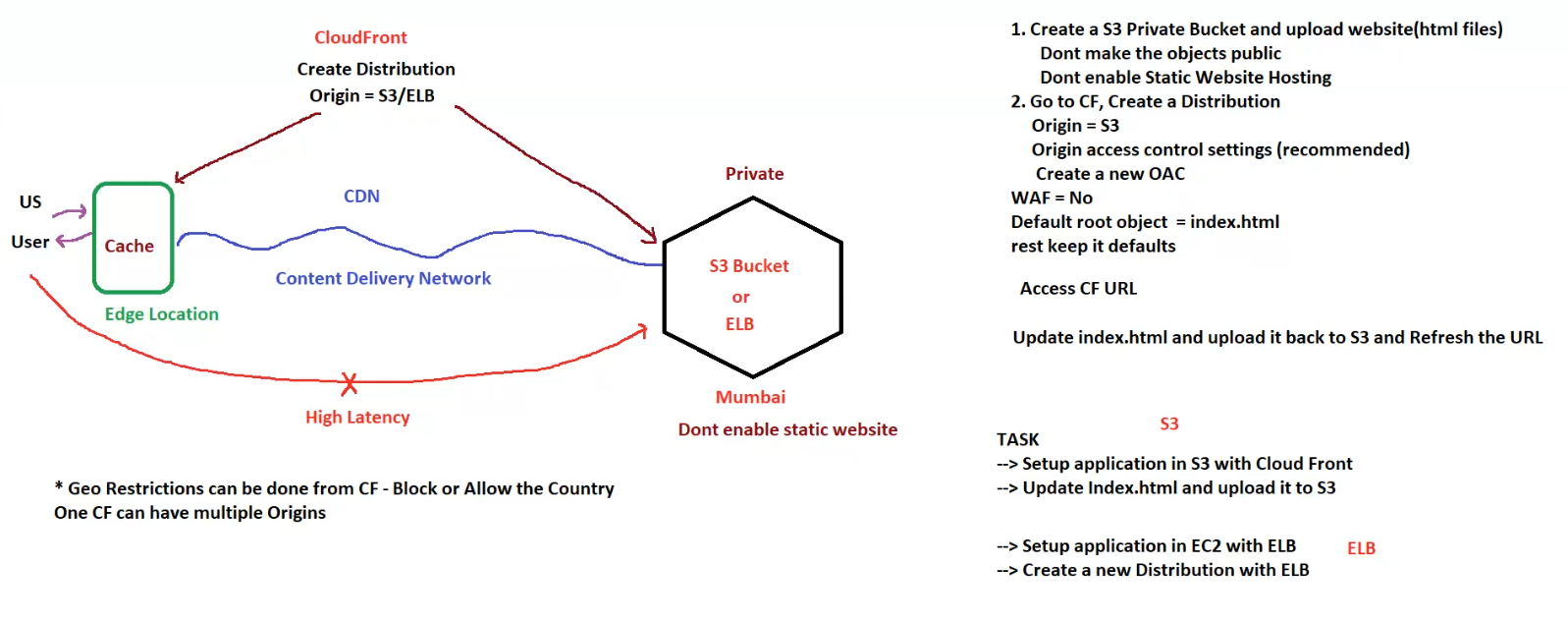
DevOps

[Cloud Front 2](#_Toc193690045)

# Cloud Front

**I. Overview**

* **Objective:** To deliver website content efficiently and securely using AWS CloudFront.
* **Key Components:**
  + **Amazon S3 (Simple Storage Service):** For storing the website's static files (HTML, CSS, JavaScript, images, etc.).
  + **AWS CloudFront (Content Delivery Network):** For caching and distributing the website content globally.
  + **Origin Access Control (OAC):** For secure access to S3 content from CloudFront.
  + **Elastic Load Balancer (ELB):** For distributing traffic to EC2 instances (optional, used if the application is running on EC2).

**II. Setting up with S3**

1. **Create an S3 Bucket:**
   * Create a new S3 bucket to store your website files.
   * **Important:** Make the bucket **private**. Do not make the objects (files) within the bucket publicly accessible.
   * Do not enable Static Website Hosting on the S3 bucket.
2. **Upload Website Files:**
   * Upload your website's HTML, CSS, JavaScript, image, and other files to the S3 bucket.
3. **Create a CloudFront Distribution:**
   * Go to the CloudFront service in the AWS Management Console.
   * Create a new CloudFront distribution.
   * **Origin:** Set the origin to your S3 bucket.
   * **Origin Access Control (OAC):**
     + Choose the "Origin access control settings (recommended)" option.
     + Create a new Origin Access Control (OAC). This allows CloudFront to securely access the S3 bucket.
   * **Web Application Firewall (WAF):**
     + Set WAF to "No" for basic setups. You can add WAF later for enhanced security.
   * **Default Root Object:**
     + Set the default root object to index.html. This tells CloudFront to serve the index.html file when a user accesses the root URL.
   * **Other Settings:**
     + Keep the other settings at their defaults for a basic setup.
4. **Access the Website:**
   * Once the CloudFront distribution is deployed, access your website using the CloudFront distribution's domain name.
5. **Updating the Website:**
   * To update your website, modify the index.html (or other) files in your local environment.
   * Upload the updated files back to the S3 bucket.
   * Refresh the website using the CloudFront URL to see the changes.

**III. Setting up with EC2 and ELB (Optional)**

1. **Set up Application on EC2:**
   * If your application requires server-side processing, deploy it on Amazon EC2 instances.
2. **Configure Elastic Load Balancer (ELB):**
   * Create an Elastic Load Balancer (ELB) to distribute traffic across your EC2 instances.
   * Configure the ELB to point to your EC2 instances.
3. **Create a CloudFront Distribution:**
   * Create a new CloudFront distribution.
   * **Origin:** Set the origin to the ELB.
   * **Note:** You can have multiple origins within a single CloudFront distribution (e.g., S3 for static assets and ELB for dynamic content).

**IV. CloudFront Features and Considerations**

* **Content Delivery Network (CDN):** CloudFront caches your website's content at edge locations around the world, reducing latency for users.
* **Edge Locations:** CloudFront uses edge locations to store cached content closer to users, improving performance.
* **Caching:** CloudFront caches content to reduce the load on your origin server (S3 or ELB).
* **Geo Restrictions:** You can use CloudFront's geo-restriction feature to block or allow access to your website from specific countries.
* **High Latency:** The image indicates "High Latency" between the user in the US and a hypothetical Mumbai server if CloudFront isn't used. CloudFront resolves this by caching content closer to the user.
* **Private Content:** Using Origin Access Control (OAC) ensures that only CloudFront can access the S3 bucket, enhancing security.
* **Don't enable static website hosting:** Since we are using cloudfront, we don't need to enable static website hosting on S3.

**V. Task Summary**

* Set up an application in S3 with CloudFront.
* Update the index.html file and upload it to S3.
* (Optional) Set up an application in EC2 with ELB.
* (Optional) Create a new CloudFront distribution with ELB as the origin.

**VI. Key Benefits of Using CloudFront**

* **Improved Performance:** Reduced latency and faster load times for users.
* **Scalability:** CloudFront automatically scales to handle traffic spikes.
* **Security:** Enhanced security through features like OAC and integration with AWS WAF.
* **Cost-Effectiveness:** Reduced bandwidth costs by caching content at edge locations.
* **Global Reach:** Distribute content to users worldwide.