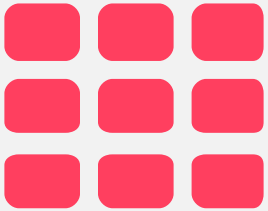

aws
certified
Solutions
Architect
Professional
DOLFINed



AWS STORAGE OPTIONS

AMAZON SIMPLE STORAGE SERVICE (S3)

Amazon S3

S3 Introduction, Data Consistency Models, Buckets and Objects

Review Topic : Simple Storage Service (S3)

S3

- S3 is a storage for the internet. It has a simple web services interface for simple storing & retrieval of any amount of data, anytime, from anywhere on the internet
- S3 is Object-based storage, and NOT a block storage
- S3 has a distributed data-store architecture, where objects are redundantly stored in multiple locations

S3 Consistency Levels

- **S3 Provides :**
  - Read-after-write (Immediate or Strong) Consistency of PUTs of new objects (new object loads to S3)
    - A PUT is an HTTP request to store the enclosed data (with the request)
  - Eventual Consistency for overwrite PUTs and DELETEs (for Changes/updates to existing Objects in S3)
- Updates to an Object are atomic, i.e when you run a PUT for an object then you read (GET) that object, you will either get the updated object or the old one (before the update), you will never get partial or corrupt data

Review Topic : Simple Storage Service (S3)

S3 Buckets

- A bucket can be viewed as a container for objects
- A bucket is a flat container of objects
  - It does not provide any hierarchical of objects (actual folders)
    - You can create folders in your bucket (available through Console only)
  - You can use object key (name) to mimic folders in a bucket when using the AWS console
- You can store unlimited objects in your bucket, but an **Object can NOT exceed 5 TB**
- You can NOT create nested buckets (a bucket inside another)
- Bucket ownership is not transferrable
- An S3 bucket is region specific
- For better performance, lower latency, and to minimize costs, create the S3 bucket closer to you client DC (or source of data to be stored)
- S3 Bucket names (Keys) are globally unique across all AWS regions
  - Buckets names can NOT be changed after they are created
- Bucket names are part of the URL used to access a bucket
  - https://<bucketname>.s3.<region>.amazonaws.com/
- The bucket name is simply two parts:

  Example, for S3 bucket named cloudbucket1 in Europe West Region (Ireland)

  https://s3-eu-west-1.amazonaws.com/cloudbucket1

Review Topic : Simple Storage Service (S3)

S3 Buckets – Sub-resources

- Sub-resources (configuration containers) for S3 buckets include:
  - Lifecycle : To decide on objects' lifecycle management
  - Website : to hold configurations related to static website hosted in S3 bucket
  - Versioning: Keep object versions as it changes (gets updated)
  - Access Control Lists (ACLs)
  - Bucket policies
  - Cross Origin Resource Sharing (CORS)
  - Logging
  - Event Notification
  - Transfer Acceleration
  - Request Payment
  - Replication
  - Object Locking

For a full list, visit

https://docs.aws.amazon.com/AmazonS3/latest/dev/UsingBucket.html#bucket-config-options-intro

Review Topic : Simple Storage Service (S3)

S3 Objects

- An object size stored in an S3 bucket can be **Zero bytes (0 bytes) up to 5 TB**
- Each object is stored and retrieved by a unique Key (ID or name)
- An Object in AWS S3 is uniquely identified and addressed through:
  - Service end point
  - Bucket name
  - Object Key (name)
  - Optionally, Object version
- Objects stored in a S3 bucket in a region **will NEVER leave that region** unless you specifically move them to another region, or enable Cross Region Replication
- S3 provides high data durability, Objects are redundantly stored on multiple devices across multiple facilities in an Amazon S3 region (where the bucket exists)

Amazon S3

- Bucket Versioning
- Copying/Uploading Objects and Multi-Part upload

Review Topic : Simple Storage Service (S3)

S3 Bucket Versioning

- Bucket versioning is a S3 bucket sub-resource used to protect against accidental object/data deletion or overwrites
  - Versioning can also be used for data retention and archive
- You will be charged for all S3 storage costs for all Object versions stored
  - You can use versioning with S3 lifecycle policies to delete older versions, OR, you can move them to a cheaper S3 storage (or Glacier)
- Once you enable Bucket versioning on a bucket, it can not be disabled, however, it can be suspended
- When enabled, bucket versioning will protect existing and new objects, and maintains their versions as they are updated (edited, written to…)
  - Updating objects refers to PUT, POST, COPY, DELETE actions on objects
- Only S3 bucket owner can permanently delete objects once versioning is enabled

Review Topic : Simple Storage Service (S3)

Bucket Versioning – MFA Delete

- Multi Factor Authentication (MFA) Delete is a versioning capability that adds another level of security in case your account is compromised

- This adds another layer of security for the following:
  - Changing your bucket's versioning state
  - Permanently deleting an object version

- MFA Delete requires:
  - Your security credentials
  - The code displayed on an approved physical or SW-based authentication device

- This provides maximum protection to your objects

Review Topic : Simple Storage Service (S3)

Multipart Upload

- Is used to upload an object (objects) in parts
  - Parts are uploaded independently and in parallel, in any order
- It is recommended for objects sizes of 100MB or larger
  - However, you can use for object sizes starting from 5MB up to 5TB
  - You must use it for Objects larger than 5GB
- This is done through the S3 Multipart upload API

Review Topic : Simple Storage Service (S3)

Copying S3 Objects (SDKs or REST API)

- The copy operation creates a copy of an object that is already stored in Amazon S3
- You can create a copy of your object up to 5 GB in size in a single atomic operation
- However, to copy an object greater than 5 GB, you must use the multipart upload API
- You will incur charges if the copy is to a destination that is another AWS region
- Can be done using AWS SDKs or REST API

- Use the copy operation to:
  - Generate additional copies of the objects
  - Renaming objects (copy to a new name)
  - Changing the copy's storage class or encrypt it at rest
  - Move objects across AWS locations/regions
  - Change object metadata
    - Once you upload an object to S3, you can NOT change its metadata, this is where the copy comes in handy
    - In this case, keep the same object key (name) for source and destination objects

Amazon S3

- Storage Classes

Storage Classes – Real time Frequent Access classes

- S3 STANDARD
  - performance-sensitive use cases (those that require millisecond access time) and frequently accessed data

- RRS – avoid not recommended
  - noncritical, reproducible data that can be stored with less redundancy

Storage Classes – Real time & Infrequent Access classes

- STANDARD_IA
  - long-lived and infrequently accessed data
  - Storing Backups
  - 128KB file size or bigger (you get charged for 128KB if you store smaller ones)
  - 30 days min charge/storage (you still can move it, delete it, but still will be charged for 30 days)
  - Primary or only copy of data that can't be recreated.

- ONEZONE_IA
  - Use if you can recreate the data if the Availability Zone fails, and for object replicas when setting cross-region replication (CRR).

Storage Classes – Intelligent Tiering

- It automatically moves data to the most cost-effective storage tier
- Is designed for storage cost optimization
- Use it to optimize storage costs automatically for long-lived data when access patterns are unknown.
- It does not cause performance impact or operational overhead.
- Is suitable for 128KB objects or larger
- For smaller objects, auto-tiering will not function.
- Min storage is 30 days
- It operates at a granular object level between and moves it between two access tiers, a frequent access tier and a lower-cost infrequent access tier (when the object is not accessed for 30 days), when access patterns change.
- A small monthly monitoring and automation fee applies per object.
- No retrieval fees
- No fees when objects are moved between the access tiers

Storage Classes – Archiving Classes – Non Real time access

- **Glacier** (Not for real time access of data)
  - Use for archives where portions of the data might need to be retrieved in minutes.
  - Minimum storage duration of 90 days
  - Can be accessed in as little as 1-5 minutes
  - You are charged for 90 days if you move the data before **the 90 days min storage duration**

- **DEEP_ARCHIVE** (Not for real time access of data)
  - Use for archiving data that rarely needs to be accessed.
  - Data stored in DEEP_ARCHIVE storage class has a **minimum storage duration period of 180 days**
  - Default retrieval time of 12 hours. If you have deleted, overwritten, or transitioned to a different storage class an object before the 180-day minimum, you are charged for 180 days.
  - DEEP_ARCHIVE is the lowest cost storage option in AWS.
  - Storage costs for DEEP_ARCHIVE are less expensive than using the GLACIER storage class.
  - You can reduce DEEP_ARCHIVE retrieval costs by using bulk retrieval, which returns data within 48 hours.

Glacier

- Is designed to sustain data loss in two facilities
- Glacier automatically encrypts data at rest using AES-256-bit symmetric keys and supports secure transfer of your data (In-transit) over Secure Sockets Layer (SSL)
- Glacier archives are visible and available only through AWS S3 not through AWS Glacier

- Glacier Retrieval Options:

Storage Class	Expedited	Standard	Bulk
GLACIER	1–5 minutes	3–5 hours	5–12 hours
DEEP_ARCHIVE	Not available	Within 12 hours	Within 48 hours

Storage Classes Comparison

https://docs.aws.amazon.com/AmazonS3/latest/dev/storage-class-intro.html

Storage Class	Designed for	Durability (designed for)	Availability (designed for)	Availability Zones	Min storage duration	Min billable object size	Other Considerations
STANDARD	Frequently accessed data	99.999999999%	99.99%	>= 3	None	None	None
STANDARD_IA	Long-lived, infrequently accessed data	99.999999999%	99.9%	>= 3	30 days	128 KB	Per GB retrieval fees apply.
INTELLIGENT_TIERING	Long-lived data with changing or unknown access patterns	99.999999999%	99.9%	>= 3	30 days	None	Monitoring and automation fees per object apply. No retrieval fees.
ONEZONE_IA	Long-lived, infrequently accessed, non-critical data	99.999999999%	99.5%	1	30 days	128 KB	Per GB retrieval fees apply. Not resilient to the loss of the Availability Zone.
GLACIER	Long-term data archiving with retrieval times ranging from minutes to hours	99.999999999%	99.99% (after you restore objects)	>= 3	90 days	None	Per GB retrieval fees apply. You must first restore archived objects before you can access them. For more information, see Restoring Archived Objects.
DEEP_ARCHIVE	Archiving rarely accessed data with a default retrieval time of 12 hours	99.999999999%	99.99% (after you restore objects)	>= 3	180 days	None	Per GB retrieval fees apply. You must first restore archived objects before you can access them. For more information, see Restoring Archived Objects.
RRS (Not recommended)	Frequently accessed, non-critical data	99.99%	99.99%	>= 3	None	None	None

Review Topic : Simple Storage Service (S3)

Glacier – Archiving Storage

- Glacier redundantly stores your data to multiple facilities, on multiple devices within each facility synchronously – You can receive SNS notification when upload is completed
  - Downloading archives is Asynchronous though
- You can upload an archive to Glacier of sizes from 1Byte to 40TB
  - Use multipart upload to load archives more than 5GB in size.
- Glacier awaits until the multiple facility upload is completed successfully before confirming a successful upload to the user
- Uploading archives is Synchronous while Downloading archives is Asynchronous
  - The contents of an archive, once uploaded, can not be updated
- Glacier does NOT archive object metadata, you need to maintain a client-side database to maintain this information about the objects (Basically which archives hold which objects)

Amazon S3

- Life Cycle Policies
- Server-Side Encryption

Review Topic : Simple Storage Service (S3)

S3 Life Cycle Policy

- It can be applied to certain objects in a bucket folder, objects with a specific tag, or objects with a specific prefix
- The purpose is to primarily perform desired actions on contents (or some) of the bucket
- You can configure two actions:
  - Transition actions: to another S3 storage tier after a configured period
  - Expiration actions: Where an expiration duration for object(s) is set, S3 will then delete the expired objects on your behalf

Review Topic : Simple Storage Service (S3)

S3 Server-Side Encryption (SSE)

- This is primarily about encrypting data at rest on S3 buckets

- There are two main ways to encrypt data stored on S3 buckets:
  - Client-side encryption:
    - Where the client (user or application) encrypts data on the Client side , then transfer the data encrypted to S3 buckets (hence data is encrypted In-transit and at Rest)

  - Server-Side Encryption (SSE):
    - Data is encrypted by the S3 service before it is saved to S3 storage disks
    - Data is decrypted when you download it

- User access to S3 bucket objects is the same whether the data is encrypted or not on S3 buckets (as long as the user has permission to access the data)

Review Topic : Simple Storage Service (S3)

S3 Server-Side Encryption using AWS S3 Managed Keys (SSE-S3)

- Each object is encrypted by a unique key, then the encryption key itself is encrypted using a master key
- S3 regularly rotates the master key
- Uses AES-256 bits

Review Topic : Simple Storage Service (S3)

S3 Server-Side Encryption using AWS KMS Managed Keys (SSE-KMS)

- KMS uses Customer Master Keys (CMKs) stored in KMS to encrypt your S3 objects
  - You can continue to use the automatically created default CMK key for encryption
  - OR, You can select a CMK that you created separately using AWS Key Management Service.
    - Creating your own CMK will allow you to create, rotate, disable, and define access controls,
    - KMS provides an audit trail for when the key was used and by whom
    - Also allows you to audit the encryption keys used to protect your data.
    - The CMK used and the bucket must be in the same region
- Separate permissions for an envelope key that protects/encrypts your object encryption keys
- This service is chargeable

Review Topic : Simple Storage Service (S3)

S3 Server-Side Encryption using Customer Provided Keys (SSE-C)

- Server-Side Encryption using Client provided keys
- Client manages the keys, S3 service manages encryption
- AWS does not store the client provided encryption key(s), so if the client looses the key(s), they can't access the object, basically they loose the data.
  - Instead, S3 stores a randomly slated HMAC (hash) of the key to validate future requests
  - Client must provide the key with every request
  - S3 validates the key using the HMAC, and if it matches, will decrypt the data using the provided key, then return the object to the Client

Amazon S3

Static WebSite Hosting & Redirection

Review Topic : Simple Storage Service (S3)

S3 Static Website Hosting

- S3 buckets can be used to host static content (not dynamic) websites
  - Static content refers to content that does not need to run server-side scripts such as PHP, JSP, or ASP.NET.
  - Content can be HTMP pages, images, videos, Client-side scripts such as Javascript
- AWS S3 Hosted websites scale automatically to meet demand
- You can use your own domain with S3 hosted static websites (Route53 CNAME/Alias)
- There is no additional charge for hosting static websites on S3
- S3 hosted static websites can enable redirection for the whole domain, pages within a domain, or specific objects.

Review Topic : Simple Storage Service (S3)

S3 Static Website Hosting

- The website S3 endpoint URL is :
  http://<S3bucketname>.S3-website-<AWS-Region>.amazonaws.com OR
  tp://<S3bucketname>.S3-website.<AWS-Region>.amazonaws.com

- Amazon website endpoints DO NOT support HTTPS

- You need to allow public read access to all the bucket used for hosting the static website.
  - Use bucket policy or ACLs for this

- S3 Requester Pays bucket do not work with website endpoint
  - Such a request (requester pays) to an S3 website endpoint will return
    - HTTP 403 Access Denied

Review Topic : Simple Storage Service (S3)

Key difference between S3 REST API and S3 Website Endpoints

Key Difference	REST API Endpoint	Website Endpoint
Access control	Supports both public and private content.	Supports only publicly readable content.
Error message handling	Returns an XML-formatted error response.	Returns an HTML document.
Redirection support	Not applicable	Supports both object-level and bucket-level redirects.
Requests supported	Supports all bucket and object operations	Supports only GET and HEAD requests on objects.
Responses to GET and HEAD requests at the root of a bucket	Returns a list of the object keys in the bucket.	Returns the index document that is specified in the website configuration.
Secure Sockets Layer (SSL) support	Supports SSL connections.	Does not support SSL connections.

S3 Static Website Hosting – Redirection

- If your bucket is configured for website hosting, you can redirect requests for an object to another object in the same bucket or to an external URL/Domain
- Redirection is a bucket level operation
- You can redirect (re-route) all requests (at the bucket level) to another website
- You can do conditional redirection based on object or prefixes in a request
- You can also redirect requests that would return an error

Amazon S3

Using Pre-Signed URLs to share objects

Sharing S3 Objects via Pre-Signed URLs

- By default, all objects are private and only the objects owner has permission to access them.
- Pre-signed URLs can be used to provide temporary access to a specific object (sharing the object), to those who do not have AWS credentials,
  - Example is customers who bought website subscription, or product subscription online
- Generally speaking, to generate a pre-signed URL to upload (or download) an object, this can only be done if the creator (IAM user, Root Account, EC2 instance/App, some with STS tokens) of the pre-signed URL has the necessary permissions to upload (or download) that object.
- The pre-signed URL will grant access to a specific object, in a specific bucket, for a specific HTTP method (GET, PUT…etc), for a limited time.
- This will work even if the bucket and object are both private

Review Topic : Simple Storage Service (S3)

Sharing S3 Objects via Pre-Signed URLs

- One way to generate this Signed URLs can be through using Lambda function behind an API Gateway
- The API gateway will pass the request after authentication to the Lambda function
- The Lambda function with the right permissions on the objects/bucket will request and return the Singed URL through the S3 API.

AWS S3

Object Locking

S3 Object Lock

- With Object Lock you can store objects in S3 buckets following the Write Once Read Many (WORM) model.
  - Hence it does help S3 users/customers meet any regulatory requirements that require WORM storage.
  - It also provides another layer of protection for objects
- It is used to protect an object(s) from being deleted or overwritten for a period of time or indefinitely
- Object lock can be configured at the bucket level (as default configuration for the bucket) or at the object version level
- Object Lock requires versioning to be enabled on the bucket
  - It will be enabled automatically when enabling Object lock (if versioning was not enabled before hand)
- S3 buckets with Object lock enabled can NOT be used as destination buckets for S3 server access logging

S3 Object Lock – Two ways to Object Lock

The following two ways can be used at the Object level to Lock objects

- **Retention Period:**
  - Specifies a period of time during which the object remains WORM protected (locked)
  - Applies at the object version level. Different object versions can have different retention periods
  - Retention period (Retain Until date) is placed in the object version's metadata
  - You can extend the retention period of a locked object by submitting a new lock request
  - This can be configured at the Object level or to all Objects in a bucket through the Bucket default Retention settings.
- You can set minimum and maximum allowed retention periods on a bucket using bucket policies
  - Comes in handy in case the bucket is the destination for an S3 Replication and you want to set min and max retention values

- **Legal Hold:**
  - It does the same function as retention period, but has no expiration.
    - It will remain in place until it is removed
    - It is completely independent from Retention period
    - Can be placed or removed by any user with the s3:PUTObjectLegalHold permission
- An object version can have both, either one, or none of the above ways applied.

S3 Object Lock – Retention Modes

- **Governance Mode:**
  - Users can not overwrite or delete an object version or alter its lock settings unless they have special permissions to do so.
  - Protects the objects from being deleted or overwritten to by most users, but not all (those who do not have the S3:BypassGovernanceRetention permission)
  - Users with the right permissions can change Retention settings or even delete the object if required
  - Users with the above permission must include the **x-amz-bypass-governance-retention:true** as a request headerto override governance mode
  - It can be used to test Retention before going into Compliance mode

- **Compliance mode:**
  - A protected object can not be deleted or overwritten by any user, not even the root user
  - The retention mode can not be changed, and the retention period can not be shortened
  - It ensures that the object is not deleted or overwritten for the entire retention period duration

- An object can have either mode, but not both.

S3 Object Lock

- To use it, you must enable Object Lock on the bucket first (can only be done at the time you create the bucket)
  - Optionally you can configure bucket default retention settings(can be done to an existing bucket as well)
  - AWS must be contacted to enable object lock on an existing bucket
- Once enabled, it can not be disabled, and versioning on that bucket can not be suspended
- When an object version is locked, Amazon S3 will store that information in the object version's metadata
- Placing object lock on an object version, will protect that version only
  - It does not stop creating new versions of the object. The new versions will not be automatically locked

Amazon S3

Same and Cross Region Replication (SRR and CRR)

Review Topic : Simple Storage Service (S3)

S3 Replication (SRR & CRR)

- Is a bucket level replication which enables
  - Automatic , Asynchronous copying of Objects across buckets.
  - The source and destination buckets can belong to the same or different AWS accounts
  - They source and destination buckets can be in the same or different AWS regions

- Types of Object Replication:
  - Same Region Replication (SRR) & Cross Region Replication (CRR)

- You can request to replicate all or a subset of objects with specific key name prefixes.
- You can configure AWS S3 Replication with S3 Lifecycle management rules
- Requires versioning to be enabled on both source and destination buckets
- If the source has S3 object lock enabled, the destination bucket mush have that enabled too.

Review Topic : Simple Storage Service (S3)

S3 Replication

- The replicas will:
  - Be exact replicas of the source bucket objects.
  - Share the same key names and metadata (Creation time, version ID, ACL, Storage Class, User-defined metadata)

- Optional:
  - You can specify a different storage class for object replicas while creating the replication configuration
    - If you did not specify it, the same storage class as the source object class will be used for the replica in the destination bucket

- AWS S3 will encrypt data in-transit across regions using SSL

S3 Replication – Use cases

- SRR Can be used for:
  - Aggregating log data into a single bucket (from same or different account buckets)
  - You can replicate between different environments that work on the same data (ex. Production and testing)
  - Replicate critical data within the same region for data sovereignty laws

- CRR can be used for:
  - Meeting compliance requirements, where you need to store copy of your data a distance away
  - Minimize latency to your users/customers by availing the data in other AWS regions closer to them
  - Availing the data for processing by any sort of clusters in two different regions

S3 Replication – What is Replicated

- Any new objects created after you add a replication configuration, and changes to existing objects
- Object metadata
- Objects encrypted using SSE-S3, and optionally those encrypted with SSE-KMS
- Unencrypted objects
- Amazon S3 replicates only objects in the source bucket for which the bucket owner has permission to read objects and read ACLs.
- Any object ACL updates are replicated
- Object lock retention information if there is any
- S3 replicates object tags, if any.

S3 Replication – What is Replicated (DELETE Operation)

If you delete an object from the source bucket, the cross-region replication behavior is as follows:

- If a DELETE request is made without specifying an object version ID,
  - AWS S3 adds a delete marker, which cross-region replication replicates to the destination bucket.

- If a DELETE request specifies a particular object version ID to delete,
  - AWS S3 deletes that object version in the source bucket, but it does not replicate the deletion in the destination bucket
    - It does not delete the same object version from the destination bucket
      - » This behavior protects data from malicious deletions.

Review Topic : Simple Storage Service (S3)

S3 Replication – What is NOT replicated

- AWS S3 will not replicate objects that existed before you added replication configuration
  - You can use Copy API to copy existing bucket data
- Objects created with SSE-C or SSE-KMS (if not enabled) are not replicated
- If the object owner is different from the source bucket owner, and bucket owner does not have permissions to the object, AWS S3 will not replicate these objects
- Actions performed by life cycle configuration
- Objects in the source bucket that were replicated from another bucket into the source
- Updates to bucket-level sub-resources are not replicated (life cycle rules, notifications)
  - This allows you to have different bucket configurations on the source and destination buckets.

Amazon S3

- Cross Origin Resource Sharing (CORS)
- Transfer Acceleration

S3 Cross-Origin Resource Sharing (CORS)

CORS is a security feature of modern web browsers

- It is a way by which a client web application that are loaded in one domain can interact (using HTTP methods) with resources from another domain
- CORS can be used to allow web applications to access resources on your S3 buckets/resources
  - An an example: You can host web fonts on your S3 bucket, then configure your bucket to allow CORS requests for web fonts.
    - Other domains (web pages) will issue CORS requests to load web fonts from your S3 bucket
  - Another example, CORS can be used to allow clients of a website hosted on S3 (http://bucket.s3-website.us-east-1.amazonaws.com) to ue JavaScript on these pages to make authenticated GET and PUT requests from the S3 bucket API endpoint (bucket.s3.us-east-1.amazonaws.com).
    - Without CORS instructing the client web browsers to allow this, it would normally get blocked by the web browser.

S3 Cross-Origin Resource Sharing (CORS)

- CORS allows to easily build web applications that use JavaScript and HTML5 to interact with resources in Amazon S3,
  - This enables:
    - The implementation of HTML5 drag and drop uploads to Amazon S3,
    - Show upload progress, or
    - Update content.

- CORS configuration on an S3 bucket, is an XML document with CORSrules that identify:
  - The origins (domains) allowed to access the bucket
  - The HTTP methods/operations permissisble (GET, PUT, POST, DELETE…)

Review Topic : Simple Storage Service (S3)

S3 Transfer Acceleration

- Is used to accelerate object uploads/downloads to S3 buckets from users over long distances
- Typical use case is when uploading objects to your S3 bucket happens from users across the world, over the internet
- Transfer acceleration is as secure as direct upload to S3 over the internet
- It utilizes AWS Cloudfront's edge location nearest to the upload source (user), once data arrives at the edge location, it gets routed to the destination S3 bucket over an optimized network path
- You can use transfer accelerated buckets for all S3 Operations except GET Service (list buckets), PUT Bucket (create bucket), DELETE Bucket
  - It also does not support cross region Copy

S3 Transfer Acceleration

- In order to use it:
  - Enable Transfer Acceleration on the S3 bucket
    - Once enabled, it can only be suspended, it can NOT be disabled
    - Can only be done by the bucket owner, or a different used with permission from the bucket owner
      - After enabling it, it may take up to 20 mts before you notice upload speed enhancements
  - Point your PUT/GET requests to:

    <bucketname>.S3-accelerate.amazonaws.com (for IPv4)

    <bucketname>.S3-accelerate.dualstack.amazonaws.com (for IPv4/IPv6)
  - The names of the bucket used must be DNS compliant, and MUST not have periods (.)

AWS S3

Performance Considerations

S3 Performance Considerations

- Amazon S3 automatically scales to high request rates.

- Applications can achieve
  - 3,500 PUT/COPY/POST/DELETE and
  - 5,500 GET/HEAD requests per second per prefix in a bucket.

- There are no limits to the number of prefixes in a bucket.
  - You can increase your read or write performance by parallelizing reads.
  - For example, if you create 10 prefixes in an Amazon S3 bucket to parallelize reads, you could scale your read performance to 55,000 read requests per second.

- S3 maintains an index of object key names in each region
  - Object keys (names) are stored in an order across multiple partitions of this index
  - The object key name dictates which partition the key is stored in

- Sequentially named objects are more likely to be saved in the same partition

Amazon Glacier Select

- At high request rates to access sequentially named objects, S3 will place a higher load on the available I/O of the partition hosting these keys (names) which will impact performance

- Some data lake applications on Amazon S3 scan millions or billions of objects for queries that run over petabytes of data.
  - These data lake applications achieve single-instance transfer rates that maximize the network interface use for their Amazon EC2 instance, which can be up to 100 Gb/s on a single instance.
  - These applications then aggregate throughput across multiple instances to get multiple terabits per second.

S3 Performance Considerations – Best Practices

- Using Caching for Frequently Accessed Content
- Timeouts and Retries for Latency-Sensitive Applications
- Horizontal Scaling and Request Parallelization for High Throughput
  - For high-throughput transfers, Amazon S3 advises using applications that use multiple connections to GET or PUT data in parallel.
- Using Amazon S3 Transfer Acceleration to Accelerate Geographically Disparate Data Transfers
- Use Byte-Range Fetches

S3 Performance Considerations – KMS Transaction Limitation

- After configuring CRR, and when many new objects with AWS KMS encryption are added, throttling (HTTP 503 Slow Down errors) can be experienced.
- Throttling occurs when the number of AWS KMS transactions per second exceeds the current limit.
- If the Amazon S3 workload uses server-side encryption with AWS KMS (SSE-KMS),
  - AWS KMS service has limits for the number of encrypt and decrypt requests (or generating keys)
    - It can cause throttling/performance issues for higher request rates.

S3 Performance Considerations - Caching

- To achieve higher transfer rates over a single HTTP connection or single-digit millisecond latencies,
  - Use Amazon **CloudFront** or Amazon **ElastiCache** for caching with Amazon S3.

- Amazon CloudFront transparently caches data from Amazon S3 in a large set of geographically distributed points of presence (PoPs). This can result in high performance delivery of popular Amazon S3 content.

- Amazon ElastiCache is a managed, in-memory cache.
  - With ElastiCache, Amazon EC2 instances can be provisioned to cache objects in memory.
  - This caching results in orders of magnitude reduction in GET latency and substantial increases in download throughput.

- Elemental MediaStore is a caching and content distribution system built for **video workflows** and media delivery from Amazon S3.
  - It provides end-to-end storage APIs for video,
  - It is recommended for performance-sensitive video workloads.

S3 Performance Considerations – Byte Range fetches

- Relies on the Range HTTP header in a GET Object request to fetch a byte-range from an object, transferring only the specified portion.
- If concurrent connections to Amazon S3 are used to fetch different byte ranges from within the same object.
  - This will help achieve higher aggregate throughput compared to a single whole-object request.
- In addition, fetching smaller ranges of a large object allows the application to improve retry times when requests are interrupted.

AWS S3

- S3 and Glacier SELECT
- Monitoring S3
- S3 Batch Operation
- Server Access Logs
- Requester Pays

Amazon S3 SELECT

- With Amazon S3 SELECT, simple structured query language (SQL) statements can be used to filter the contents of Amazon S3 objects and retrieve just the required subset of data.

- By using Amazon S3 SELECT, the amount of data that Amazon S3 transfers can be reduced, which reduces the cost and latency to retrieve this data.
  - Amazon S3 SELECT works on objects stored in CSV, JSON, or Apache Parquet format.
  - It also works with objects that are compressed with GZIP or BZIP2 (for CSV and JSON objects only), and server-side encrypted objects.

- You can specify the format of the results as either CSV or JSON, and you can determine how the records in the result are delimited.

- You pass SQL expressions to Amazon S3 in the request. Amazon S3 SELECT supports a subset of SQL.
  - You can perform SQL queries using AWS SDKs, the SELECT Object Content REST API, the AWS Command Line Interface (AWS CLI), or the Amazon S3 console.

Amazon Glacier Select

- Using Amazon S3 SELECT and Glacier SELECT you can run queries and custom analytics on your data that is stored in Glacier, without having to restore your entire object to Amazon S3.
  - Amazon S3 SELECT and Glacier SELECT support only the SELECT SQL command.
- Querying Archived Objects can be performed (filtering operations) using simple SQL statements directly on your data that is archived by Amazon S3 to Glacier.
- When you provide an SQL query for an archived object, SELECT runs the query in place and writes the output results to an S3 bucket.
- When you perform SELECT queries, Glacier provides three data access tiers—expedited, standard, and bulk.
  - All of these tiers provide different data access times and costs, and you can choose any one of them depending on how quickly you want your data to be available.

Review Topic : Simple Storage Service (S3)

S3 Monitoring

- CloudWatch
  - For S3, metrics that can be monitored by CloudWatch include:
    - S3 Requests, Bucket Storage, Bucket Size, All requests, HTTP 4XX, 5XX errors among others
  - Daily CloudWatch, Bucket level, storage metrics is turned on by default at no additional cost
  - 1 minute CloudWatch metric can be configured both at the bucket and object level
- CloudTrail
  - CloudTrail captures all API requests made to S3 API
  - By default, AWS CloudTrail logs bucket level actions, however you can configure it to log the object level actions (such as DELETE, GET, PUT, POST object events)
  - CloudTrail delivers these logs to a S3 bucket that you configure
- Event Notifications:
  - When certain bucket events occur, AWS S3 Can be configured to automatically send notifications to SNS Topics, SQS Queue, or Lambda Function
  - This is a bucket level configuration, and you can configure multiple events as required
  - You can set notifications for, Create object, Object delete, Object Delete marker created.. and many other bucked related events

Batch Operations

- Amazon S3 batch operations can be used to perform large-scale batch operations on Amazon S3 objects.
- Amazon S3 batch operations can execute a single operation on lists of Amazon S3 objects that you specify.
- A single job can perform the specified operation on billions of objects containing exabytes of data.
- Amazon S3 tracks progress, sends notifications, and stores a detailed completion report of all actions, providing a fully managed, auditable, serverless experience.
- You can use Amazon S3 batch operations through the AWS Management Console, AWS CLI, AWS SDKs, or REST API.

- Use Amazon S3 batch operations to:
  - Copy objects
  - Set object tags
  - Access control lists (ACLs).
  - Initiate object restores from Amazon S3 Glacier
  - Invoke an AWS Lambda function to perform custom actions using your objects.
- These operations can be performed on a custom list of objects, or
- Amazon S3 inventory report can be used to make generating larger lists of objects easy.

Server Access Logging

- Server access logging provides detailed records for the requests that are made to a bucket.
- These logs can be useful for many applications for security purposes or access auditing, it can help analyze the S3 bill, and can help understand the customer base.
- It is disabled by default.
- Each access log record provides details about a single access request, and it provide information such as the requester, bucket name, request time, request action, response status, and an error code, if relevant.
- There is no extra charge for using the feature aside from storage costs
- You need to specify the source bucket, and the target bucket where the logs will be delivered.

To enable it

- Turn on the log delivery on the source bucket
- Grant the Amazon S3 Log Delivery group write permission on the target bucket

Notes:

- Both the source and target S3 buckets must be owned by the same AWS account and in the same AWS Region
- Amazon S3 only supports granting permission to deliver access logs via bucket ACL, not via bucket policy.
- Adding deny conditions to a bucket policy may prevent Amazon S3 from delivering access logs.
- Default bucket encryption on the destination bucket may only be used if AES256 (SSE-S3) is selected.
  - SSE-KMS encryption is not supported currently

Requester Pays Buckets

- Assume that a bucket owner would like to share large data sets (zip code directories, reference data, geographical information, or web crawling data) from an S3 bucket, but does not want to incur charges for Requesting or Downloading the data
  - Requester pays buckets is the solution in this case.
  - The requester pays for data download charges and the request fees
  - The owner of the bucket pays for the data storage fees

- With requester pays buckets
  - Anonymous access to the bucket is not allowed, since the user must be authenticated (known) to pay for their usage
  - The requests made to the requester pay buckets must include the x-amz-request-payer header in their requests

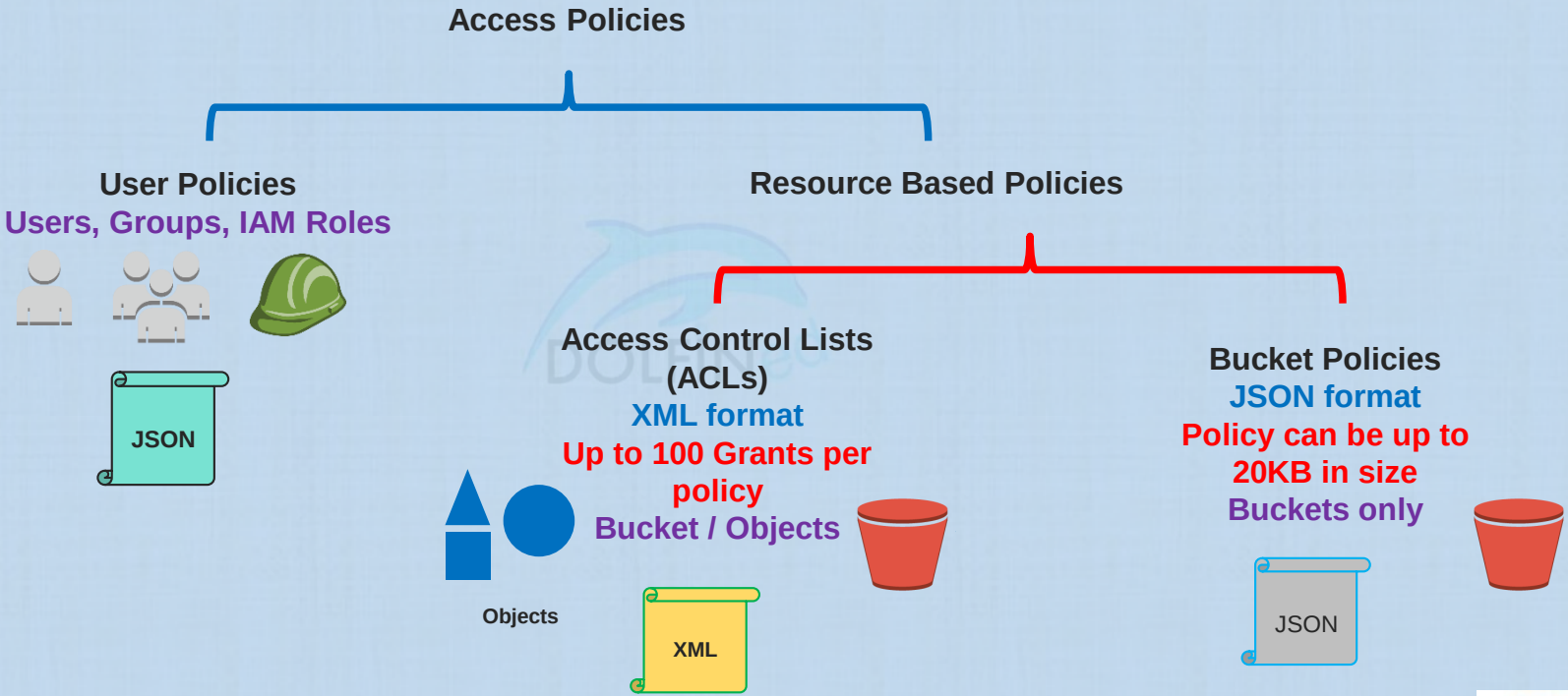
Amazon S3

Identity and Access Management in S3

Resource Access Management

- Managing access refers to granting AWS accounts and users (or to everyone through anonymous/public access) permissions to perform actions/operations on resource by writing and applying an access policy
- Amazon S3 resources are either buckets or objects
- Buckets and objects has many sub resources each, including Access Control List (ACL) where each Object or Bucket has one
- An Access policy is the policy that provides grants to use or request actions against resources
- An access policy defines and controls, through permissions, who has access to what
- When granting a permission you need to define:
  - Who is this permission granted to
  - What resources this permission applies to
  - Specific actions the permission allows on the resources

Access Policy Categories – User policies or Resource-based policies



Depending on the required, you can use user policies only, Resource-based policies only or both together

Resource Ownership

- Objects and Buckets are private by default.
- Only the respective resource owner has FULL CONTROL access to the resource by default
- The resource owner is the account the created the resource (Bucket or object)
  - If an IAM user in an account creates a bucket or uploads an object
    - Still the account that user belongs to, is the resource owner

Request to a bucket – Request types

- Authenticated Requests:
  - The request must include a signature value identifying the requester
    - The AWS Signature version 4 algorithm is a protocol used by AWS, and is supported by S3 , to authenticate inbound API requests to the S3 API.
    - The signature is derived from the user's access and private access key

- Un-Authenticated (anonymous or public) Requests:
  - Are requests that do not include a signature value (i.e the requester is un-identified)
  - AWS recommends against unauthenticated/anonymous access to your buckets or objects unless absolutely required

Evaluation steps for any request

- User Context
  - Verify that the request by the user (or IAM Role) is authorized by the account owner
- Bucket Context
  - Validate if the bucket owner allows the requested operation
  - Checks the bucket ACL, Bucket Policy
- Object Context
  - If the request has an Object as the resource
  - Check the Object ACL to validate whether the operation requested is permitted

Access Control Lists (ACLs)

- ACLs are a list of grants
  - Each grant defined a grantee (to whom permission is allowed) and the allowed permissions
  - ACLs provide basic Read/Write operations on the resources
- ACLs can not do conditional permissions, Bucket policies can

- ACLs can define permissions for:
  - Accounts (Cross Account permissions) – They can not provide user level permissions directly
  - Pre-defined S3 groups:
    - Authenticated Users Groups (Users with AWS Credentials)
    - All users group (includes everyone – even those with AWS credentials)
    - Log Delivery Groups – Used for S3 Access Management to write Logs to an S3 bucket

- **Only use object ACLs to manage permissions for specific scenarios and only if ACLs meet your needs better than IAM and S3 bucket policies.**

Object Access Control Lists (ACLs)

- Are only used when it is required to provide permissions on Objects in a bucket, that are not owned by the bucket owner
  - This is the case of Cross Account permissions to upload object in a bucket owned by other account
  - The ACLs need to be defined by the Object Owner (not the bucket owner)
- Limited to 100 grants per ACL
- The bucket owner can (although they do not own these objects)
  - Deny access to any object in their bucket – including these they do not own
  - Delete any object in their bucket – including these they do not own
  - Archive/Restore any objects in their bucket – including these they do not own

Bucket Access Control Lists (ACLs)

- Can provide permission to Accounts or pre-defined groups
- Can provide cross account permissions to other accounts (not to specific users within other accounts)
- Provide the WRITE permission over the bucket for Log Delivery Groups to write Access Logs into the bucket

Access Control Lists (ACLs) – Possible permissions

Permission	When granted on a bucket	When granted on an object
READ	Allows grantee to list the objects in the bucket	Allows grantee to read the object data and its metadata
WRITE	Allows grantee to create, overwrite, and delete any object in the bucket	Not applicable
READ_ACP	Allows grantee to read the bucket ACL	Allows grantee to read the object ACL
WRITE_ACP	Allows grantee to write the ACL for the applicable bucket	Allows grantee to write the ACL for the applicable object
FULL_CONTROL	Allows grantee the READ, WRITE, READ_ACP, and WRITE_ACP permissions on the bucket	Allows grantee the READ, READ_ACP, and WRITE_ACP permissions on the object

ACL limitations

- Through S3 ACLs, only the following permissions sets: READ, WRITE, READ_ACP, WRITE_ACP, and FULL_CONTROL can be defined.
- Only an AWS account or one of the predefined Amazon S3 groups can be defined as a grantee for the Amazon S3 ACL.
- When specifying email address or the canonical user ID for an AWS account, the ACL applies to all entities in the grantee AWS account (not specific users, Roles or Groups).
- You can't use an ACL to restrict access to individual IAM users or roles or apply ACLs to different objects that share the same prefixes.
- An ACL doesn't support the condition for the S3 operation that the ACL authorizes. Therefore, the bucket owner might not have full control over the objects uploaded by the ACL grantee (unless the bucket owner is included in the object ACL that is uploaded), and this can't be enforced using the Amazon S3 ACL.

Bucket Policies

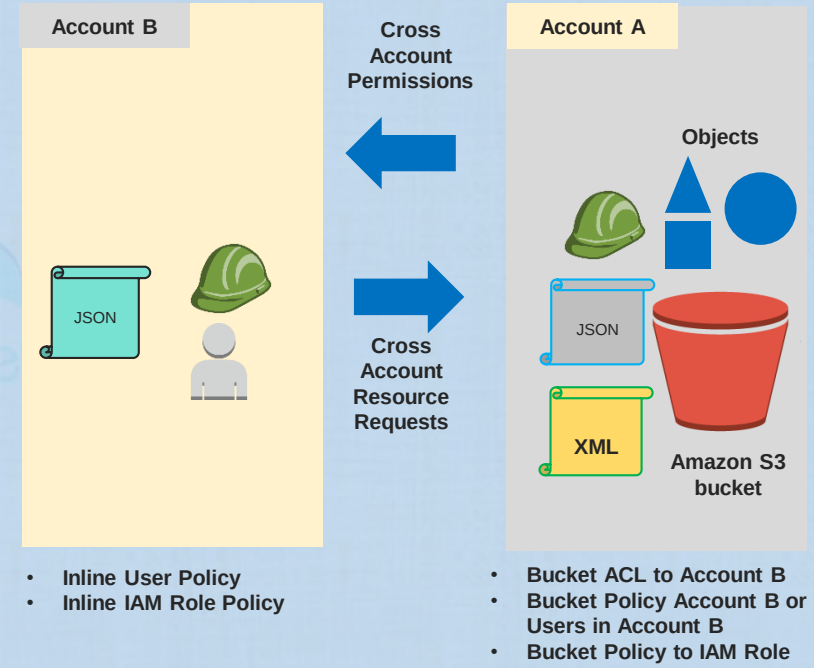
- Can grant access to users or accounts (Same account or Cross Account) to bucket or objects owned by the account owner
- They apply only to resources (bucket or objects) owned by the bucket owner
  - What if the bucket owner allowed other accounts to upload objects in its bucket that it does not own, can the bucket owner provide permissions on these objects to its own account users or cross account?
    - Yes, They can provide in-account or cross-account access to these objects, Provided that the object owner account(s) grant Full-Control permission to the bucket owner, through Object ACLs.
- Are limited to 20KB in size, hence, they can't scale to be granular at the object level for every object in the bucket
- Can do conditional permissions
- Can do the full set of S3 operations

User policies

- Attaches to a user, group, or IAM Role (basically who uses them is authenticated)
- You can NOT grant anonymous access through a user access policy
- IAM User or IAM Role policies are inline-policies
- IAM Group policies are standalone policies

Cross-Account permissions

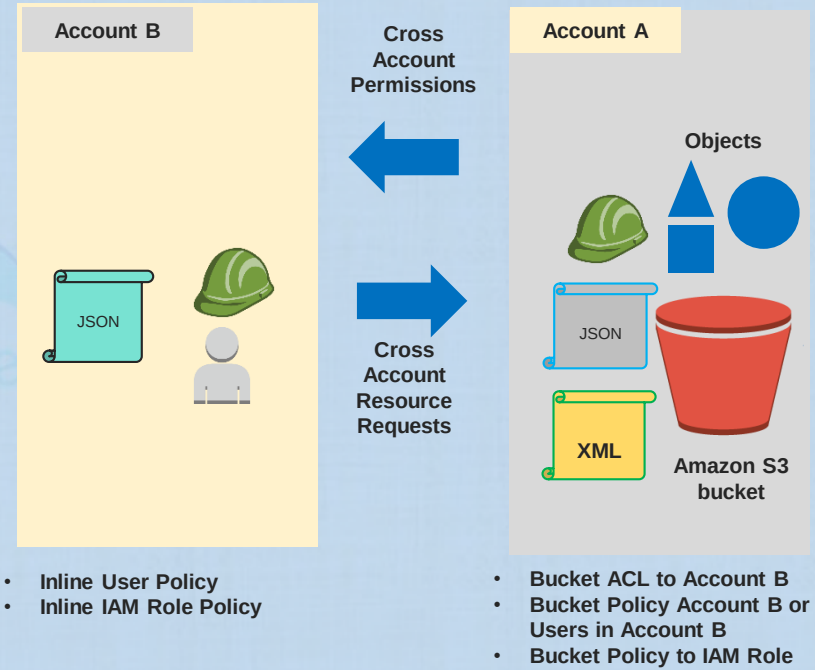
- A bucket owner can grant cross-account permissions to other accounts or to users in other accounts to upload objects
- When the other accounts (or users of those accounts) upload objects into the bucket, they become the resource owner(s) of these objects
- The bucket owner has no permissions over these objects except
  - Can deny access to any object in the bucket included these created by other accounts
  - Can archive/restore, and delete any objects in the bucket (since the bucket owner pays for the storage costs)



Cross-Account permissions – How to!

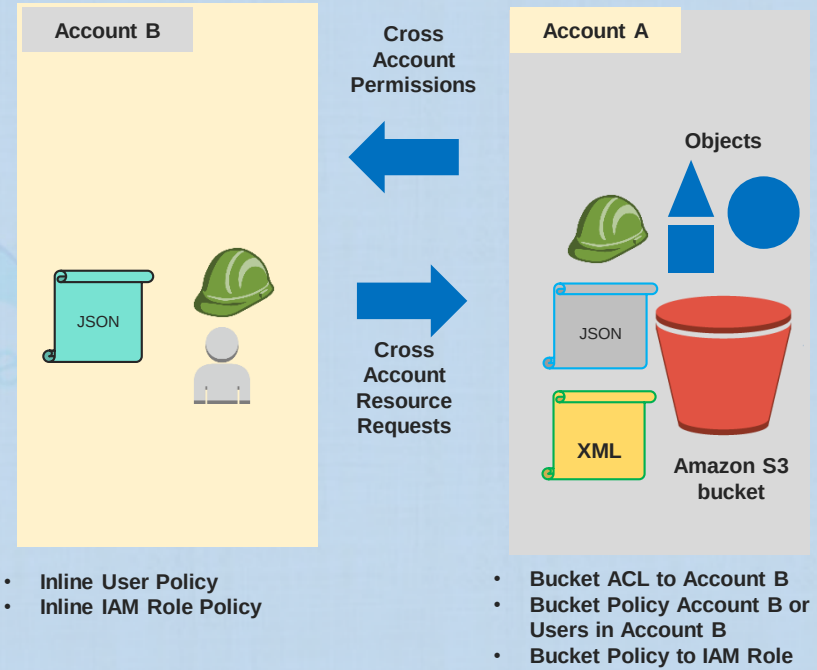
- Option 1) :
  - **Account A:** Resource based policy allowing the required permissions for Account B (or users within), and
  - **Account B:** IAM policy for users/groups to allow users/groups or IAM Roles to use the resource policy permissions on the bucket in account A

- Option 2) :
  - Account A:
    - Create a bucket ACL to allow Account B the WRITE permission to the bucket
    - Create an Object ACL to allow Account B the READ permission on the objects
  - Account B: Same like in Option 1) above



Cross-Account permissions – How to!

- Option 3) :
  - **Account A:** Create an IAM Role with the right permissions on the S3 bucket
    - In the IAM Role trust policy, grant a User or Role in Account B permission to assume this role.
  - **Account B:** Grant through an IAM policy the users or Role permission to Assume the role in Account A.



Amazon S3

Billing

Review Topic : Simple Storage Service (S3)

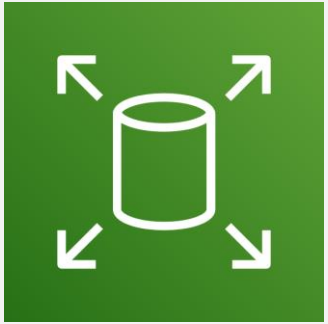
Billing Charges in S3

- No charge for data transferred from/to EC2 to/from S3 in the same region
  - This includes data transferred via Copy command
- Data transfer into S3 is free of charge
- Data transfer out of S3 to EC2 instance in the same region is free
  - Data transferred out of S3 bucket region is charged
- Data transfer via Copy to other regions is charged at the Internet data transfer rates
- Data transferred from S3 to Cloudfront is free

Review Topic : Simple Storage Service (S3)

Requester Pays

- If Requester pays is enabled on a bucket:
  - The bucket owner will only pay for Object Storage fees
  - The requester will pay for:
    - Requests to S3 to upload/download objects
    - Data transfer charges
- Buckets with Requester Pays enabled do not allow anonymous access & Do not support BitTorrent
- Requester pays buckets can not be the target for user logging
- You can NOT specify requester pays at the Object level
- Can be enabled from the bucket properties under the AWS Console



AMAZON ELASTIC BLOCK STORE (EBS) REFRESHER

AWS Elastic Block Store

EBS

Review Topic : EBS

EBS Persistence

- By default, EBS volumes created when an EC2 Instance is launched are deleted when the instance is terminated, including the root volume

- EBS is persistent and can retain the data it has even when the EC2 instance is stopped or restarted,
  - If configured, it can persist after the EC2 instance is terminated
    - You can change this by changing the “DeleteOnTermination” attribute of the instance’s block store (EBS) volume(s)
      - » An EBS volume (root or not) with a DeleteOnTermination attribute set to “false”, will not be deleted when the instance is terminated
      - » This comes in handy when you want to keep the EBS volume for future use while you terminate the EC2 instance

IOPS Performance – Instance Store vs. EBS

- **Use Instance Store instead of EBS** if very high IOPS rate is required
  - Instance store, although can not provide for data persistence, but it can provide for much higher IOPS compared to, network attached, EBS storage

Review Topic : Elastic Compute Cloud

EBS types

- **General Purpose (gp2)**
  - SSD-Backed (Solid State Drives)
  - Are better for transactional workloads and Dev/Test environments where performance is highly dependent on IOPS
  - Use cases include, System boot volumes, Virtual desktops, Low-latency interactive apps, and Development and test environments

- **Provisioned IOPS (io1)**
  - SSD-backed
  - Highest-performance SSD volume for mission-critical low-latency or high-throughput workloads (Critical business application or Production DBs)
  - Provides sustainable IOPS performance & Low latency
  - Max IOPS/Volume , 64,000 IOPS

Review Topic : Elastic Compute Cloud

EBS types

- **Throughput Optimized HDD (not SSD) (st1)**
  - Ideal for streaming, big data, log processing, and data warehousing
  - Can NOT be used as a boot volume
  - Use for frequently accessed, throughput intensive workloads

- **Cold HDD (sc1)**
  - Ideal for less frequently accessed workloads
  - Throughput-oriented storage for large volumes of data that is infrequently accessed
  - Scenarios where the lowest storage cost is important
  - Cannot be a boot volume

More can be found here

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-volume-types.html

AWS EBS

- Snapshots
- Encryption

Review Topic : EBS

EBS Snapshots

- EBS Snapshots are point-in-time Images/copies of your EBS volume(s)
  - Can be taken from root or non-root (Data) EBS volumes and will only include the data that was on the volume when the snapshot process started (but not the data written afterwards)
- Any data written to the volume after the snapshot process is initiated, will NOT be included in the resulting snapshot (but will be included in future, incremental, updates)
- EBS Snapshots are stored on S3, however you can NOT access them directly. You can only access them through EC2 APIs
  - This is unlike Instance-Store AMIs (where you specify a S3 bucket of your own)
- Snapshots can be automated (scheduled using Amazon Data Lifecycle Manager) or user-initiated manually
- If the instance is running, you can create a snapshot and access the volume simultaneously

EBS Snapshots - Characteristics

- Snapshots are Region specific
  - To migrate an EBS from one AZ to another, create a snapshot (region specific), and create an EBS volume from the snapshot in the intended (other) AZ
- You can only create/restore an EBS volume (from a snapshot) of the same or larger size than the original volume size, from which the snapshot was initially created
- You can NOT create a snapshot of an instance store volume, however you can:
  - Backup individual files on the instance store volume (to S3 for example)
  - Copy the data to a new EBS volume attached to the same EC2 instance (then create a snapshot of the EBS volume)
- EBS snapshots are stored incrementally
- EBS snapshots are asynchronously created
  - This refers to the fact that, updates or changes to snapshots do not have to happen immediately when the respective volume data changes

Review Topic : EBS

EBS Encryption Support

- Is done by the servers hosting the EC2 instances
- Uses AWS Key Management Service (KMS) Customer Master Keys (CMKs)
- EBS Encryption is supported on **all EBS volume types**, all instance families, but **NOT on all EC2 instance types with all families.**
- Snapshots of encrypted volumes are also encrypted.
- Creating an EBS volume from an encrypted snapshot will result it an encrypted volume.
- You can expect the same IOPS performance on encrypted volumes as on unencrypted volumes, with a minimal effect on latency.
- Data encryption at rest means, encrypting data while it is stored on the data storage device
- There are many ways you can encrypt data on an EBS volume at rest, while the volume is attached to an EC2 instance
  - Use 3rd party EBS volume (SSD or HDD) encryption tools
  - Use encrypted EBS volumes
  - Use encryption at the OS level (using data encryption plugins/drivers)
  - Encrypt data at the application level before storing it to the volume
  - Use encrypted file system on top of the EBS volume

EBS Encryption – Data In-Transit

- Remember that the EBS volumes are not physically attached to the EC2 instance, rather, they are virtually attached through the AWS infrastructure
  - This means when you encrypt data on an EBS volume, data is actually encrypted on the server hosting the EC2 instance then transferred, encrypted, to be stored on the EBS volume
    - This means data in transit between EC2 and Encrypted EBS volume is also encrypted
- Data at rest in the EBS volume is also encrypted
- Encrypted is handled transparently, where encrypted volumes are accessed exactly like unencrypted ones
- You can attach an encrypted and unencrypted volumes to the same EC2 instance

Review Topic : EBS

EBS Encryption

- There is no direct way to change the encryption state of a volume.

- To change the state (indirectly) you need to follow either of the following two ways:
  1. Attach a new, encrypted, EBS volume to the EC2 instance that has the data to be encrypted then,
     - Mount the new volume to the EC2 instance
     - Copy the data from the un-encrypted volume to the new volume
     - Both volumes MUST be on the same EC2 instance

  2. Create a snapshot of the un-encrypted volume,
     - Copy the snapshot and choose encryption for the new copy, this will create an encrypted copy of the snapshot
     - Use this new copy to create an EBS volume, which will be encrypted too
     - Attach the new, encrypted, EBS volume to the EC2 instance
     - You can delete the one with the un-encrypted data

EBS Volume/Snapshot - Encryption Keys

- To encrypt a volume or a snapshot, you need an encryption key, these keys are called Customer Master Keys (CMKs) and are managed by AWS Key Management Service (KMS)

- When encrypting the first EBS volume, AWS KMS creates a **default CMK key**,
  - This key is used for your first volume encryption, encryption of snapshots created from this volumes, and subsequent volumes created from these snapshots

- After that, each newly encrypted volume is encrypted with a unique/separate AES256 bits encryption key
  - This key is used to encrypt the volume, its snapshots, and any volumes created of its snapshots

Review Topic : EBS

EBS Volume Migration

- **EBS volumes are AZ specific** (can be used in the AZ where they are created only)
  - To move/migrate **your EBS volume to another AZ** in the same region,
    - Create a snapshot of the volume,
    - Use the snapshot to create a new volume in the new AZ
    - Snapshots are Region specific
  - To move/migrate **you EBS volume to another region**,
    - You need to create a snapshot of the volume,
    - Copy the snapshot and specify the new region where it should be,
    - In the new region, create a volume out of the copied snapshot

Review Topic : EBS

Sharing EBS Snapshots

- By default, only the account owner can create volumes from the account snapshots

- You can share your **un**encrypted snapshots with the AWS community by:
  - **Making them public (modifying the snapshot permissions to public) or,**
  - Share your unencrypted snapshots with a selected AWS account(s), **by making them private** then selecting the AWS accounts to share with

- You **can NOT** make your encrypted snapshots public
  - However, you can share it with other AWS accounts if needed, but you need to mark it "private" then share it.
  - You need to also provide the other accounts permissions on the CMK key used to encrypt the snapshot
    - You can NOT share encrypted snapshots that were encrypted with the default CMK key

Review Topic : EBS

Copying Snapshots

- The following is possible for your own snapshots:
  - You can copy an un-encrypted snapshot
    - During the copy process, you can request encryption for the resulting copy
  - You can also copy an encrypted snapshot
    - During the copy process, you can request to re-encrypt it using a different key
- Snapshot copies receive a new Snapshot ID, different from that of the original snapshot
- You can copy a snapshot within the same region, or from one region to another
- To move a snapshot to another region, you need to copy it to that region
- You can only make a copy of the snapshot when it has been fully saved to S3 (its status shows as “complete”), and not during the snapshot’s “pending” status (when data blocks are being moved to S3)
- Amazon S3’s Server Side Encryption (SSE) protects the snapshot data in-transit while copying (Since the snapshot and the copy are both on S3)

Use Cases - Copying a Snapshot

Use cases for Copying a snapshot

- Geographic expansion
- Disaster Recovery : backing up your data and logs in another region, restoring from that data/logs in case of a disaster
- Migration to another region
- Encryption (of unencrypted volumes)
- Data retention and auditing requirements
  - Copy data and logs to another AWS account for auditing
  - This also protects against account compromise

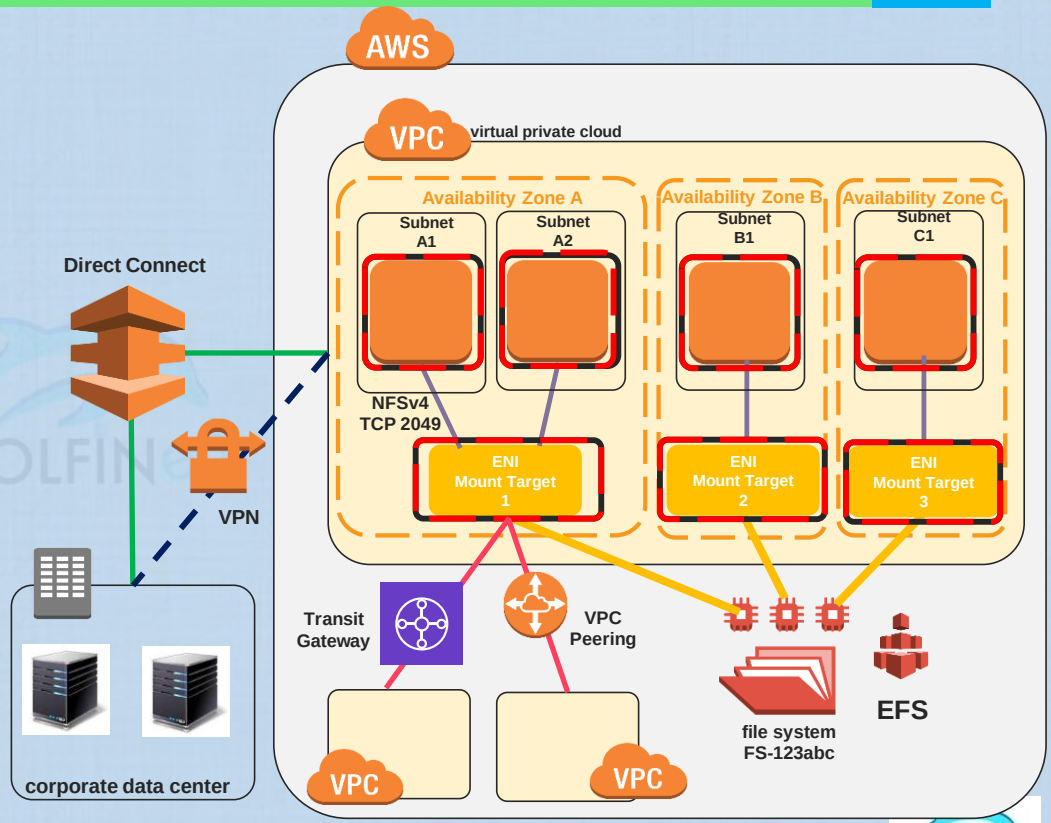
AMAZON ELASTIC FILE SYSTEM (EFS)

Amazon Elastic File System (EFS)

Introduction to EFS

Amazon Elastic File System (EFS) – What is it?

- Amazon Elastic File System (Amazon EFS) provides simple, highly available, highly durable, and scalable file storage in the cloud for use with Amazon EC2 or on-premise servers.
- Amazon EFS file systems store **data and metadata** across multiple Availability Zones in an AWS Region.
- Amazon EFS file systems can be mounted on Amazon EC2 instances, on-premises servers through an AWS Direct Connect or VPN connection, through a VPC Peering Connection or a transit gateway (same or different Account VPCs).



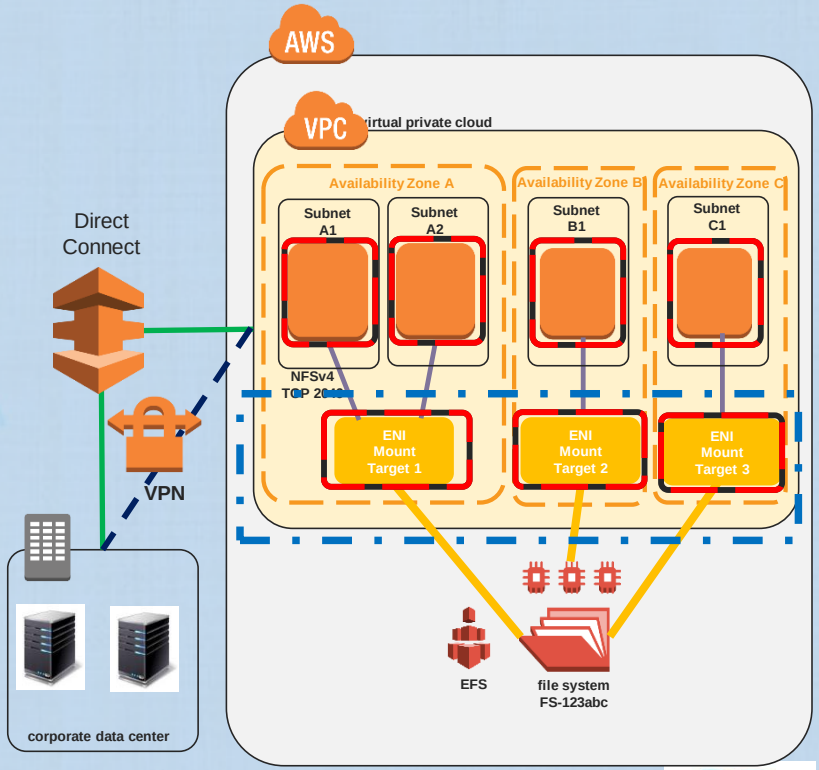
Amazon Elastic File System (EFS) – What is it?

- EFS service manages all the file storage infrastructure
- EFS file systems (EFS storage capacity) is elastic, growing and shrinking automatically as files get added and removed.
  - EFS file systems can grow to petabyte scale, drive high levels of throughput, and allow access to the stored data from a large number of EC2 instances in parallel
- Amazon EFS provides file system access semantics, such as strong data consistency and file locking.
- Portable Operating System Interface (POSIX) Compliant. **It is limited to Linux Instances**
- For some AMIs, you'll need to install an NFS client to mount your file system on your Amazon EC2 instance.
- You can control access to your file system and data through IAM and POSIX file permissions
- Amazon EFS supports the Network File System version 4 (NFSv4.1 and NFSv4.0) protocol.
  - The applications and tools that you use today work seamlessly with Amazon EFS.
- EFS file system has mount targets, through which EC2 instances can mount the file system
- Multiple Amazon EC2 instances in the same region, same VPC and in different availability zones, can access an Amazon EFS file system at the same time
  - This provides a common data source for workloads and applications running on more than one instance or server.

Amazon Elastic File System (EFS) – Mount Targets

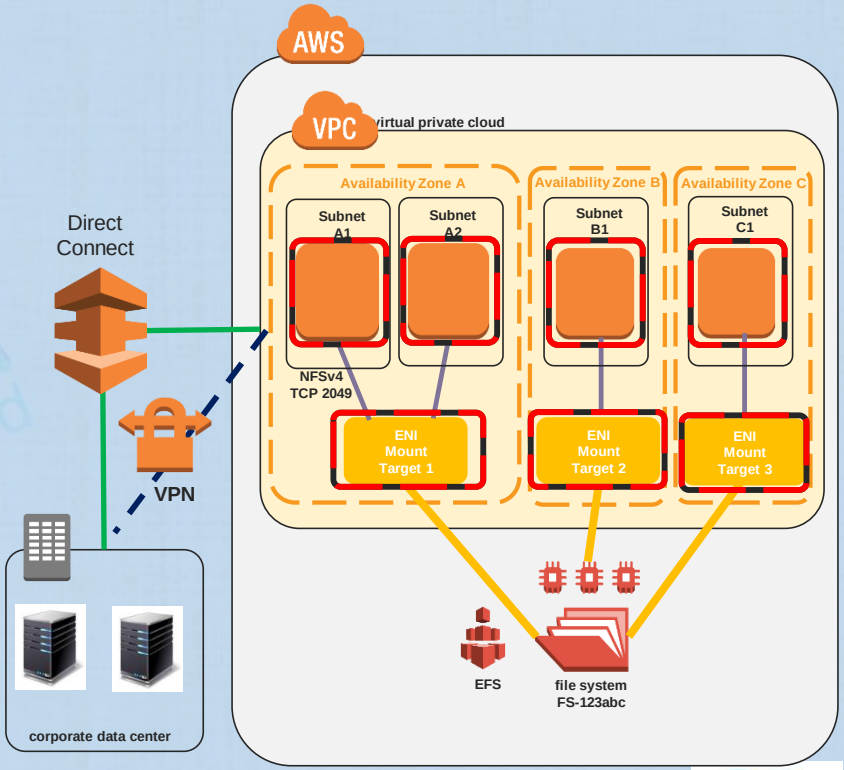
To access your Amazon EFS file system in a VPC, you create one or more mount targets in the VPC.

- **You can create only one mount target in each Availability Zone in an AWS Region.**
- The IP addresses and DNS for your mount targets in each AZ are static.
  - Keep this in mind when you design for high availability and failovers to other Availability Zones (AZs)
- If there are multiple subnets in an Availability Zone in your VPC, you create a mount target in one of the subnets which will be used by all EC2 instances in the different subnets in that AZ.
- A mount target provides an IP address for an NFSv4 endpoint at which you can mount an Amazon EFS file system.
- The file system can be mounted using its Domain Name Service (DNS) name, which resolves to the IP address of the EFS mount target.
- Mount targets need to have associated security groups.
- They act as a virtual firewall that controls the traffic



Amazon Elastic File System (EFS) – Mount Targets

- Mounts targets themselves are highly available.
- After mounting the file system by using the mount target, it can be used like any other **POSIX-compliant file system**.
- AWS recommends that you create mount targets in all the Availability Zones
- A File System can be used through mount Targets in a single VPC at a time.
  - You can not have mount targets in different VPCs for the same File system at the same time
    - If you need to use the file system (create mount targets to the file system) in a different VPC, delete the mount targets in the existing VPC and create new ones in the new VPC.
- Mount targets need to have associated security groups.
- These security groups act as a virtual firewall that controls the traffic between them.



Amazon Elastic File System (EFS)

EFS:

- **Use Cases**
- **EFS and On-Premise Compute**
- **EFS Storage Classes**

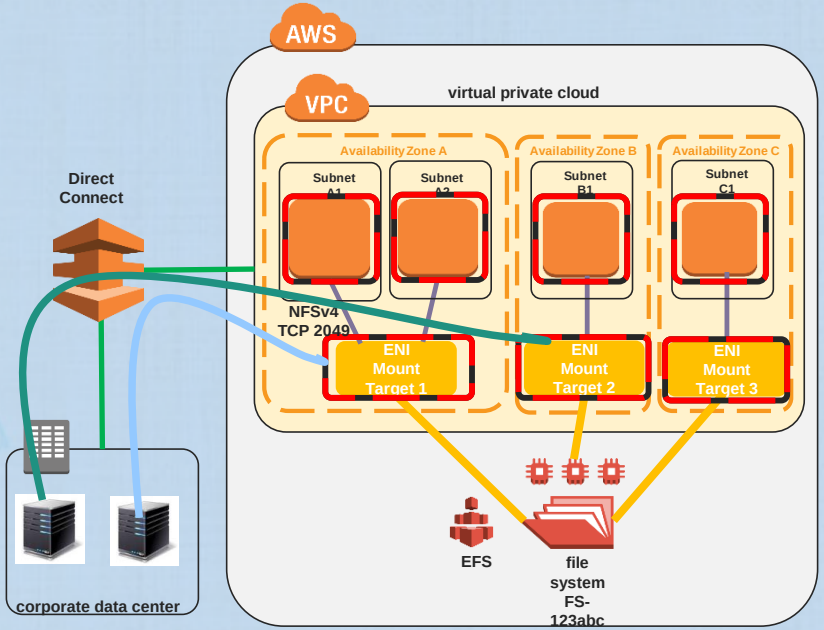
Amazon Elastic File System (EFS) – Use Cases

- Amazon EFS is designed to meet the performance needs of the following use cases.
  - **Big Data and Analytics**
  - **Media Processing Workflows**
  - **Content Management and Web Serving**
  - **Home Directories**

Amazon Elastic File System (EFS) – Mounting to On premise Servers

- You can mount your Amazon EFS file systems on your on-premises data center servers when connected to your Amazon VPC with AWS Direct Connect or VPN.
  - AWS recommends using the IP address of the mount targets not the DNS name
  - You can mount your EFS file systems on on-premises servers to migrate data sets to EFS, which will:
    - Enable cloud bursting scenarios, or
    - Backup your on-premises data to EFS.

- For High Availability
  - To ensure continuous availability between your on-premises data center and your Amazon VPC,
    - AWS recommends configuring two AWS DX connections.
  - AWS recommends that your application be designed to recover from potential connection interruptions.



The on-premises server must have a Linux-based operating system.

EFS and On-Premise servers – Cloud Bursting use case

- Using an Amazon EFS file system mounted on an on-premises server, you can migrate on-premises data into the AWS Cloud hosted in an Amazon EFS file system. You can also take advantage of bursting.

- A Customer can move data from their on-premises servers into Amazon EFS and analyze it on a fleet of Amazon EC2 instances in their Amazon VPC.
  - Then, results can be stored in EFS permanently or be moved back to the on-premises server(s).

Amazon Elastic File System (EFS) Storage Classes

- Amazon EFS file systems have two storage classes available:
  - **Standard**
    - The Standard storage class is used to store frequently accessed files.
  - **Infrequent Access**
    - It meets the high availability, high durability, elasticity, or POSIX file system access that EFS provides.
    - Example use cases include audit requirements, performing historical analysis, or performing backup and recovery.
- EFS IA storage is compatible with all EFS features, and is available in all AWS Regions where Amazon EFS is available.

EFS Lifecycle Management

- EFS Lifecycle Management
  - EFS Lifecycle Management automatically manages cost-effective file storage.
  - All writes to files in IA storage are first written to Standard storage, then transitioned to IA storage.
  - When enabled, lifecycle management automatically migrates files that have not been accessed for 30 days to the EFS IA storage class.
  - When enabled, lifecycle management automates moving files from Standard storage to IA storage.

- Notes:
  - File metadata, such as file names, ownership information, and file system directory structure, is always stored in Standard storage to ensure consistent metadata performance.
  - Files smaller than 128 KB are not eligible for lifecycle management and are always stored in the standard class.

Amazon Elastic File System (EFS) – Pricing

- Using EFS, you only pay for the storage used by the file system(s) you create/use
  - No minimum fee or setup cost.
- EFS IA billing is based on the amount of data stored in IA and the data I/O used to access the data stored in IA.
  - Access charges are incurred when files in IA storage are read, and when files are transitioned to IA storage from Standard storage.
- For On Premises servers:
  - Customers will be charged for the AWS Direct Connect connection to their Amazon VPCs.

Amazon Elastic File System (EFS)

EFS:

- **Data Encryption**
- **Backup**

Amazon Elastic File System (EFS) – Data Encryption

- Amazon EFS supports two forms of encryption for file systems, encryption in transit and encryption at rest.
  - Amazon EFS automatically manages the keys for encryption in transit, client to EFS encryption.

- Amazon EFS uses AWS Key Management Service (AWS KMS) for key management.

- If you create a file system that uses encryption at rest, data and metadata will be encrypted at rest.
  - When you create a file system using encryption at rest, you specify a Customer Master Key (CMK).
    - The CMK can be:
      - The AWS-managed CMK for Amazon EFS (aws/ elasticfilesystem), or
      - It can be a CMK that you manage.
- **File Data :**
  - Will be encrypted at rest using the CMK that you specified when you created your file system.

- **Meta Data :**
  - The AWS-managed CMK for your file system is used as the master key for the metadata in your file system,
  - This includes file names, directory names, and directory contents.

Amazon Elastic File System (EFS) – Backing up EFS

- There are two options available for backing up your EFS file systems.
  - The EFS-to-EFS backup solution
  - AWS Backup service

- **The EFS-to-EFS backup** solution is suitable for all Amazon EFS file systems in all AWS Regions.
  - It includes an AWS CloudFormation template that launches, configures, and runs the AWS services required to deploy this solution.
  - This solution follows AWS best practices for security and availability.
  - The solution includes a Cloud Watch Event that invokes an Orchestrator AWS Lambda
    - https://aws.amazon.com/solutions/efs-to-efs-backup-solution/

Amazon Elastic File System (EFS) – Backing up EFS Using AWS Backup

- AWS backup can be used to provide a comprehensive backup solution for EFS File systems.
- AWS Backup is a backup service designed to simplify the creation, migration, restoration, and deletion of backups, while providing improved reporting and auditing.
- Amazon EFS is integrated with AWS Backup.
  - You can use AWS Backup to set backup plans where you specify your backup frequency, when to back up, how long to retain backups, and a lifecycle policy for backups.
    - Then assign the EFS file system to this backup plan
- AWS backup does incremental backups of EFS file system
- You can restore from a backup to a new EFS file system or to the same EFS File system

Amazon EFS & AWS DataSync – Data Migration to EFS

- AWS DataSync is a data transfer service that simplifies, automates, and accelerates moving and replicating data between on-premises storage systems and AWS storage services **over the internet or AWS Direct Connect.**
- AWS DataSync can transfer your file data, and also file system metadata such as ownership, time stamps, and access permissions.
- AWS recommends using AWS DataSync to transfer data into Amazon EFS.
- You can use AWS DataSync to:
  - Transfer files from an existing file system to Amazon EFS.
  - Transfer files between two EFS file systems, this includes fie system in different AWS Regions and file systems owned by different AWS accounts.
- Using DataSync to copy data between EFS file systems, you can perform:
  - One-time data migrations,
  - Periodic data ingestion for distributed workloads, and
  - Automate replication for data protection and recovery.

Amazon Elastic File System (EFS) - Monitoring

- You can use the following automated monitoring tools to watch Amazon EFS and report when something is wrong:
  - **Amazon CloudWatch Alarms**
    - Watch a single metric over a time period that you specify and perform one or more actions based on the value of the metric relative to a given threshold over several time periods.
      - ➢ The action is a notification sent to an Amazon SNS topic or Amazon EC2 Auto Scaling policy.
    - CloudWatch alarms do not invoke actions simply because they are in a particular state; the state must have changed and been maintained for a specified number of periods.
  - **Amazon CloudWatch Logs**
    - Monitor, store, and access your log files from AWS CloudTrail or other sources.
  - **Amazon CloudWatch Events**
    - Match events and route them to one or more target functions or streams to make changes, capture state information, and take corrective action.
  - **AWS CloudTrail Log Monitoring**
    - Share log files between accounts, monitor CloudTrail log files in real time by sending them to CloudWatch Logs, write log processing applications in Java, and validate that your log files have not changed after delivery by CloudTrail.

Amazon Elastic File System (EFS)

- EFS Data Consistency
- Monitoring
- Mounting File Systems from other accounts

Amazon EFS - Data Consistency in Amazon EFS

- Amazon EFS provides the **Close-to-open (Open-after-Close) Consistency** semantics that applications expect from NFS.
  - NFS provides no guarantees that all clients see exactly the same data at all times. (Remember eventual consistency)

- In Amazon EFS, write operations are durably stored across Availability Zones in these situations:
  - An application performs a synchronous write operation, or
  - An application closes a file.

- Depending on the access pattern, Amazon EFS can provide **Stronger Data Consistency** guarantees than open-after-close semantics.
  - Applications that perform synchronous data access and perform non-appending writes have **read-after-write (Strong) consistency** for data access.

Amazon EFS - Mounting EFS File Systems from Another Account or VPC

- You can mount your Amazon EFS file system from Amazon EC2 instances that are in a different account or virtual private cloud (VPC).

- **Mounting from Another Account in the same (shared) VPC**
  - Using shared VPCs, you can mount an Amazon EFS file system that is owned by one account from Amazon EC2 instances that are owned by a different account.
  - EC2 instances can mount the EFS file system using Domain Name System (DNS) name resolution or the EFS mount helper

- **Mounting from another VPC in the same or different account**
  - **When you use a VPC peering connection or transit gateway to connect VPCs**, Amazon EC2 instances that are in one VPC can access EFS file systems in another VPC, even if the VPCs belong to different accounts.
  - You can't use DNS name resolution for EFS mount points in another VPC.
    - To mount your EFS file system, use the **IP address of the mount points** in the corresponding Availability Zone. Alternatively, you can use Amazon Route 53 as your DNS service.
      - In Route 53, by creating a private hosted zone and resource record set, you can resolve the EFS mount target IP addresses from another VPC

Amazon Elastic File System (EFS) – Performance Modes

- Amazon EFS offers **two performance modes**, that allow EFS to support a wide variety of cloud storage use cases, these are:
  - **General Purpose Performance Mode:**
  - **Max I/O Performance Mode**

- General Purpose is:
  - Ideal for latency-sensitive use cases, like web serving environments, content management systems, home directories, and general file serving.
  - AWS recommends the General Purpose performance mode for the majority of your Amazon EFS file systems.

- Max I/O Performance Mode
  - File systems in the Max I/O mode can scale to higher levels of aggregate throughput and operations per second with a tradeoff of slightly higher latencies for file operations.
    - Use it for highly parallelized applications and workloads, such as big data analysis, media processing, and genomics analysis.

AMAZON FSX

Amazon FSx

- Introduction to Amazon FSx
- Single vs Multi-AZ Deployments
- Data Encryption

Amazon FSx for Windows File Server – What is it?

- Amazon FSx for Windows File Server provides a fully managed native Microsoft Windows file system.
  - This is useful for Windows-based apps that need file storage in AWS.
  - Amazon FSx provides shared file storage with full support for the SMB protocol, Windows NTFS, and Active Directory (AD) integration.
  - Amazon FSx uses SSD storage to provide the fast performance with high levels of throughput, IOPS, and consistent sub-millisecond latencies.

- With Amazon FSx, highly durable and available Windows file systems that can be created and accessed from up to thousands of clients using the SMB protocol.

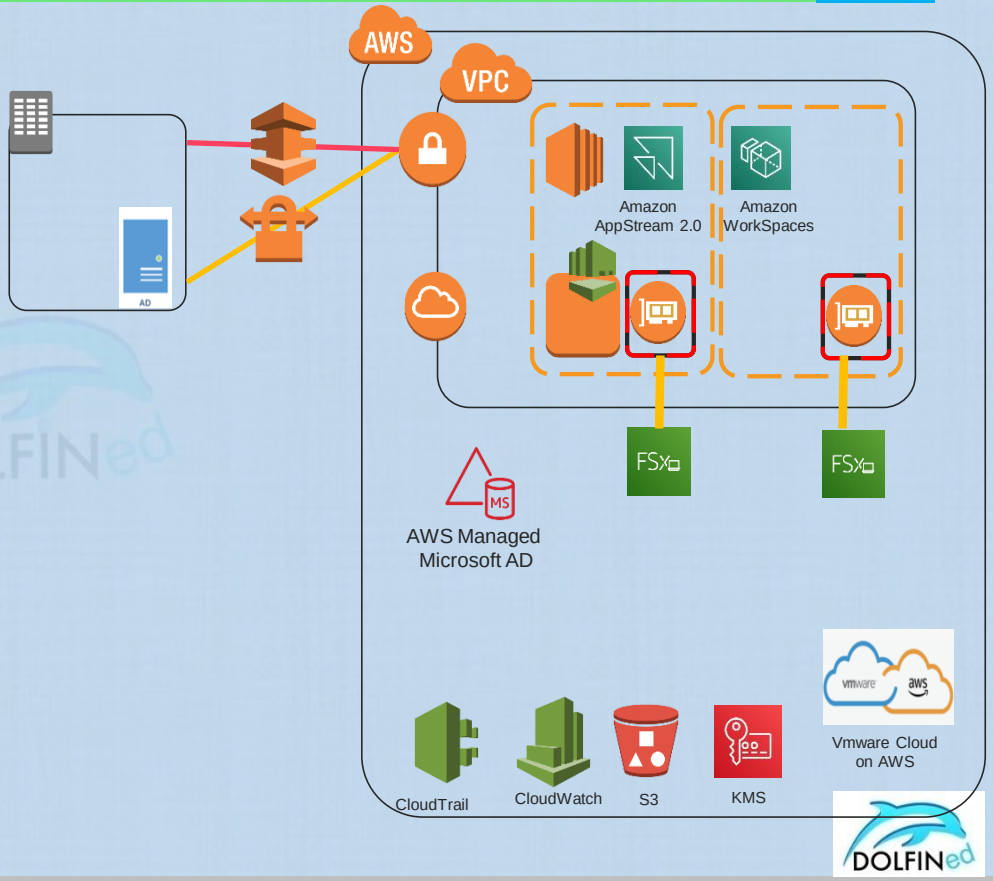
- Amazon FSx needs to work with an AWS Directory Service for Microsoft Active Directory (AWS Managed MS AD in the same VPC, different VPC in the same account using VPC peering, or a completely different account using directory sharing), or an AD that is self-managed by the client in AWS or On-premise.

Amazon FSx for Windows File Server – What is it

- Integrates with many other AWS services including IAM, CloudWatch, CloudTrail, KMS, S3, AD services

- The **file system** is the primary resource in Amazon FSx. It is the server hosting the files and folders.
  - The file system is configured with a storage amount and a throughput capacity.
  - It also has a DNS name for accessing it.

- A **file share** is a specific folder (and its subfolders) within your file system that is made accessible to the clients and compute instances



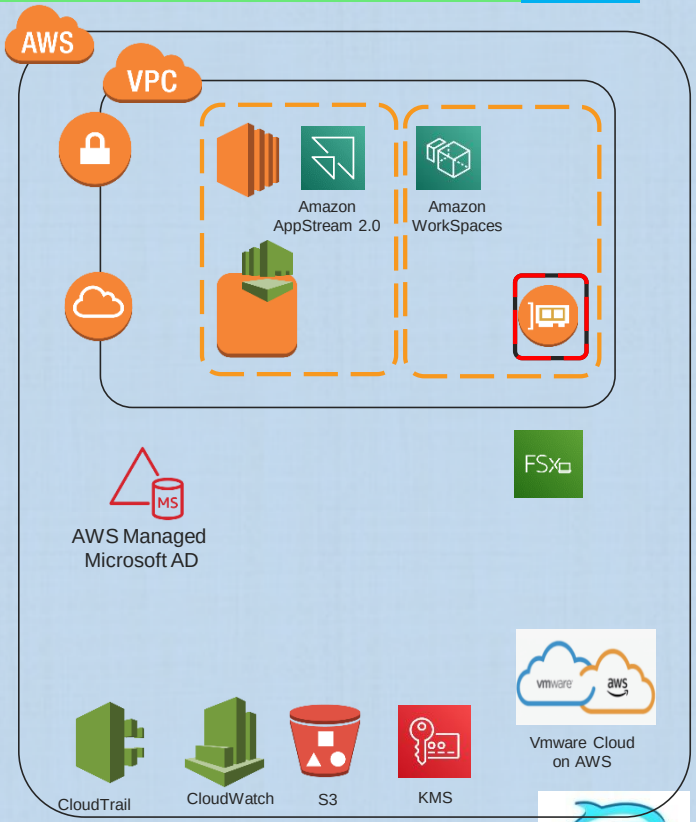
Amazon FSx for Windows File Server – Use cases

- Amazon FSx is suitable for the following use cases where a Windows shared file storage is required:
  - CRM, ERP, custom or .NET applications,
  - Home directories,
  - Data analytics,
  - Media and entertainment workflows,
  - Web serving and content management,
  - Software build environments, and
  - Microsoft SQL Server.

Amazon FSx for Windows File Server – Availability and Durability

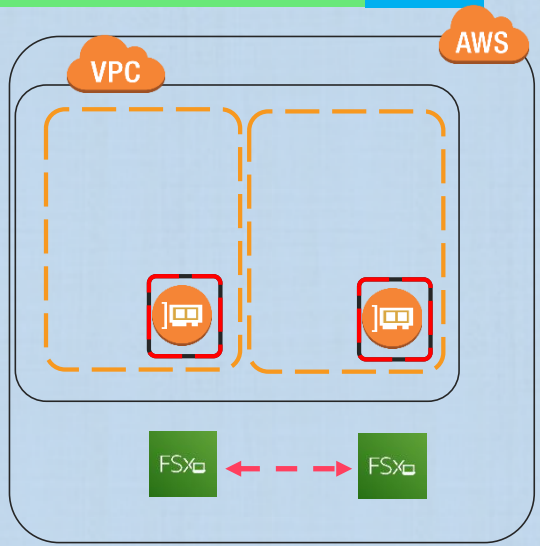
- Amazon FSx can be configured in Single-AZ or Multi AZ file systems

- **For Single AZ file systems**
  - They are designed to be highly available and durable within an AZ
  - FSx replicates data within an AZ to protect it from component level failures,
  - FSx automatically replaces infrastructure components in the event of a failure.
  - Amazon FSx takes highly durable backups (stored in S3) of the file system daily using Windows's Volume Shadow Copy Service, and also allows for taking additional backups at any point.



Amazon FSx for Windows File Server – Availability and Durability

- **For Multi-AZ file systems**
  - In addition to supporting all single AZ features, Multi AZ file system provides protection against instance failure and AZ disruption/failure
  - Amazon FSx automatically provisions and maintains **a standby file server** in a different Availability Zone.
  - Any changes written to disk in the file system **are synchronously** replicated across AZs to the standby.
  - Amazon FSx Multi-AZ feature is particularly helpful during planned system maintenance.
  - Amazon FSx automatically fails over to the secondary file server, to continue accessing data without manual intervention.
    - It fails back to the then active file server when it is brought up (takes 30 seconds to fail over or fail back)
- Windows clients automatically fail over and fail back with the file system, such that users or apps running on Windows clients automatically benefit from Multi-AZ file systems.
- Linux clients don't automatically connect to the standby file server upon a failover.



Amazon FSx for Windows File Server – Encryption

- Amazon FSx for Windows File Server always **encrypts file system data and backups at-rest** using keys you manage through AWS KMS.
  - Data is automatically encrypted before being written to the file system, and automatically decrypted as it is read.
  - These processes are handled transparently by Amazon FSx.
  - Amazon FSx uses an industry-standard AES-256 encryption algorithm to encrypt Amazon FSx data and metadata at rest.
  - The encryption key can be an AWS-managed CMK or Customer-managed CMK

- Amazon FSx **encrypts data-in-transit** using SMB Kerberos session keys, from clients that support SMB 3.0 (and higher).
  - Encryption in-transit can be enforced on all connections to the file system by limiting access to only those clients that support SMB 3.0 and higher to help meet compliance needs.
  - You can use file share-level encryption controls to have a mix of encrypted and unencrypted file shares on the same file system.

Amazon FSx

- Data Protection, Backup and Restore
- Scaling Out Storage
- Migration from On Premise to FSx
- Access from On Premise and other VPCs/Accounts/Regions
- Securing the FSx file system using Security Groups

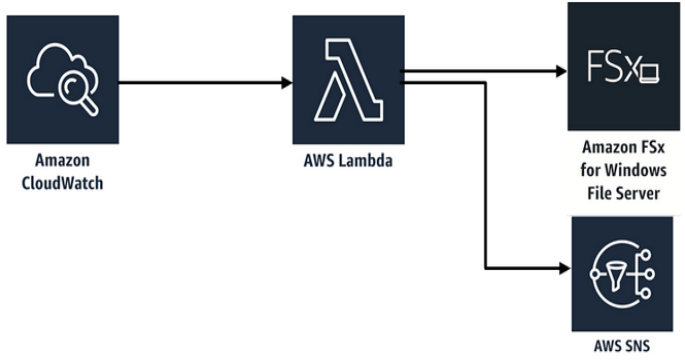
Amazon FSx for Windows File Server – Data Protection, Backup, Restore

- Amazon FSx for Lustre automatically replicates the file system's data to ensure high availability.
- In addition to that, Amazon FSx provides two options to further protect the data stored on file systems:
  - **Windows shadow copies (MS Windows Volume Shadow copy Service (VSS)) –**
    - Needs to be enabled on the file system
    - MS shadow copies are point in time snapshots of the file system stored in S3
    - When Shadow copies is enabled, users can view files, undo file changes, compare file versions by restoring to previous versions,
    - This can help in ensuring file (and folder) consistency by restoring to earlier versions
    - Administrators using Amazon FSx can easily **schedule shadow copies** to be taken periodically using Windows PowerShell commands.
    - Shadow copies are included in the backups taken of your file systems
    - Shadow copies data is stored with the file system's data and therefore they incur storage costs
      - Shadow copies are incremental snapshots

Amazon FSx for Windows File Server – Backup & Restore

- Amazon FSx **automatically takes a daily, incremental, backups** of the file systems during a daily, configurable, 30 min backup window.
- It supports backup retention and compliance needs.
- By default these automatic backups are retained for 7 days (configurable 0 – 35 days)
- Backups are stored in S3
- Available backup can be used to create a new file system by restoring a point-in-time snapshot of another file system.

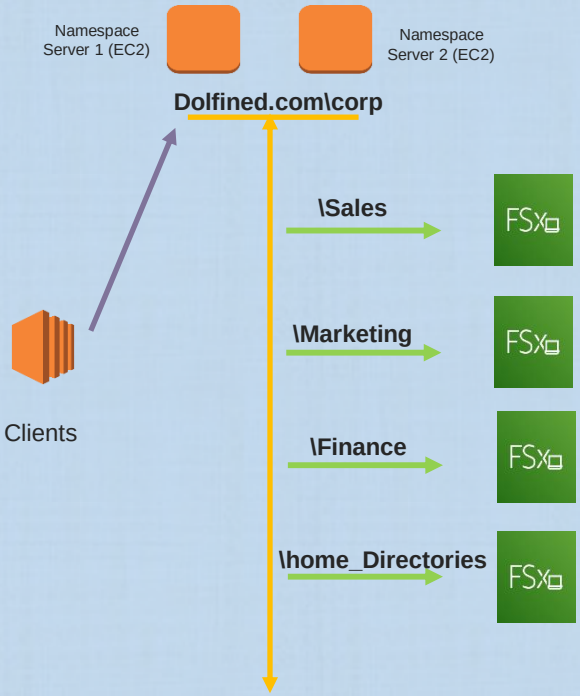
- **User-initiated (manual) backup**s are also supported with FSx.
  - Manual backups can also be automated using a custom schedule using CloudWatch events with Cron/Scheduled events to trigger a lambda function with the proper IAM roles to initiate an FSx for Windows File Server backup of the file system(s).



https://docs.aws.amazon.com/fsx/latest/WindowsGuide/custom-backup-schedule.html

Amazon FSx for Windows File Server – Scaling Out Storage and Throughput

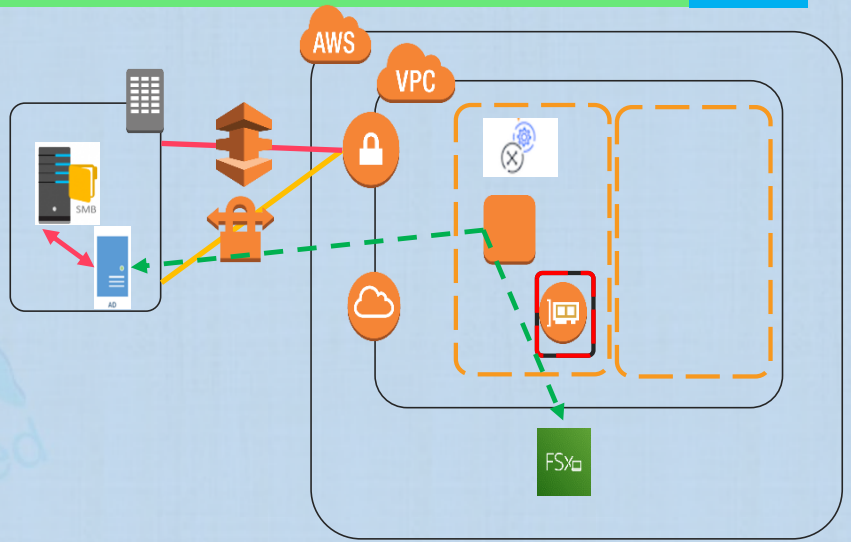
- Amazon FSx file system can scale up to 64 TBs.
- If you need to scale beyond this to hundreds of Petabytes you can use Windows MS Distributed File System (DFS) namespaces which are supported by Amazon FSx for windows file server.



source: aws.amazon.com

Amazon FSx for Windows File Server – Migrating on Premise to AWS

- FSx for Windows file servers can be accessed from on-premise through a DX or VPN connections.
- Windows's Robust File Copy (RoboCopy) can be used to copy existing data files (both the data and the full set of metadata like ownership and Access Control Lists) directly to Amazon FSx.
- To achieve this the EC2 instance must join the same AD Forest (top most logical container in AD) where all users, domains, file servers…etc.
- This allows for the use of MS Distributed File System (DFS) to replicate between on premise file servers and Amazon FSx



MS DFS is also required to replicate between multiple Amazon FSx file systems across AWS Regions for data migration, cloud-bursting, data backup, and disaster recovery workflows.

Access to Amazon FSx File Systems/shares from other VPC/Account/Region

- The Amazon FSx file system can be accessed **both from On Premise and from other VPCs/Accounts/Regions**.

- Using Amazon Direct connect or VPN to access your file system in a VPC.
  - With on-premises access, Amazon FSx can be used for:
    - Hosting user shares accessible by on-premises end-users,
    - Cloud bursting workloads to offload your application processing to the cloud,
    - Backup and disaster recovery solutions.

- Using VPC Peering, Transit Gateway other VPCs, Accounts in the same or different Regions can access the Amazon FSx file systems.
  - This way, it will be possible to share the file systems' file data sets across users and applications in multiple VPCs, AWS accounts, and/or AWS Regions.

Amazon FSx for Windows File Server – Security Group for the File System

- Clients access the FSx file system through an ENI in the VPC configured during the creation process.
  - This VPCto associate the file system with and Subnets (AZ) are specified while creating the File System
  - Clients use the DNS to access the Private IPv4 address of the ENI

- Ensure that the following is allowed through the security group rules:
  - Inbound and outbound rules to allow the following ports:
    - TCP/UDP 445 (SMB)
    - TCP 135 (RPC)
    - TCP/UDP 1024–65535 (Ephemeral ports for RPC)
    - From and to IP addresses or security group IDs associated with:
      - Client compute instances from which you want to access the file system.
      - Other file servers that you expect this file system to participate with in DFS Replication groups.
  - Outbound rules to allow all traffic to the security group ID associated with the AWS Managed Microsoft AD directory to which the file system needs to join.

Amazon FSx for Lustre

Introduction to Amazon FSx for Lustre

- What is it
- How it works
- Access to FSx for Lustre file systems
- Security and Encryption

Amazon FSx for Lustre – What is it?

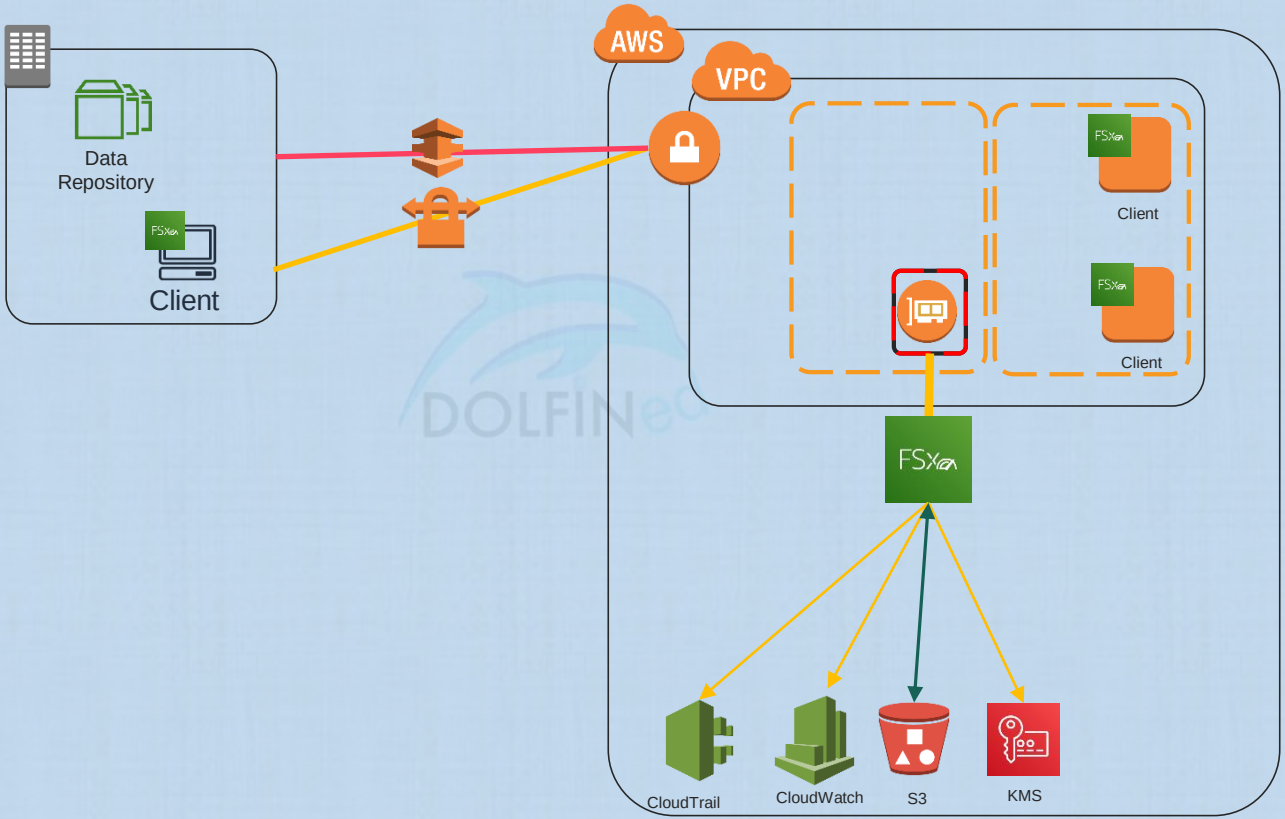
- Amazon FSx for Lustre provides **fully managed Lustre file systems** that are optimized for compute-intensive workloads requiring high performance and low latencies of scale-out, parallel file systems.

- Use cases include:
  - High-performance computing (HPC),
  - Machine learning, Electronic Design Automation (EDA)
  - Video Processing and Financial modeling

- With Amazon FSx for Lustre, in minutes, a Lustre file system can be launched that can process massive datasets at:
  - Up to **hundreds of gigabytes per second of throughput**, **millions of IOPS**, and **sub-millisecond latencies**.

- Amazon FSx for Lustre provides **high-performance storage at low cost, because it is nonreplicated, on-demand storage for short-term, compute-intensive** processing of datasets.

Amazon FSx for Lustre – What is it?



Amazon FSx for Lustre – Access to file systems

- An Amazon FSx for Lustre file system is accessed through an ENI created inside the VPC where you associated the file system
- Lustre clients (Linux machines only) are required to mount the FSx for Lustre file system to use them
  - This requires the installation of the Lustre client software on the client machines
- FSx for Lustre file system can be mounted to a Lustre Client in AWS (same VPC or in a VPC peered with the File system's VPC) , or to a Lustre client On-premise.
- For on-premise case, copy the data into the Lustre file system and run the compute-intensive workloads on in-cloud instances (not the on-premise instances).
- You can configure your EC2 instances to remount the FSx file system automatically after an instance reboot
- Amazon FSx for Lustre keeps the underlying software powering the file systems up-to-date
  - It also has a rich integration with other AWS services like Amazon S3, CloudWatch, and CloudTrail.

Amazon FSx for Lustre – Using Data from Repositories

- Amazon FSx for Lustre is designed for **short-term, compute-intensive workloads** where the long-term data is stored in a durable data repository, such as Amazon S3 or an on-premises data store.
  - An Amazon FSx for Lustre repository **isn't meant to store durable, long-term data.**
  - The file system is generally brought up only for the duration of intended compute job (typically several hours or days).

- When Amazon FSx for Lustre is used with a durable storage repository:
  - Large volumes of file data can be ingested and processed in a high-performance file system.
  - Intermediate results can be written to the data repository periodically.
    - This allows to restart the workload at any time from the latest data stored in the data repository.

Amazon FSx for Lustre – Dealing with data in S3

- Amazon FSx for Lustre is deeply integrated with Amazon S3.
- When the Amazon FSx for Lustre file system is being created, there is an option to link it to an Amazon S3 data repository
  - This means, FSx for Lustre file systems can automatically copy Amazon S3 data into the file system to run analysis for hours to days, then
    - ➢ Write results back to S3, and then delete the file system.
- Also, applications mounting the Amazon FSx for Lustre file system can seamlessly access the objects stored in the Amazon S3 buckets

Amazon FSx for Lustre – Dealing with data in On-Premise Repositories

- Amazon FSx for Lustre can be used to process data stored in on-premises data repository with in-cloud compute instances.
  - Amazon FSx for Lustre supports cloud bursting workloads with on-premises data repositories
- This can be achieved by **enabling the copying of data from on-premises Lustre clients** using AWS Direct Connect or VPN.
- Compute-intensive workloads can be run on Amazon **EC2 instances in the AWS Cloud** and then results are copied back to the on-premise data repository when the workload processing in AWS is done.
  - Processing is not done in on-premise servers

Amazon FSx for Lustre – Security and Encryption

- Like FSx for Windows File Server, an ENI is created in the VPC to associate with the File System
- Security groups are used to protect the ENI
  - Specific ports needs to be allowed Inbound and Outbound to ensure successful access to the FSx for Lustre
- **Encryption**
  - FSx service automatically encrypts data before it is written to the file system, and decrypts it as it is being read
  - FSx service manages the KMS keys
  - Amazon FSx for Lustre does not support using Amazon S3 buckets encrypted using SSE-KMS and SSE-C for data repositories.

EFS vs FSx

EFS vs Amazon FSx for Windows File Server vs Amazon FSx for Lustre

EFS vs FSx for Windows File Server vs. FSx for Lustre

EFS

- Amazon EFS is a cloud-native fully managed file system that provides simple, scalable, elastic file storage accessible from Linux instances via the **NFS protocol**.
- Can only be used **with Linux clients**/applications.

FSx for Windows File Server:

- Use it for Windows-based applications,
- Amazon FSx provides **fully managed Windows file servers** with features and performance optimized for "lift-and-shift" business-critical application workloads including home directories (user shares), media workflows, and ERP applications.
- It is accessible from **Windows and Linux** instances via the **SMB protocol.**

FSx for Lustre:

- For compute-intensive and fast processing workloads such as HPC, Machine learning, EDA, and Media processing,
- Amazon FSx for Lustre, provides a file system that's optimized for performance, with input and output stored on Amazon S3.
- Only for **Linux clients**