GLOBAL CONTENT DELIVERY AND OPTIMIZATION

Cloudfront Architecture

CloudFront is a Content Delivery network (CDN) within AWS.

- Origin: The source location of your content (can be S3 or custom origin)
- Distribution: The 'configuration' unit of CloudFront
- Edge Locations: Local cache of your data
- Regional Edge Cache: Larger version of an Edge Location, provides another layer of caching NOTE: Can use SSL certs with CloudFront as it integrates with AWS Certificate Manager (ACM) for HTTPS NOTE: CloudFront is for DOWNLOAD operations only, uploads go directly to Origin with NO WRITE CACHING. CloudFront only does Read-Only type of caching.

CloudFront (CF) - Behaviors

CloudFront Behaviors control much of the TTL, protocol and privacy settings within CloudFront

- Behaviors have caching options, and restrict viewer access options
- Distribution can have multiple behaviors, but there is one Default(*) behavior

CloudFront - TTL and Invalidations

How CloudFront handles object expiry and invalidation

- TTL: Object considered not expired when within its TTL
- Invalidations: More frequent cache hits result in lower origin load -> better performane of application (goal of CloudFront) Problem: If old version of object is cached and newer version is uploaded in origin -> cached version has to expire in edge location cache because edge location is not notified when newer version exists. Only when cache expires and a check at the origin is performed. TTL (Time To live) value can be set in 3 ways:

1. Default TTL: 24 hours (validity period) (EXAM) -> is used when no TTL is specified (eg in headers)
2. Minimum TTL / Maximum TTL can be set (EXAM) -> are limiters for the specified TTL: values below minimum or above maximum are limited to minimum or maximum TTL -> min/max TTL value is used if outside of range
3. Specific TTLs using Origin Headers that can be set in Custom Origins or S3 (object metadata)

- Origin Header: Cache-Control max-age(seconds) ->specifies seconds until expiry
- Origin Header: Cache-Control s-maxage(seconds) -> specifiesseconds until expiry (same as max-age)
- Origin Header: Expires(DateTime) -> specifies Date & Time for expiry EXAM: need to know headers

2. Cache Invalidation: Are performed on a distribution and applied to all Edge Locations (takes time to implement)

- Immediately expires objects regardless of their TTL based on the invalidation pattern that you specify e.g. /images/whiskers1.jpg, /images/whiskers* (wildcard), /images/* or /* (all objects are immediatly

invalidated)
- Instead of Cache Invalidation, you can use Versioned File Names (whiskers1_v1.jpg // _v2.jpg // _v3.jpg) which is a better practice (always in EXAM !!!)
    - versioning is best practice because new filename makes sure latest version is always recovered
    - and logging is more effective because you know which actual file is being used
    - you keep all version of all versions !! Don't confuse versioned filenames with S3 object versioning (is different)

AWS Certificate Manager

A service which allows the creation, management and renewal of certificates. It allows deployment of certificates onto supported AWS services, mainly `CloudFront`,`ELB` but also Cognito, CloudFormation, API GateWay, Elastic Beanstalk

- HTTP was/is simple and insecure.
- HTTPS introduced a layer of encryption to HTTP encrypting the data in-transit.
- HTTPS also allows for servers to prove their identity with Certificates using SSL/TLS
- These SSL/TLS certificates get signed by a trusted authority AKA Chain of Trust
- ACM can be a Public or Private Certificate Authority (CA)
- Private CA: applications need to trust your private CA: private CA has to be added manually in computer build or with a policy that configures this trust
- Public CA: Browsers trust a list of providers, which can trust other providers (chain of trust)
- S3 doesn't use ACM
- EXAM - ACM can generate or import certificates. If self-generated, it can auto-renew. If imported, you are responsible for renewal
- EXAM - Certificates can only be deployed out to SUPPORTED services (Eg. pretty much just CloudFront and ALBs, `specifically NOT EC2`)
- EXAM - Certificates are always stored encrypted in ACM and deployed in a secure way in supported services
- EXAM - Reminder, cannot use ACM with EC2: because with root access you can see unencrypted certificate -> goal of ACM is to secure the storage and deployment of your certificates
- EXAM - ACM is regional service: `Certs can't leave the region they are generated/imported into` cert are region specific
- EXAM - To use a cert with an ALB in ap-southeast-2, you need a CERT IN ACM in that region: ap-southeast-2 (ALWAYS EXAM question)
- EXAM -- GLOBAL services like CloudFront operate as though within us-east-1 (like all global services) -> so you need ACM is us-east-1

- EXAM - ACM comes up a lot about which certificates can be used in which regions (problem solving questions) -> solution in same region as service or us-east-1 if global service.

Cloudfront and SSL/TLS

- Each CloudFront distribution gets a Default Domain Name (CNAME DNS record) when created: eg d111111abcdef8.cloudfront.net
- SSL supported by default as long as you use the Default Domain name because certificate uses *.cloudfront.net cert (* = wildcard)
    - Can have Alternate Domain Names Eg. cdn.catagram.com using DNS provider like route 53 that points to default domain
    - You need to verify ownership using a matching certificate for the alternate domain name
    - you can allow both HTTP or HTTPS traffic. Or point HTTP traffic to HTTPS. Or allow only HTTPS
    - for HTTPS you either generate a certificate or import in ACM a cert that must be in us-east-1, Eg. CloudFront (Will come up in the exam: CloudFront = us-east-1)

When using CloudFront, you actually have TWO SSL connections:

1. Viewer => CloudFront: PUBLIC certificate issued by a trursted Certificate Authority(Comodo, DigiCert or ACM (us-east-1)
2. CloudFront => Origin: if S3 no need for certificate because S3 handles this. If ALB or Custom Origin (EC2 or On-Prem): public trusted CA authority or ACM for ALB, trusted CA authority only for Custom Origin because no support for ACM.

- Both need valid PUBLIC certificates. Self-signed certs will not work with CloudFront (exam)

CloudFront (CF) - Origin Types & Origin Architecture

CloudFront origins store content distributed via edge locations.

- The features available differ based on using S3 origins vs Custom origins

Origin Categories

- S3 Buckets
- AWS Media Package Channel Endpoints
- AWS Media Store Container Endpoints
- Everything else (web servers); Custom Origins. Note: An S3 as a static website is treated not as S3 but as a Web Server aka custom origin
- EXAM: if question about control of protocol (HTTP/HTTPS), port (80/443) or certificate version -> Custom Origin is the answer because you can't modifiy this for S3

DEMO - CloudFront (CF) - Adding a CDN to a static Website

Implement a CloudFront distribution using an S3 bucket as the origin.

Steps:

1. 1-click deploy. At this point, site in demo is static S3

2. Host site on CloudFront: CloudFront > Create Distro > Origin domain, select s3 bucket "cfands3-top10cats[...]" > Cache key and origin requests, select "CachingOptimized" > Default root object "index.html" > Deploy/Create Note: You can now access CDN cached files that are cached in Edge Locations. You can Invalidate (billed per Inval., do it less frequently). In this part, we changed merlin.jpg file to new image and played with caching in CloudFront.
3. Add custom Domain name: Didn't have custom Domain made in R53 so had to watch at this point
4. SSL via ACM:

Securing CF and S3 using OAI

Origin Access Identities (OAI) are a feature where virtual identities can be created, associated with a CloudFront Distribution and deployed to edge locations.

- OAI can only be used when using S3 bucket as an origin (not custom origins like: EC2, ALB, On-Premise servers, static S3 website is also a custom origin !!)
- Access to an S3 bucket can be controlled by using these OAI's - allowing access from an OAI, and using an implicit DENY for everything else.
- They are generally used to ensure no direct access to S3 objects is allowed when using private CF Distributions.

Origin-side Security

S3 Origin

You can have S3 Origin, or Custom Origin (S3 static website). OAI's are a type of Identity that are associated with CF Distros to 'become' that OAI to be used in S3 Bucket Policies ; S3 Origin is locked down to only be accessible by this OAI, all else Implicit Denied.

- Explicit Allow on S3 Policy for OAI
- Best practice: Create 1 OAI for 1 CF Distro

Custom Origin - Custom Headers or Traditional Firewall

There are 2 ways to secure Custom Origins:

1. Utilize custom headers and Viewer control policy (HTTPS), then a related Origin Control Policy that has a required Custom Header attached. Origin requires the custom header
2. Traditional Security Method: IP Ranges of CloudFront with access to a Firewall

CloudFront - Private Distribution & Behaviours

Signed URLs and Signed Cookies

CloudFront can be Public access or Private Access (requiring signed Cookie or signed URL)

- CF Distro is created with ONE behavior for the whole distro, which is public or private
- It'll end up having multiple behaviors, private and public Old way required a CloudFront KEY created by account root user and account is added as a TRUSTED SIGNER New way is TRUSTED KEY GROUPS added which can be managed with CloudFront API

Signed URLs VS Cookies

- SignedURLs provided access to ONE object
- Signed Cooked for access to GROUPS of objects
- SignedURLs if client doesn't support cookies
- Maintain your URL? Signed Cookies

DEMO - CloudFront (CF) - Using Origin Access Control (OAC) (new version of OAI) - SKIPPING

Lambda@Edge

Lambda@Edge allows cloudfront to run lambda function at CloudFront edge locations to modify traffic between the viewer and edge location and edge locations and origins.

- Not the full feature-set of Lambda: only Node.js and Python as run times. No VPC based resources. Lambda Layers not supported. Different limits than Lambda.
- You can run the Lambda@Edge functions at 4 points: 1. Viewer request 2. Origin request 3. Origin response 4. Viewer response

Lambda@Edge Use Cases:

- A/B Testing (Viewer Request)
- Migration between S3 Origins (Origin Request)
- Different Objects delivered based on Device (Origin Request)
- Content by Country (Origin Request)

AWS Global Accelerator

AWS Global Accelerator is designed to improve global network performance by offering entry point onto the global AWS transit network as close to customers as possible using Anycast IP addresses

- Designed to optimize the flow of data from user to AWS infrastructure
- Reduce number of "hops"
- When to use CloudFront VS Global Accelerator?
    - CloudFront is focused on improving content delivery to end-users via Edge Caching. Question mentions caching? Prolly CloudFront
    - Global Accelerator is focused on optimizing traffic routing to your applications -- by entering the network closer to the customer. Question mentions TCP/UDP? Prolly Global Accelerator Global Accelerator is a good fit for non-HTTP use cases, such as gaming (UDP), IoT (MQTT), or Voice over IP, as well as for HTTP use cases that specifically require static IP addresses or deterministic, fast regional failover.

Global Accelerator Resource: https://aws.amazon.com/global-accelerator/faqs/