**Content Delivery Network**

* You want to deliver content to your global audience
* Content Delivery Networks distribute content to multiple edge locations around the world
* AWS provides 200+ edge locations around the world
* Provides high availability and performance

**Amazon CloudFront**

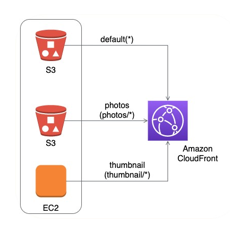
* How do you enable serving content directly from AWS edge locations?
  + Amazon CloudFront (one of the options)
* Serve users from nearest edge location (based on user location)
* Source content can be from S3, EC2, ELB and External Websites
* If content is not available at the edge location, it is retrieved from the origin server and cached
* No minimum usage commitment
* Provides features to protect your private content

**Amazon CloudFront**

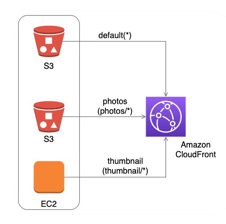
* Use Cases
  + Static web apps. Audio, video and software downloads. Dynamic web apps
  + Support media streaming with HTTP and RTMP
* Integrates with
  + AWS Shield to protect from DDoS attacks
  + AWS Web Application Firewall (WAF) to protect from SQL injection, cross- site scripting, etc
* Cost Benefits
  + Zero cost for data transfer between S3 and CloudFront
  + Reduce compute workload for your EC2 instances

**Amazon CloudFront Distribution**

* Create a CloudFront Distribution to distribute your content to edge locations
  + DNS domain name - example abc.cloudfront.com
  + Origins - Where do you get content from? S3, EC2, ELB, External Website
  + Cache-Control
    - By default objects expire after 24 hours
    - Customize min, max, default TTL in CloudFront distribution
    - (For file level customization) Use Cache-Control max-age and Expires headers in origin server
* You can configure CloudFront to only use HTTPS (or) use HTTPS for certain objects
  + Default is to support both HTTP and HTTPS
  + You can configure CloudFront to redirect HTTP to HTTPS

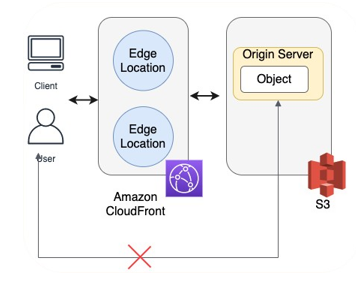


**Amazon CloudFront - Cache Behaviors**



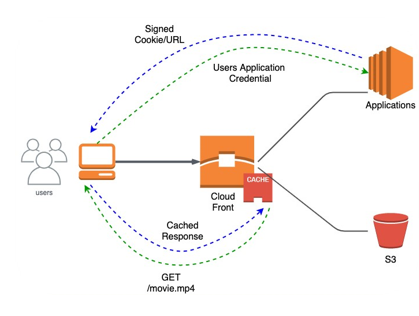
* Configure different CloudFront behavior for different URL path patterns from same origin
  + Path pattern(can use wild cards - \*.php, \*.jsp),
  + Do you want to forward query strings? Should we use https?
  + TTL

**Amazon CloudFront - Private content**



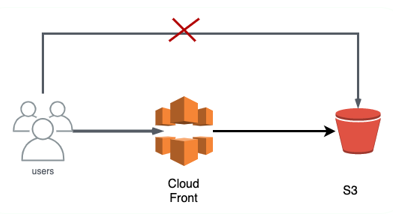
* Signed URLs
* Signed cookies using key pairs Origin Access Identities(OAI)
  + Ensures that only CloudFront can access S3
  + Allow access to S3 only to a special CloudFront user

**Amazon CloudFront - Signed URLs and Cookies**



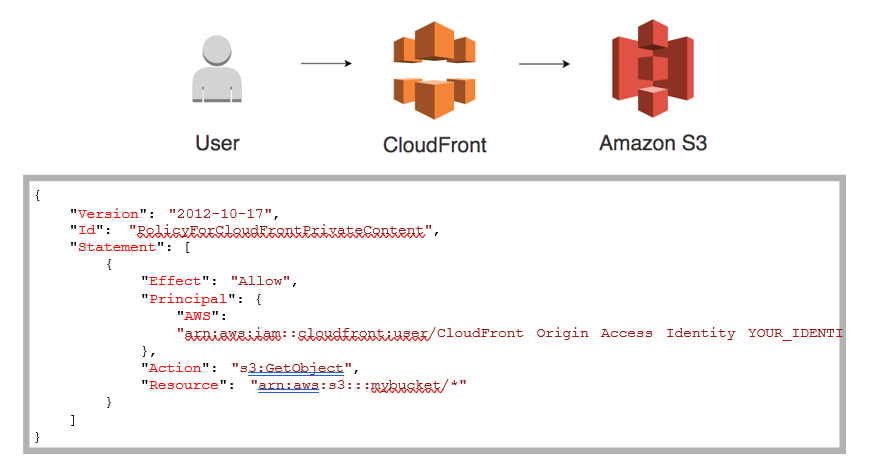
* Signed URLS
  + RTMP distribution
  + Application downloads (individual files) and
  + Situations where cookies are not supported
* Signed Cookies
  + Multiple files (You have a subscriber website)
  + Does not need any change in application URLs

**Amazon CloudFront - Origin Access Identities(OAI)**



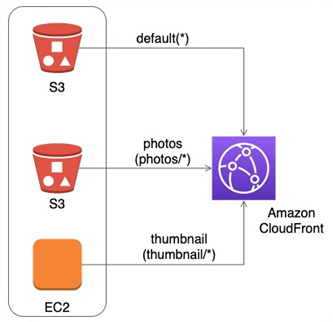
* Only CloudFront can access S3
* Create a Special CloudFront user - Origin Access Identities(OAI)
* Associate OAI with CloudFront distribution
* Create a S3 Bucket Policy allowing access to OAI

**Bucket Policy - S3 ONLY through Cloud Front**



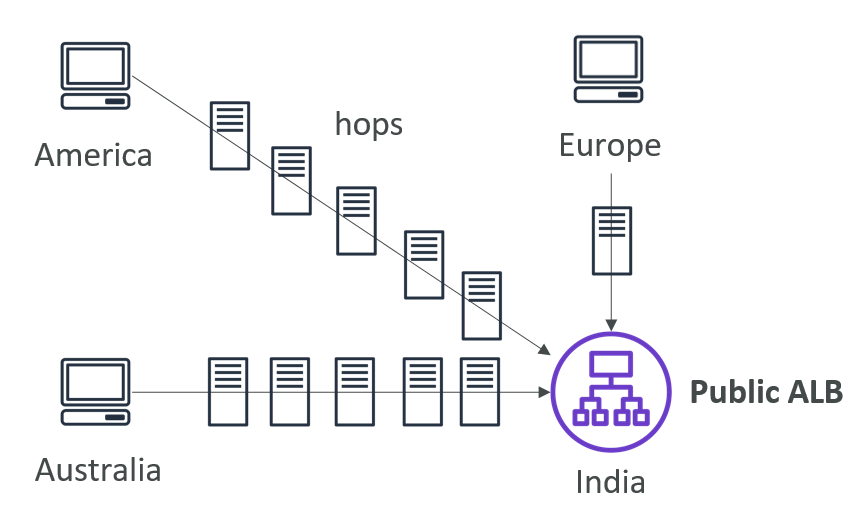
**Amazon CloudFront - Remember**

* Old content automatically expires from CloudFront
* Invalidation API - remove object from cache
  + REMEMBER : Designed for use in emergencies
* Best Practice - Use versioning in object path name
  + Example : /images/profile.png?version=1
  + Prevents the need to invalidated content
* Do not use CloudFront for
  + all requests from single location
  + all requests from corporate VPN
* Scenario: Restrict content to users in certain countries
  + Enable CloudFront Geo restriction
  + Configure Whitelist(countries to be allowed) and Blacklist(countries to be blocked)



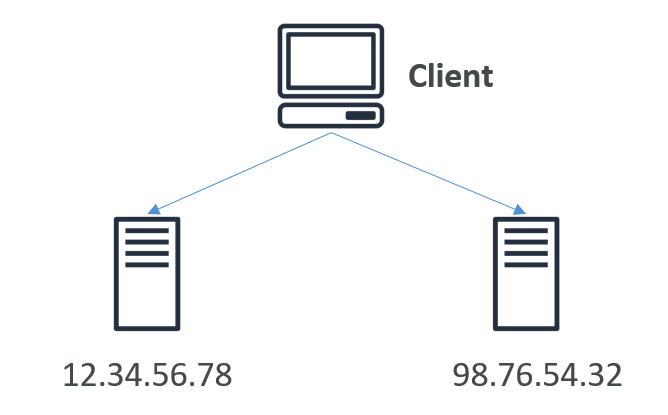
**Global users for our application**

* You have deployed an application and have global users who want to access it directly.
* They go over the public internet, which can add a lot of latency due to many hops
* We wish to go as fast as possible through AWS network to minimize latency

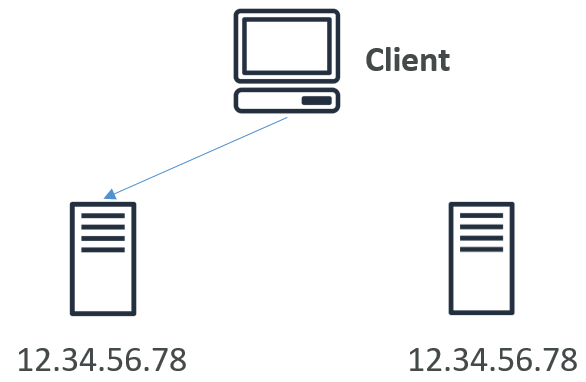


**Unicast IP vs Anycast IP**

• Unicast IP: one server holds one IP address

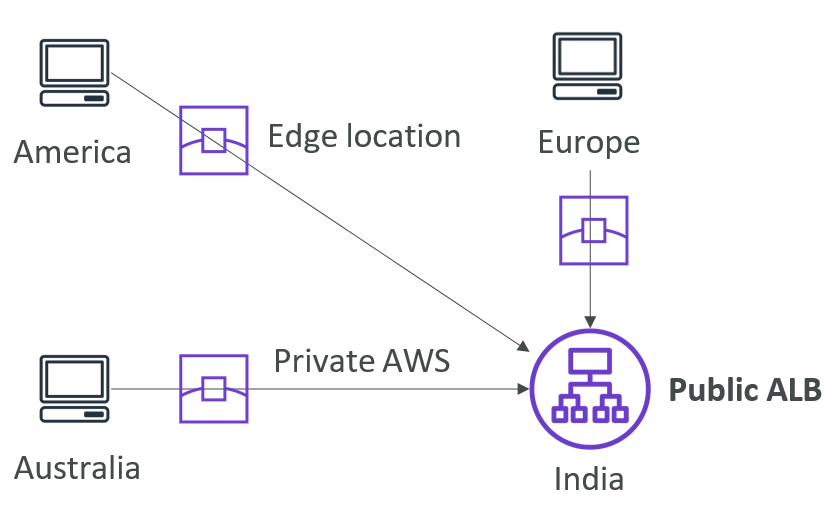


• Anycast IP: all servers hold the same IP address and the client is routed to the nearest one

****

**AWS Global Accelerator**

* Leverage the AWS internal network to route to your application
* 2 Anycast IP are created for your application
* The Anycast IP send traffic directly to Edge Locations
* The Edge locations send the traffic to your application



**AWS Global Accelerator**

* Works with Elastic IP, EC2 instances, ALB, NLB, public or private
* Consistent Performance
  + Intelligent routing to lowest latency and fast regional failover
  + No issue with client cache (because the IP doesn’t change)
  + Internal AWS network
* Health Checks
  + Global Accelerator performs a health check of your applications
  + Helps make your application global (failover less than 1 minute for unhealthy)
  + Great for disaster recovery (thanks to the health checks)
* Security
  + only 2 external IP need to be whitelisted
  + DDoS protection thanks to AWS Shield

**AWS Global Accelerator vs CloudFront**

* They both use the AWS global network and its edge locations around the world
* Both services integrate with AWS Shield for DDoS protection.
* CloudFront
  + Improves performance for both cacheable content (such as images and videos)
  + Dynamic content (such as API acceleration and dynamic site delivery)
  + Content is served at the edge
* Global Accelerator
  + Improves performance for a wide range of applications over TCP or UDP
  + Proxying packets at the edge to applications running in one or more AWS Regions.
  + Good fit for non-HTTP use cases, such as gaming (UDP), IoT (MQTT), or Voice over IP
  + Good for HTTP use cases that require static IP addresses
  + Good for HTTP use cases that required deterministic, fast regional failover