Cantrill Review – Global Delivery and Optimization

CloudFront Architecture

* Terms
  + Origin – the source location of your content
    - S3 Origin or Custom Origin
      * Custom Requires Public IPv4
  + Distribution – the configuration unit of CloudFront
  + Edge Location – local cache of your data
    - Can’t deploy infrastructure in Edge Locations
  + Regional Edge Cache – Large version of an edge location
    - Cache less frequently accessed data
* Integrates with ACM for SSL and supports HTTPS
* Upload direct to origins – no caching
  + Client upload to cache location is NOT supported.
    - Download ONLY capabilities
* Behavior – a configuration within a distribution
  + Pattern matching
* Origins are linked to behaviors
  + Origins are used by behaviors as content sources
* Distributions have 1 or more behaviors
* A distribution can have many behaviors which are configured with a path pattern. If requests match that pattern, that behavior is used, otherwise the default is used.
* Origins, Origin Groups, TTL, Protcol Policies, Restricted Access are configured via Behaviors

CloudFront Behaviors

* Set at behavior level within a distribution
  + Lambda Functions
  + TTL
  + SSL
  + Restrict Viewer Access – force signed cookie or signed URL usage

CloudFront – TTL and Invalidations

* More frequent cache HITS = lower origin load
* Default TTL (behavior) = 24 hours (validity period)
* You can set Minimum TTL and Maximum TTL values
  + Applied at behavior level
  + Influence and apply to object level configurations
* Object level configurations –
  + Origin Header – Cache-Control max-age (seconds)
  + Origin Header – Cache-Control s-maxage (seconds)
  + Origin Header – Expires (Date & Time)
  + Note – all header configurations (bucket-level) are restricted by the Min/Max TTL applied to behavior.
    - If no Min/Max TTL exists, default (24 hours), is used.
* Custom Origin or S3 Origin
  + Custom Origin – TTL applied in application or web server
  + S3 Origin – applied via Object metadata
* Cache Invalidations
  + Performed on a distribution
  + Applies to all edge locations - takes time, not immediate
  + Immediately expires TTL based on /path
    - Ex.
      * /images/whiskers1.jpg
      * /images/whiskers\*
      * /images
      * /\* - invalidates all objects which are cached by a distribution
  + Costs regardless
    - Only use to correct errors
  + Version File Names
    - Whiskers1\_v1.jpg // \_v2.jpg // \_v3.jpg
    - Logging is more effective, know which object was actually used
    - Less expensive – no continued invalidation
    - Different from S3 Versioning
      * Different file names for different file names and different data.
      * Cached independently in edge locations

AWS Certificate Manager (ACM)

* HTTP – Simple and Secure
* HTTPS – SSL/TLS layer of encryption added to HTTP
  + Data is encrypted in transit
* Certificates prove identity
* Chain of trust – signed by a trusted authority
* ACM lets you run a public or private Certificate Authority (CA)
* Private CA – Applications need to trust your private CA
* Public CA – Browsers trust a list of providers, which can trust other providers (chain of trust)
* ACM can generate or import Certificates
  + If generated, it can automatically renew
  + If imported, you are responsible for renewal (must renew with external CA + reupload)
* Certificates can be deployed ONLY to supported services
* Supported AWS Services ONLY
  + CloudFront and ALBs
  + NO EC2 certificates
  + No S3 certificates (S3 Origins handles certificates natively)
* ACM is a regional service
* Certs cannot leave the region they are generated or imported in
* To use a cert with an ALB in ap-southeast-2, you need a cert in ACM in ap-southeast-2
* Global Services such as CloudFront operate as though within us-east-1
  + All global services default to us-east-1
  + The cert is deployed by ACM to the distribution – then the distribution sends to Edge Locations
  + Edge Locations use certs to prove identity, establish trust, and secure connections

CloudFront and SSL/TLS

* CloudFront Default Domain Name (CNAME)
  + Ex. <https://d111111111abcdef8.cloudfront.net/>
* SSL Supported by default – \*cloudfront.net cert
* Alternate Domain Names supported (CNAME)
  + Ex. Cdn.catagram.com
* Verify Ownership (optionally HTTPS) using a matching certificate
* Generate or import in ACM – us-east-1
* Supported Viewer Protocols
  + HTTP or HTTPS
  + HTTP => HTTPS
  + HTTPS Only
* Two SSL connections:
  + Viewer => CloudFront
  + CloudFront => Origin
  + Note – BOTH need valid PUBLIC certificates (and intermediate certs where applicable)
    - Self-signed certificates are NOT supported in CloudFront
* CloudFront and SNI
  + SNI – Server Name Indication
    - Tells client which domain name it is trying to reach
    - TLS extension – allowing a host to be included
    - One IP can host many HTTPS websites
    - Older browsers don’t support SNI
      * CF charges extra for dedicated IP
  + SNI Mode or Dedicated IP
    - SNI is free
    - Dedicated IP is 600 USD per month
* Origins need to have certificates issued by a trusted certificate authority (CA)
  + ALB can use ACM, others need to use an external generated certificate
    - NO self-signed certificates
  + S3 Origins handle certificates natively
* Certificate applied to CloudFront needs to match the DNS name of the Origin and vise versa

CloudFront Origin Types and Architecture

* Origin Group – can be used by behaviors to group multiple Origins – added resiliency
* S3 bucket has one set of features
  + If S3 statis website is used for CloudFront – CloudFront treats it as a webserver (custom) origin
  + Origin access – restrict the access to only allow CloudFront Distribution
    - Legacy config – Legacy Access Identities

Securing CF and S3 using OAI

* Origin Access Identity (OAI)
  + Only applicable for S3 origins
    - Not using the static website feature of S3
  + An OAI is a type of identity
  + It can be associated with CloudFront Distributions
  + CloudFront becomes that OAI
  + That OAI can be used in S3 Bucket Policies
    - Best practice is default deny except OAI associated with CloudFront – explicityly allow CloudFront, implicit deny all other traffic
  + Once OAI is associated with the distribution, accesses are FROM the OAI
  + OAI can be associated with multiple distributions
    - Best practice is 1:1 OAI:CF Distribution
* Security Custom Origins
  + Custom Headers –
    - Configure HTTPS viewer policy
    - Require all requests to have the header.
      * Injected at the CF Edge Location
  + Use same location as origin
    - Origin protocol policy
  + Specify CloudFront IP\_Ranges allowed in WAF
  + Custom Origins can use ither approach or a combination of both

CloudFront Private Distributions & Behaviors

* CloudFront Private Behaviors, Signed URLS & Cookies
* CloudFront Distributions
  + Public – Open access to objects
  + Private – requests require Signed Cookie or URL
  + Distributions are created with a single behavior
  + 1 Behavior – Whole Distribution Public or Private
  + Multiple Behaviors – each is Public or Private
  + Old – a CloudFront Key is created by an Account Root User
  + Old – THE ACCOUNT is added as a Trusted Signer for the Private distribution/behavior
  + New – Trusted Key Group(s)
    - More secure – not using trusted signer by account root user
    - Associate a higher number of keys for the distribution/behavior

CloudFront Signed URLs vs. Signed Cookies

* Signed URLs – provide access to ONE object
* Historically RTMP distributions couldn’t use cookies – no longer a limitation
* Use URL’s if your client doesn’t support cookies
* Signed Cookies – provide access to groups of objects
* Use Cookies for groups of files/all files of a type – e.g. all cat GIFs
* Use Cookies if maintaining application URLs is important

Lambda@Edge

* You can run lightweight Lambda at Edge Locations
* Adjust data between the Viewer & Origin
* Currently supports Node.JS and Python
* Run in the AWS Public Space (Not VPC)
* Layers are not supported
* Different Limits vs. Normal Lambda Functions
* Viewer Request –
  + After CF receives a request from a viewer (customer)
* Origin Request –
  + Before CF forwards the request to an origin
* Origin Response –
  + After CF receives a response from an origin
* Viewer Response –
  + Before a response is forwarded to the viewer (customer)
* Viewer Limitations – 128MB + 5 seconds
* Origin Limitations – Normal Lambda MB + 30 seconds
* Use cases
  + A/B testing – viewer request
  + Migration Between S3 Origins – Origin Request
  + Different Objects based on Device Type – Origin Request
  + Vary Content Display by Country - Origin Request

Global Accelerator

* 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
* 2x ANYCAST IP Addresses
  + 1.2.3.4 & 4.3.2.1
* ANYCAST IPs allow a single IP to be in multiple locations.
  + Routing moves traffic to closest location
* Traffic initially uses public internet & enters a Global Accelerator Edge Location
  + From the edge, data transits globally across the AWS Global Backbone network.
  + Less Hops, directly under AWS control
  + Significantly better performance vs. CloudFront
* Key Concepts
  + Moves the AWS network closer to customers
    - CF moves the content by caching in locations closest to customers
  + Connections enter at edge – using anycast IPs
  + Transit over AWS backbone to 1+ locations
  + Can be used for NON HTTP/S (TCP/UDP) – Difference from CF
  + Doesn’t cache anything
  + Doesn’t understand HTTP/S