the same CMK for a long time, one needs to rotate the CMK at regular intervals (decrypt the data key and re-encrypt the data key using the new CMK). If user selects automatic key rotation for the customer managed CMK AWS generates new cryptographic key material every year, AWS KMS also saves the old cryptographic key material to decrypt the OLD data keys

Manual rotation of the Key – customer creating a new Key and then using that key to encrypt the data. This is best suited when the key rotation schedule is different from the annual key rotation which is offered by AWS when selected automatic key rotation.

HOW EBS ENCRYPTION TAKE PLACE USING AWS KMS?

1. when an EBS encrypted volume is created , EBS send a `GenerateDataKeyWithoutPlainText` request to KMS specifying the CMK that was chose to encrypt the EBS volume.
2. AWS KMS generates a data-key based encrypts it under the specified CMK and sends back the encrypted data key back to EBS.
3. EBS stores the encrypted data key in the within the volume datadata.
4. When the volume is attached to the EC2 instance, it retrieves the encrypted data key form the EBS volume metadata and send it KMS for decryption.
5. AWS KMS decrypts the data key and sends the plain-text data key back to EC2 instance.
6. Amazon EC2 instance uses the plaintext data key in the hypervisor memory to encrypt I/O to the EBS volume. The plain text data key persisted in EC2 hypervisor memory as long as the EBS volume is attached to the EC2 instance.

Securing data at rest – Encryption

Client-Side Encryption	**Server-Side Encryption**
Encrypt the data on the client side then uploaded the encrypted data instead of uploading un-encrypted data.	Encryption that are performed on the AWS Service (for example S3), client uploads un encrypted data which AWS encrypts and stores it on their servers.
Client-side encryption can be form on in filesystem level or at block level.	
Client-side encryption (filesystem or block level) cannot be applied to root volume.	
AWS SDK provided APIs for client-side encryption – one can also use third party software for the same.	
AWS also provide AWS HSM (Hardware Security Module) as KMI (Key management infrastructure) to manage client side encryption key – however in this , AWS will NOT manage any key rotation or policy management etc.	

There are three flavors of S3 – Server-Side encryption

- SSE-S3 (each of object will be encrypted using a unique data key).
- SSE-C (Customer manage the key)

- SSE-KMS (*if auditing is required on the key utilization, use SSE-KMS*).

NOTE: while uploading a data into S3 using PUT request one needs to include x-amz-server-side-encryption header with the value of desired S3 encryption.

s3: x-amz-server-side-encryption: “aws:kms”

AWS Glacier – all data are always encrypted

AWS Storage Gateway – all data are always encrypted

AWS EMR – while storing the data within S3 bucket one can chose to encrypt the data using AWS SS SSE-KMS

RDS – for an RDS encrypted instance, RDS snapshots, automated backup and read replicas are encrypted. When Read replicas and the master DB instance are in the same instance then the data will be encrypted using the same key, if read replica are in different region then one need to chose a different key to encrypt the data.

NOTE: if the master data base is unencrypted – one can create an encrypted read replica BUT reverse is not true, if the master RDS instance is encrypted, one CANNOT create a unencrypted read replicas.

For Oracle RDS and MS SQL Server – provide TDE (Transparent Data Encryption) to encrypt the RDS data without any additional cost.

CloudTrail – cloudTrails is enable on the AWS account by default – however sending the cloudTrail data to S3 bucket is NOT enable by default. By default , for 90 days cloudTrail will store the data beyond that if one needs to store the cloudTrail data then they needs to store it in S3 bucket.

CloudTrail can be integrated with CloudWatch which can also be used for raising Cloud Watch event based on Cloud Trail events.

CloudTrail can be applied to a single region (specified) OR can be applied to all region.

NOTE: to store cloud Trail logs from all region to a single S3 bucket – one needs to create a cloudTrail in one region and then apply that to all region – this will store the cloudTrail logs from all region into the S3 bucket specified.

Advantage of create cloudTrail and apply it for all region are

- There is low risk of missing CloudTrail configuration for the new region.

- Standard configuration for all region

- CloudTrail logs generated from all region can be sent to a single SNS topic, Centralized Notification.

There is a limit of 5 CloudTrail per region. If one cloudTrail is apply it for all region then it would be counted a 1 trail per region.

Default Option for CloudTrail API and CLI is a single trail per region, while CloudTrail created from AWS Console is single CloudTrail for all region.

One can configure a cloudTrial that can send logs to any S3 bucket in any region.

There are two types of events in CloudTrail – **Data Event** (one needs to enable it specifically) and **Management Event** (log by default).

CloudTrail logs send to S3 bucket are encrypted by default – one add MFA for additional delete protection.

How to enable cloudTrail from different AWS account to log in a single bucket ?

- Create a S3 bucket and add bucket policy to allow cross-account to upload data into the bucket.
- Configure cloudTrail to send logs to the S3 bucket for which the cross-account access has been turned on.

How to grant access to different account administrators to read CloudTrail logs from a centralized S3 bucket?

Create an IAM role for each account to read CloudTrail logs, administrators need to programmatically needs to assume role to read the log file(s).

How to grant access to third party auditor to read CloudTrail logs from a centralized S3 bucket?

Create a trust policy for the third-party AWS account, allowing them to assume role from the account where centralized S3 bucket is located. Create an IAM role attaching detailed policies defining what is/isn't allowed. Third party user need to assume the role to access the S3 Bucket

CloudTrail can be configure to CloudWatch – CloudWatch event metric filter can be applied to trigger CloudWatch Alarm. [*For example; how any times create EC2 instance API was called in last 24 hours*]

CloudWatch Best Practice

1. Limit the access to the cloudTrail stored logs. READ-ONLY access needs to be share to the users/administrators who wants to read or analyzed the logs.
2. Configure MAF delete as additional security to prevent accidental delete.
3. Configure SNS events for the configuration changes for the centralized CloudTrail S3 bucket.
4. Configure S3 lifecycle policy to reduce storage cost by archiving OLD logs.
5. For accessing cloudTrail from VPC use VPC Interface Endpoint for CloudTrail (this will avoid connecting CloudTrail from VPC over one internet instead the request will be routed within AWS infrastructure.)
6. Enable Integrity validation for the cloudTrail logs send to S3 bucket – this will ensure that logs are NOT tempered to remove evidence – this is done using industry standard SHA-256.

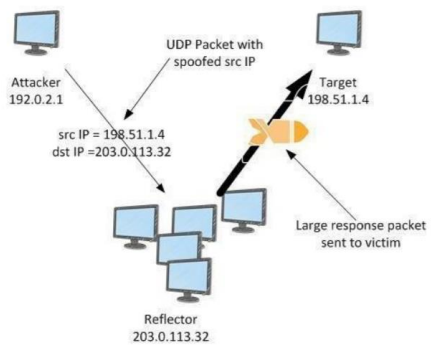
How CloudTrail logs integrity validation is achieved?

- Cloud Trail Create a Hash code for all the files it delivers to the S3 bucket.
- Every hour cloudTrail creates a digest file which contains the Hash keys of all the files it generated in that hour.
- CloudTrail then sign the digest file using a private key of public-private keypair and delivers the digest file to the S3 bucket.
- Each digest file will have signature of the previous digest file.
- The current digest file signature will be in the metadata properties of the S3 digest file Amazon S3 object.
- Log files and digest files can be stored in separate folders.
- AWS CLI can validate the digest files stored on the origin folder – incase the log files folders are changed, then one needs to use third party tool to validate the integrity of the log files.

DDoS (Distribute Denial of service) – DDoS attacks are operated at layer 3, 4 and 6, 7. The one that operate at layer 3 and 4 are called as Infrastructure layer attacks and the one that operates at layer 6 and 7 are called as application layer attacks.

Infrastructure Layer Attacks.

UDP reflection attacks – In case of UDP reflection attack, the attacker spoof the UDP packets by changing srcIP of the packet with the target IP, and send the request to the multiple systems (Reflector/mediator) , on receiving the requests the system will send their response to the srcIP with amplification of request of the UDP packet(s) which is the target's IP address.



TCP SYN Flooding attacks In a SYN flood attack, a malicious client sends a large number of SYN packets, but never sends the final ACK packets to complete the handshakes. The server is left waiting for a response to the half-open TCP connections and eventually runs out of capacity to accept new TCP connections.

WordPress XML-RPC attack (also known as WordPress pingback flood) – in this attack attacker misused XML-RPC function of the sites hosted on WordPress, the XML-RPC function of WordPress sites helps sites to ping back another site hosted on WordPress once link is generated.

Application Layer Attack:

HTTP Flood Attack: In this types of attack specific HTTP request is targeted OR in case of a more complex HTTP Flooding attack, it emulates a human interaction pattern with the application with an intend to outrun the server resource capacity. These attacks are difficult to track using request limiting.

Cache Busting attack: In this type of attack, a type of HTTP flood that force CDN not to cache objects, rather reach out to the origin server for each request overloading the origin server. This is achieved by creating variation in the query string.

DNS Query Flood Attacks: In this case the DNS queries are altered in a way that DNS need to forward the request to authoritative servers to resolve the DNS quires, in order to bring down or overload the DNS service.

TLS negotiation Attack web application is delivered over TLS, an attacker can also choose to attack the TLS negotiation process. TLS is computationally expensive so an attacker can reduce a server's availability by sending unintelligible data.

Mitigating DDoS

AWS default protect DDoS are followed free of charges. AWS offer multiple layer protection for DDoS attack. User can also use other AWS service to reduce the risk of DDoS attack.

Different techniques to minimized the DDoS attack

- Minimized the surface area
- Learn the normal behaviour
- Be ready to scale
- Safeguard expensive resources that are hard to scale.
- Have a plan to deal with DDoS

Additionally, you can leverage AWS services that operate from edge locations, like Amazon CloudFront and Amazon Route 53, to build comprehensive availability protection against all known infrastructure layer attacks. IT helped in improving DDoS resilience of the application when the serve web application traffic from edge locations distributed around the world.

Several specific benefits of using Amazon CloudFront and Amazon Route 53 include the following:

- AWS Shield DDoS mitigation systems that are integrated with AWS edge services, reducing time-to-mitigate from minutes to sub-second.
- Stateless SYN Flood mitigation techniques that proxy and verify incoming connections before passing them to the protected service.
- Automatic traffic engineering systems that can disperse or isolate the impact of large volumetric DDoS attacks.
- Application layer defense when combined with AWS WAF that does not require changing your current application architecture (for example, in an AWS Region or on-premises datacenter).

Infrastructure Layer Defense from DDoS Attack.

- Scale up horizontal or vertically to address the additional capacity need to mitigate the DDoS attack. Upgrade Instance size / enable enhance networking for high speed network throughput. Add more instances.
- Classic Loadbalancer or Application Loadbalancer does not support UDP traffic, which can help from mitigating UPD related attacks.
- Instead of using own DNS service and securing it with 3rd party software – use Route53 where possible.
- CloudFront can limit the attack, by preventing malform request from reaching to the origin server.

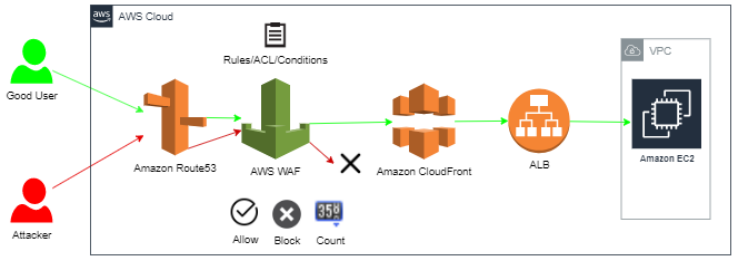
Infrastructure Application Defense from DDoS Attack.

- Using CloudFront distribution to reduce malicious request.
- Using WAF in CloudFront can help in filtering out malicious request.
- [scale of absorb] Autoscaling group can help in absorbing sudden spikes in the network-traffic.
- [reduce attack surface] eliminate un necessary entry points.
- [reduce attack surface] for necessary entry point *obfuscate* them. (*obfuscate = make obscure, unclear, or unintelligible.*)

Scalable, Secure, well monitored, DDoS protected applications on AWS

- Use globally distributed services like CloudFront and Rout53 which has built in protection for L3 and L4 – DDoS attacks.
- Use Elastic Loadbalancer to prevent UDP packets from reaching to the web applications. ONLY well form packets are allowed.
- Build scalable application to absorb DDoS attacks.
- Hide instances from internet.
- Well monitor your applications.
- Use WFA to baseline normal traffic and protect it against DDoS attacks.

Web Application Firewall



AWS WAF controls how an **Amazon CloudFront distribution**, an **Amazon API Gateway API**, or an **Application Load Balancer** responds to web requests.

Web ACLs – You use a web access control list (ACL) to protect a set of AWS resources. You create a web ACL and define its protection strategy by adding rules. Rules define criteria for inspecting web requests and specify how to handle requests that match the criteria. You set a default action for the web ACL that indicates whether to block or allow through those requests that pass the rules inspections.

Rules – Each rule contains a statement that defines the inspection criteria, and an action to take if a web request meets the criteria. When a web request meets the criteria, that's a match. You can use rules to block matching requests or to allow matching requests through. One can also use rules just to count matching requests.

Rules groups – You can use rules individually or in reusable rule groups. AWS Managed Rules and AWS Marketplace sellers provide managed rule groups for your use. You can also define your own rule groups.

Condition – Group Conditions into rules: There can be three possible outcomes – ALLOW, BLOCK and COUNT.

- ALLOW – https request
- BLOCK – https request
- COUNT – increment request

Rate based rules can be used to set threshold for any specific IP or all Ips – for example all request from a specific ip will be blocked if more than 2000 request per min (threshold based rules).

For WAF pay only for the web ACLs and rule groups that you create, and for the number of HTTP(S) requests that AWS WAF inspects.

NOTE: famous **XSS (Cross Site Scripting)** & **SQL injection** are blocked by WAF.

CloudFront and WAF are running on the edge location.
WAF can be integrated with internet facing ALB as wells as internal ALBs.

Third party WAF – the request is not automatically logged by CloudTrail as they are not from AWS ecosystem.

One can build WAF solution using third party WAF solution available on the AWS market place. Using a WAF sandwich Architecture – where WFA EC2 instance is placed on the DMZ, sandwich between two ELBs.

Intrusion Detection and Intrusion Protection:

Intrusion Detection: Is a device or a software that monitors an infrastructure /network or system for malicious activity or policy violation. Any policy violation or malicious activity are reported to administrator OR collected centrally & send to Security Information and Event Management (SIEM) system in Security Operation Control (Security Operation Control).
There are two types of IDS

- Host base IDS (HIDS) – IDS solutions that are installed on a host (EC2 instance) are called as HIDS.
- Network base IDS (NIDS) – IDNS solution that analyze the network traffic are called as NIDS.

IPS – intrusion protection system: are the system that stop any malicious traffic.
Promiscuous Port/Network tap/Sniffing are NOT allowed in AWS, even for owned EC2 instance network tapping is not allowed.

Different option of deploying IDS/IPS in AWS VPC

- Deploying in the AWS host – each of the ec2 instance will have a IDS agent installed which will send packets information to a single instance for analysis. This is a costly architecture as the host needs extra processing capacity for running IDS agents.
- Deploying on NAT instance – install IDS and IPS in NAT instance
- Deploying in a sandwich model between internet facing ELB and internal ELB.

AWS RAM (Resource Access Manager)

- AWS RAM allows sharing of the AWS resource with any AWS account or throughout AWS Organization. There are some resources which can be share within AWS Organization and some Resources that can be shared within AWS account – not all resources can be shared with AWS account as well as AWS organization.
- For multi-account organization resources – can be created centrally and AWS RAM can be used to share those resources with/among the accounts.
- There is no additional cost of creating and sharing resources centrally. (Charges are applicable as per the resource it creates).
- AWS RAM can be access from – RAM console, Query API, AWS CLI, and AWS tool for power shell.
- When a resource is shared with another account, then the account is granted access to the resources – any policies and permission in that account applies to the share resources. (for example : Account A share a resource with Account B – then IAM policies / SCP (service control polices) of account B is applicable to the resource in addition to the granted permission by RAM.
- For sharing a resource with RAM – when a resource is created, principle with whom to share are specified – a principle can be an AWS account, organization unit, or entire organization with from AWS organization.

- The account that share the resource retains the full ownership of the resource.

Benefit of RAM

- Reduce operational overhead – centrally created and shared with other , reducing the overhead of creating & provisioning the resources in each of the account.
- Provide Security & Consistency – Centrally created polices and permission help in improving security and consistence.
- Provide visibility and auditability – integrated with CloudWatch and CloudTrail.

What can be shared with RAM

- VPC Sharing allows multiple AWS accounts to create their application resources such as EC2 instance, Amazon RDS database, Amazon Redshift cluster, AWS Lambda functions, etc.
- Account can share – subnets with other organizations with AWS organizations - participants can view, create, modified, delete resources that they create – but doesn't have access/view to resources that are created by other participants or VPC owner.
- Transit Gateway can be shared through RAM with different account or with organization within once's AWS organizations.
- Route 53 forwarding rule can be shared through
- License Manager - License configuration is shared with different accounts.

AWS RAM – AZ ID For sharing load constantly across different accounts the AZ name are NOT consistent. Thus, one need to use AZ ID instead of AZ name.

AWS RAM can be configured with CloudWatch and CloudTrail for monitoring and compliance prospective.

Section 18: Application Integration

[Revision topics]

SNS (Simple Notification Services) – push messaging managed service from AWS.

- Reliability – each of the messages are stored in three separate AZ within the same region. SNS is a regional service. Provide high reliability within a region.
- Policies can be attached to the SNS topic to decide who can and who cant send the messages to the queue.
- Secure communication – API call made to SNS needs AWS ID and signature, and its recommended to send over https (secure channel).
- Authentication – request consist of AWS Secret key and the messages are signed using AWS Secret Key.

Mobile Push notification using SNS – read through SAA notes

Mobile Push notification direct messaging – precise messaging to a single subscriber. Unique messages to a each subscriber.

SNS integrates with CloudTrail for auditing.

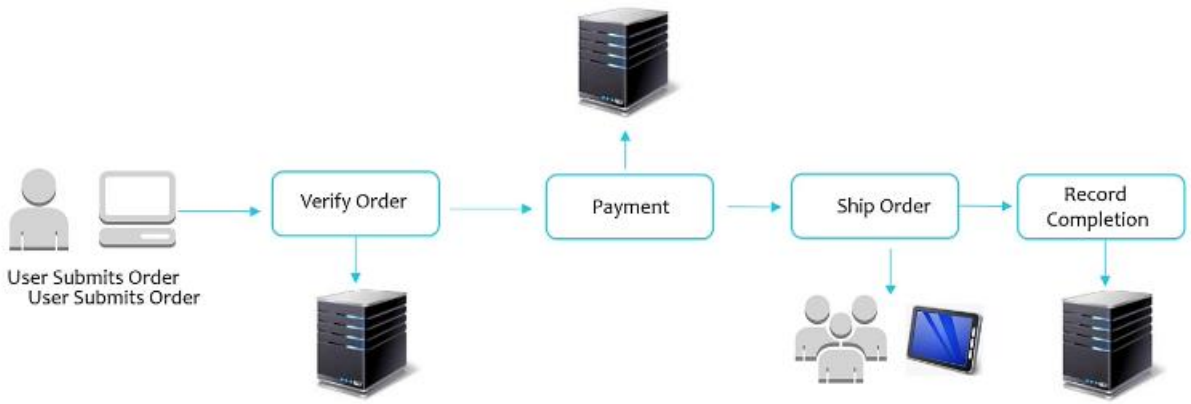
SQS (Simple Queue Service) – pull messaging managed service.

Transcoder sample.

- Short polling Vs Long polling interval.
- Retention period: time message is retained in the queue.
- Visibility timeout: time after which the message re-appear in the queue, allowing it to reprocess the fallout messages.
- Message encryption – server-side encryption, only message body is encrypted NOT the message metadata.
- 1-min monitoring (detailed monitoring) is not available for SQS, default monitoring (5min) is enable by default and comes with free of cost.
- Dead letter queue – failure queue.
- Default ONLY the owner of the account can send/received message to queue.

Mechanical Turk - Amazon marketplace MTurk: On demand manual processing virtual workspace of the pre-defined task.

Simple Workflow – SWF – Complex workflow management



Step function – state machine

Step function are defined in using amazon state language. It's easy to define multiple step application, each of the define state will have a different color. There are two types of task – ACTIVITY TASK and SERVICE TASKS. A task in step function context is build block of the step function, a task perform work by using an activity or an AWS lambda function or by-passing parameters to the API action of another service.

Activity worker can be running anywhere that can be communicated via HTTP, or can be running in EC2 instance, a mobile device, or in premises. A step function can pool work from another step function or does work for another step function or return results to another step functions.

SERVICE TASK – this task helps state function to connect to another AWS services, it can push the task to another AWS service and wait for the service to complete the task and return the control back to state function for it to flow to the next action. Using service task, state function can be connected to the following AWS services

- AWS lambda functions
- Can run ECS or AWS fargate tasks
- Get and Put item from/to Amazon Dynamo DB
- Submit AWS batch jobs, wait for it complete
- Publish a message to SNS topic
- Send a message to SQS queue
- Start a Glue Job
- Start a StageMaker job to train a machine learning model or batch transform a data set
- Have state function wait for a specific task to return a specific task token.

Step functions can be used for

- Data science – data processing
- DevOps and IT automation
- E-Commerce applications
- Web-applications

SWF	**Step Function**
Fully managed	Fully managed
Code needs to be written for decided task	Code to be written in Amazon State Language
Its more suited for - when there are external signaling required in the workflow. - required to launch child task	Mostly suite for all orchestration needs
Can be integrated with CloudTrail and CloudWatch	Can be integrated with CloudTrail and CloudWatch

Section 19: AWS Machine Learning

Amazon Rekognition : It's service to add image & video base visual analysis to client applications. There are two separate API from video and from image.

Common use case for Amazon Rekognition

1. **Searchable video and image libraries:** Amazon Rekognition make image and videos searchable.
2. **Face base user verification:** face authentication, image to reference image comparison.
3. **Sentiment and demographic analysis** gender analysis, sentiment analysis.
4. **Facial Recognition**
5. **Unsafe content detection** visual /image base tagging**.**
6. **Celebrity recognition**

Amazon Rekognition API can do the following

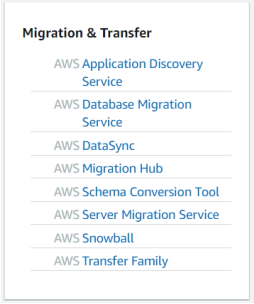
- Detect Labels
  - To detect labels in images, use api – detectLabels
  - To detect labels in videos, use api startLabelDetection
- Detect Faces
  - To detect faces, use API detectFaces
  - To detect faces, use API startFaceDetection
  - Face search can deal with streaming vedios – CreateStreamProcessor
- PeoplePath
  - It can detect person in store videos – using startPersonTracking API
- Celebrities
  - It can detect 1000's of celebrities in stored videos or images
  - Use API recognizedCelebrities API for detecting celebrity in stored images
  - Use API startCelebrityRecognition to detect celebrity in store videos
- Text Detection
  - Use detectText to detect text from an image.
- UnsafeContent
  - Use detectModerationLabels to detect unsafe content in images.
  - Use startContentModeration to detect unsafe content in videos.

Amazon Rekognition can Analyze images that are supplied as image bytes or images that are store in S3 bucket. If store in S3 bucket it needs to be .png or .jpg format, if supplied to API directly then the images should be in base64 encoded.

Amazon Rekognition can Analyze videos that are either stored in S3 bucket or streamed through Kinesis video streaming. It's an asynchronous process, it starts with calling startFaceDetection API once completed Amazon Rekognition will publish the completion status in Amazon SNS topic.

Amazon Rekognition video – video operation can read input data from kinesis video stream and publish its output in kinesis data stream.

Section 20: Hybrid Cloud & Data Migration Solution



Application Data Discovery Service

Application Discovery Service: It's an AWS service that helps in planning migration of AWS cloud by collecting usages and configuration data about on-premises servers.

AWS Migration hub performs three prominent task that are performed by AWS Application Discovery Services are :

- Discoveries of physical and virtual hardware machine using Application Discovery Services
- Grouping of services based on the application functionalities
- Migration & tracking of the migration progress for each service.

Discovery information can be exported into MS Excel or Athena

How Application Discovery Service works?

- **Agentless discovery Connector – VM Ware / vCenter**
    - This can perform by deploying an AWS Agentless discovery connector (OVA file), through the on premises vmware vCenter.
    - After AWS Agentless discovery connector is configure, appropriate right is provided – it identifies virtual machine (VMs) and host associated with vCenter
        - Static information collected by AWS discovery connector – server hostname, IP addresses, MAC Addresses, disk resource allocation.
        - Additional information collected are – utilization data for each of the connected VM, average and peak utilization metrics for CPU, RAM, Disk I/O
    - Collected data is sent to Application Discovery services using Secure Socket Layer (SSL) encryption.
- **Agent Base discovery connector –** for detailed information discovery like details of the processes running on each VMs or the network connectivity between each VMs then one need to use agent base discovery connectors. The agent connector are available of both Lunix and windows VMs, once installed on the VMs it send the collected information to Application Discovery Services using Secure Socket Layer (SSL) encryption. With agent-based discoveries – one can collect static configuration details, detailed time series system performance information, inbound and outbound network connections, list of processes that are running on each VMs.
  NOTE: VMs hosted on vmware can run agent base and agentless discovery connectors simultaneously.

Agentless discovery connector	Agent base discovery connector
VMs ONLY	Can be used for VMs and physical servers
Per vCenter	Per VMs

ONLY static data can be collected – like servername, IP address, MAC addresses, disk allocations	VM utilization metrics Timed performance metrics Network input/output Connection.
ONLY for VM hosted OS	Windows and Lunix

Agent base connector doesn't support a standalone ESX host (*VM with ESXi Hypervisor*), ESX host must be a part of vCenter server instance than it can be discovery by agentless discovery connector.

Data Exploration:
Data exploration in Amazon Athena allows the analysis of the data collected by the discovery connector.

Once the data exploration in Amazon Athena is turned on through AWS Migration Hub console or through StartContiniousExport API & the data collection is turned on – discovery connectors will get the data automatically stored in S3 bucket on which Amazon Athena can run predefine quires against the stored data to analyze the time series system performance for each servers, the type of process that are running on the VM/servers and the network dependencies between different servers.

Also, one can configure custom queries that are build to run on Athena can be used to create configuration management databases in associated with the discovered servers with actual business applications.

Athena database results can be visualized using AWS QucikSight for better clarity.

Pending - https://aws.amazon.com/server-migration-service/
https://aws.amazon.com/dms/

Migration Hub: Application discovery service integrates with the migration hub, for storing and analyzing the data collected – on need to define the migration hub home region in-order for application discovery service to send their collected data.
After the discovery of on-premises server, migration hub helps in grouping those servers w.r.t applications and then track the migration progress for each individual application. Collected discovery data can be exported to MS Excel or can be analyze using Athena and QuickSight.

There are three ways to discover a on-premises server through AWS migration hub

1. Using AWS migration hub import – allows import details of your on-premises environment directly into Migration Hub without using the Discovery Connector or Discovery Agent. Download/generate csv file populate with on-premises server data.

```
aws discovery start-import-task --import-url s3://BucketName/ImportFile.csv --name
ImportName
```

2. Using Agent less application discovery connector
3. Using Agent base application discovery connector

Migration hub has a built-in logic to match fields for identify the server instances and group then into separate applications.
The following fields are used to match servers when discovery tools are used.

- ExternalId – This is the primary field used to match servers. If the value in this field is identical to another ExternalId in another entry, then Application Discovery Service matches the two entries, regardless of whether the other fields match or not.
- IPAddress
- HostName
- MacAddress

- VMware.MoRefId and VMware.vCenterId – Both of these values must be identical to the respective fields in another entry for Application Discovery Service to perform a match.

AWS Server Migration Service

AWS Server Migration Service automates the migration of your on-premises VMware vSphere, Microsoft Hyper-V/SCVMM, and Azure virtual machines to the AWS Cloud. AWS SMS incrementally replicates your server VMs as cloud-hosted Amazon Machine Images (AMIs) ready for deployment on Amazon EC2. Working with AMIs, you can easily test and update your cloud-based images before deploying them in production.

Benefits of using AWS SMS

- **Simplify the cloud migration process:** Once the migration is trigger it manages the complexities of the automatic replication of the live volume, periodic creation of the AMIs
- **Orchestrate multiple server migration:** It can migrate and tracks the progress of group of servers group within an application.
- **Test server migration incrementally:** Allows to replication of the incremental changes, thus save band-width and time – one can test iteratively the same changes.
- **Support mostly wildly used OS**
- **Minimized downtime**
- **There is no-additional charge for using SMS (Server Migration Service).**
- **SNS can be configure to notify list of recipients once the replication job is completed.**
- **AWS SMS can generate encrypted AMIs – default CMK or non-default key can be specified.**

Limits for migrating servers through SMS

- 50 concurrent VM migrations per account, unless a customer requests a limit increase.
- 90 days of service usage per VM (not per account), beginning with the initial replication of a VM. We terminate an ongoing replication after 90 days unless a customer requests a limit increase.
- 50 concurrent application migrations per account, with a limit of 10 groups and 50 servers in each application.
- After migration VM can have only one ENI (single virtual interface) and that uses DHCP address
- Launch VM will received a private IP address when created – later one can add EIP if needed.
- IPv6 are not supported
- AMIs with more than 22 volumes are not supported.
- SMIS with encrypted EBS volumes are not supported (encrypted AMI are supported).

All major windows and LUNIX OS are supported by AWS SMS.

Limitation of SMS

- Image Format – SMS does not support migration of raw device mapping ONLY VMDK disk migration is supported.
- File System – ONLY 64bit image are supported, does not support 32-bit image / default kernel advice custom kernel may NOT work properly.
- Booting – dual booting instances are not supported by SMS, A migrated VM might fail to boot if the root partition is not on the same virtual hard disk as the MBR.
- Networking - Multiple network interfaces are not currently supported. After migration, your VM will have a single virtual network interface that uses DHCP to assign addresses. Your instance receives a private IP address. IPV6 is not supported.
- Volume – VM with more than 22 volume attached cannot be migrated
- AMIs with volumes using EBS encryption are not supported. When migrating servers using AWS SMS, do not turn on encryption by default. If encryption by default is already on and you are experiencing delta replication failures, turn off this feature.
- AWS SMS does not install the single root I/O virtualization (SR-IOV) drivers except with imports of Microsoft Windows Server 2012 R2 VMs. These drivers are not required unless you plan to use enhanced networking,

which provides higher performance (packets per second), lower latency, and lower jitter. For Microsoft Windows Server 2012 R2 VMs, SR-IOV drivers are automatically installed as a part of the migration process.

Licensing options for SMS

- Auto(default) – Detect the source system OS and apply appropriate licenses to migrate the VM.
- AWS – Replace the source system license with AWS license.
- BYOL – Bring your own license.

Replicating VMs Using the AWS SMS Console

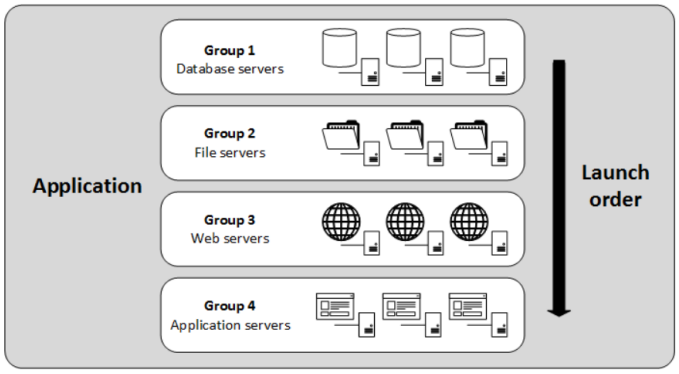
Use the AWS SMS console to import your server catalog and migrate your on-premises servers to Amazon EC2. During the replication process, AWS SMS creates an Amazon S3 bucket in the Region on your behalf, with server-side encryption enabled and a bucket policy to delete any items in the bucket after seven days. AWS SMS replicates server volumes from your environment to this bucket and then creates EBS snapshots from the volumes.

Replicating a Server

AWS SMS automatically replicates live server volumes to AWS and creates an Amazon Machine Image (AMI) as needed. Shedule job or on demand job can be run to migrate live server

Migrating Application using SMS

AWS Server Migration Service supports the automated migration of multi-server application stacks from your on-premises data center to Amazon EC2. Where server migration is accomplished by replicating a single server as an Amazon Machine Image (AMI), application migration replicates all of the servers in an application as AMIs and generates an AWS CloudFormation template to launch them in a coordinated fashion.



Application migration relies on Application discovery connector to discover on-premises resources and group them in groups, one can also manually add/remove servers to a group or create/delete groups. After these on-premises resources discoveries are imported into SMS as server catalogue one can configure the setting for the applications, replication and launch, as well can also monitor the migration status. Maximum migration replication job can be keep running till 90days from the day it was trigger, after 90 days the replication job will be automatically terminated.

NOTE: Migration Hub cater for both physical servicer and virtual servers, however SMS ONLY caters for virtual servers.

SMS imports application related servers from migration hub ONLY if the servers exist in SMS Catalogue. If none of the servers mention in migration hub application exists on the SMS Service Catalogue then the import will fail silently and the application will NOT be visible is SMS.

Migration hub imported application can be imported into SMS, but they can't be edited. The imported application can ONLY be edited in migration hub.

If integration between migration hub and SMS is established then the SMS catalogue can also be visible on the migration hub.

AWS CloudWatch can be integrated with AWS Server Migration Service (SMS), to trigger automated action (AWS Lambda) based on the migration workflow milestones. For example – once the SMS finishes the creation of the AMI, AWS lambda function can be triggered to launch an EC2 instance based on the AMI created.

All calls to SMS API are logged in CloudTrail.

AWS Data migration Service

AWS service that facilitates the migration of data from/to AWS cloud, it supports varieties of instances – like Relational Databases, NO-SQL databases, Data Warehouse and others. It supports both migration of like-to-like db migration as well as it can migrate databases into RDS, Redshift, DynamoDB or S3 for storage.

NOTE: The only requirement to use AWS DMS is that one of your endpoints must be on an AWS service. You can't use AWS DMS to migrate from an on-premises database to another on-premises database.

AWS DMS supports one-time migration of databases or on-going replication of databases to keep source and target databases in sync.

The data migrated through AWS DMS are encrypted at transit as well as at rest.

AWS DMS Replication instance – is an EC2 instance that facilitates the migration of data from source to target (homogeneous or heterogeneous) databases.

AWS DMS main components are

1. Replication instance
2. Replication task – one can schedule replication task, more than one replication task can be run at a given time.
3. Endpoints

NOTE: 50 or 100 GB migration space (space in replication instance / EC2 instance space) options are available, one can request for more space if required. All buffering data , cache and logs are getting written to this space if the rate at which source db is read does not match to the source at which target DB are read.

Migration Service allows manual and automatic replications, in case of manual migration user need to identify and creates the target schemas on their own, while in case of automated migration AWS DMS with the help of tools like schema creation tool (SCT) identify and creates target schema. AWS DMS does not perform schema/ code conversion – it's been done by the SCT. If SCT is used for migration, any dependencies between tables such as foreign key constraints need to be disabled during the migration's "full load" and "cached change apply" (CCA) phases. If performance is an issue, removing or disabling secondary indexes during the migration process helps. For more information on the AWS SCT

In case of the homogeneous DB migration, it's advisable to use DB native tool for migrating schema, data can be migrated by DMS, in case of heterogeneous DB migration SCT (Schema Conversion Tool) can help to migrate the schema.

For migrating Casandra DB to Dynamo DB

As DMS does not support Casandra as one of the source/targets supporting endpoint – on need to use SCT Schema Conversion Tool to perform the migration along with AWS DMS. The Casandra DB clone will be created to avoid disruption on the live instance, then migration agent reads the Casandra DB structure and then move it S3 bucket, from where it can be imported to Dynamo DB.

DMS - Replication Instance

It's one of the important components of DMS, it's an EC2 instance that host the replication task. More than one replication task can be hosted on replication instance.
Replication instance come with variety of instance types to choose from depending upon the requirement, its comes with two variation of base storage capacity 50GB and 100GB, depending upon special need this can be extended if required.

DMS infrastructure overview
DMS replication instance always run on client VPC, within a region. If multi-AZ option is selected then the replication instance will be running in two separate AZ within the VPC.
In case of DB migration from on-premises DB then, direct connect or VPN connection need to be established with the on-premises network and the DMS replication instance VPC.

Security Group and NACL of the DMS replication instance need to be configure such that it can connect to source and target DB instances.

Appropriate IAM role need to be attached to the replication instance in order to read and/or write to source/target DB.

DMS – Endpoint

Source endpoint – configuration to reach source DB.

Target endpoint – configuration to reach target DB.

This will vary based on the choice of the source/target DB instances. One endpoint can be used by more than on replication task.

DMS – Replication Task
DMS replication task moves the set of data from source endpoint to target endpoint. The following task setting needs to be configured to replicated task.

1. Replication Instance – ec2 instance to host replication task.
2. Source Endpoint
3. Target Endpoint
4. Migration Type options
5. Target Table preparation mode options: There are three modes of target table preparation
    a. Do-Nothing: AWS DMS assume that target table are already created & prepared.
    b. Drop-table on target: by selecting this option AWS DMS drop the tables & recreate then on the target before migrating the data into it.
    c. Truncate: In this option the target tables are truncated, before migrating the data into the target tables.
6. Large Object Binary (LOB) mode options: there are three mode available to migrate the large binary objects
    a. Don't include LOB: All LOB columns are excluded.
    b. Full LOB mode: All LOB columns are included; this is slower option as none of the LOB columns are excluded.
    c. Limited LOB mode: in this option, ONLY those LOB values are included that has LOB column size less than the set max-size of the LOB column.
7. Table mappings
8. Data transformation
9. Data validation
10. Amazon CloudWatch Logging configurations

Migration Type Options

- **Full Load (Migrating Existing data):** With this option, data from source DB to target DB , where the source DB can be made offline while migration is in place. DMS creates only those objects that are required for the effective migration for example it creates tables, primary keys, and in some cases unique indexes, but doesn't create any other objects that are not required to efficiently migrate the data from the source. For example, it doesn't create secondary indexes, nonprimary key constraints, or data defaults.
- **Full Load + Change Data Capture [CDC] (Migrate existing data and replicate ongoing changes)**: It's a full load migration while capturing the on-going changes on the database post migration.
- **Change Data Capture [CDC] Only (Replicate Data change ONLY)**: Where existing data is migrated using other tool, post migration replication can be achieved through AWS DMS.
  For homogeneous database migration – AWS recommended to use native DB tools to migrate the existing data, and enable CDC only potion to replicate changes.

AWS DMS Supported source/Target

DB or Datasource	Source for DMS	Source Located at	Target for DMS	Target Located at
MS Azure SQL	YES	MS Azure	**NO**	
MongoDB	YES	On premises | EC2 instance.	**NO**	
DB2	YES	On premises | EC2 instance.	**NO**	
RedShift	**NO**		YES	AWS RDS
Elastic Search	**NO**		YES	AWS
Kinesis Data Stream	**NO**		YES	AWS
DynamoDB (w/mongoDB compatibility)	**NO**		YES	AWS
DocumentDB	**NO**		NO	AWS
Amazon S3	YES	AWS	YES	AWS
MariaDB (Supported as mySQL compatible)	YES	On premises | EC2 instance| AWS RDS	YES	On premises | EC2 instance | AWS RDS
AuroraDB	YES	RDS	YES	RDS
Oracle DB	YES	On premises | EC2 instance | RDS	YES	On premises | EC2 instance | RDS
MySQL	YES	On premises | EC2 instance RDS	YES	On premises | EC2 instance | RDS
MS SQL Server	YES	On premises | EC2 instance | RDS	YES	On premises | EC2 instance | RDS
PostgresSQL	YES	On premises | EC2 instance | RDS	YES	On premises | EC2 instance | RDS
SAP Adaptive Server Enterprise (ASE)	YES	On premises | EC2 instance.	YES	On premises | EC2 instance.

Migration of MongoDB – to on premise OR to EC2 instance.

There are two possible mode of migration w.r.t mongoDB

- **Document Mode (DMS default mode)**: In document mode, the mongoDB document are migrated as – it can be migrated into DynamoDB or DocumentDB
- **Table Mode:** In table mode, the mogoDB top level fields in mongoDB document into a column in the target table.

Integration of AWS DMS with other AWS Service

- Target DB can be in EC2 instance or in RDS
- Can be integrated with SCT to convert source schema and SQL code into target schema and SQL code.
- When migrating large data, DMS can be integrated with S3 for temporary storage.
- AWS CloudFormation template can be created to set up migration infrastructure – DMS replication instance, replication task, certificates, endpoints, target RDS instance or EC2 instance to host target RDS instance.

Storage Gateway – Refer SAA notes
Snowball / Snowball edge – Refer SAA notes
Refer below for calculating time required for migrating the data from on-premises to cloud.
[1Gbps = 1/8 GBps (0.125 GBps)]
[1 Mbps = 1/8 MBps (0.125 MBps)]

Snowball	Snowball Edge	Snowmobile
50 TB (42TB) – outside us 80 TB (72TB) – within us	100 TB with compute (EC2 or Lambda)	Exabyte migration
	Durable storage	

Use case	Snowball	Snowball Edge
Import data into Amazon S3	✓	✓
Export from Amazon S3	✓	✓
Durable local storage		✓
Local compute with AWS Lambda		✓
Amazon EC2 compute instances		✓
Use in a cluster of devices		✓
Use with AWS IoT Greengrass (IoT)		✓
Transfer files through NFS with a GUI		✓

Storage capacity (usable capacity)	Snowball	Snowball Edge
50 TB (42 TB) - US regions only	✓	
80 TB (72 TB)	✓	
100 TB (83 TB)		✓
100 TB Clustered (45 TB per node)		✓