Exercise 2.1: Creating a CloudFront Distribution

Exercise 2.1: Creating a CloudFront distribution

We already have a working static website. How can we make it perform better?

As that was last week, bring up the website in Chrome and remind yourself of what we have done.

We would now like to put a content delivery network (CloudFront) in-front of our static website host (S3) which we call the "origin".

Doing this will give our users around the world a lower latency, as Adam talked about.

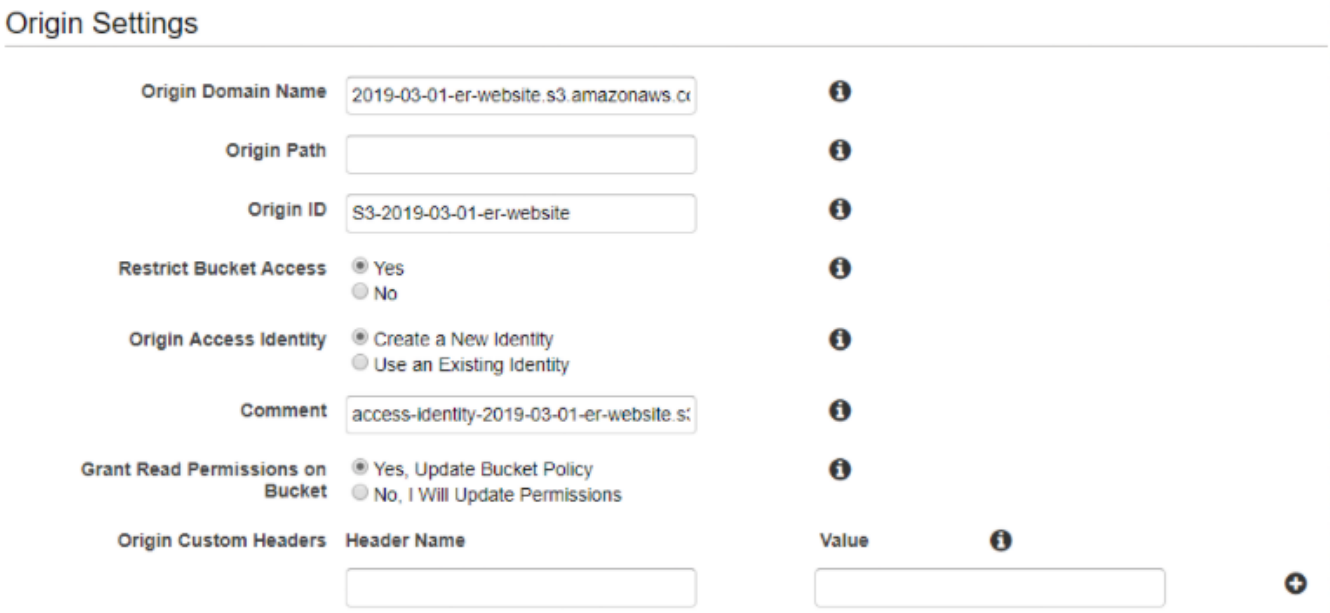
It will also alleviate the load that is placed on the origin bucket.

Once we configure the next few steps for setting up the content delivery network (Amazon CloudFront) it usually takes around 10 to 15 minutes to fully propagate. During that time we will go back to the S3 bucket Origin and prevent people from accessing content directly from S3. It will literally force the users to go via CloudFront, which will be much better for them.

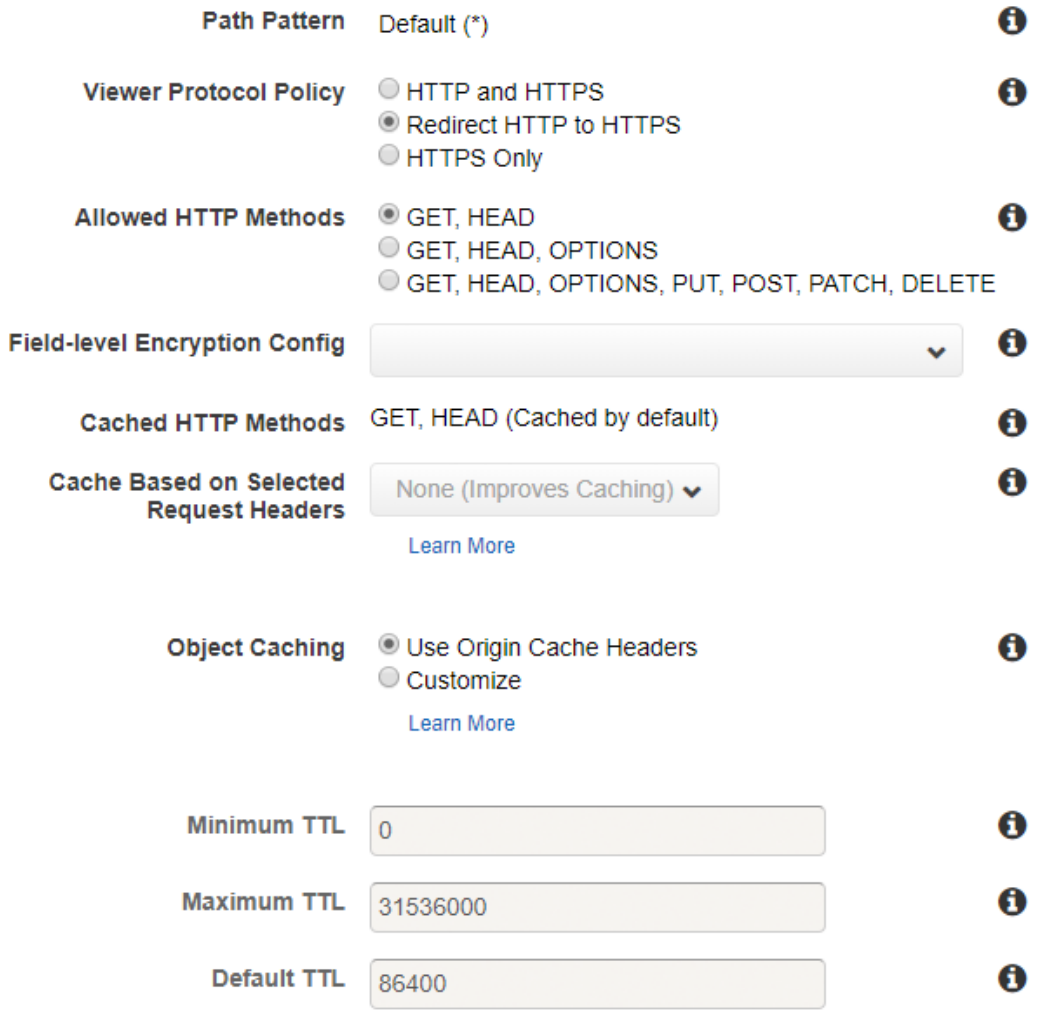
Let's start with configuring CloudFront for our website.

1. Steps for creating a CloudFront distribution

- Sign in to the AWS Management Console and in the **Find Services** search box type cloud and choose **CloudFront**.
- You should **Global** for the region at the top right.
- Click **Create Distribution**.
- Under **Web** click **Get Started**.
- For **Origin Domain Name** once you place the cursor in there you should see your available S3 buckets.
- Pick the website bucket you created.
- If it's not listed type it in: e.g `2019-03-01-er-website.s3.amazonaws.com` *Using your bucket name*
- Leave **Origin Path** blank.
- The **Origin ID** should have been pre-populated when you chose your bucket.
- Click **Yes** to **Restrict Bucket Access**.
- Under **Origin Access Identity** select **Create a New Identity**.
- It will pre-populate the **Comment** and append the bucket name.
- For **Grant Read Permissions on Bucket** check **Yes, Update Bucket Policy**. This will update the bucket policy for us.
- Leave the **Origin Custom Headers** blank.

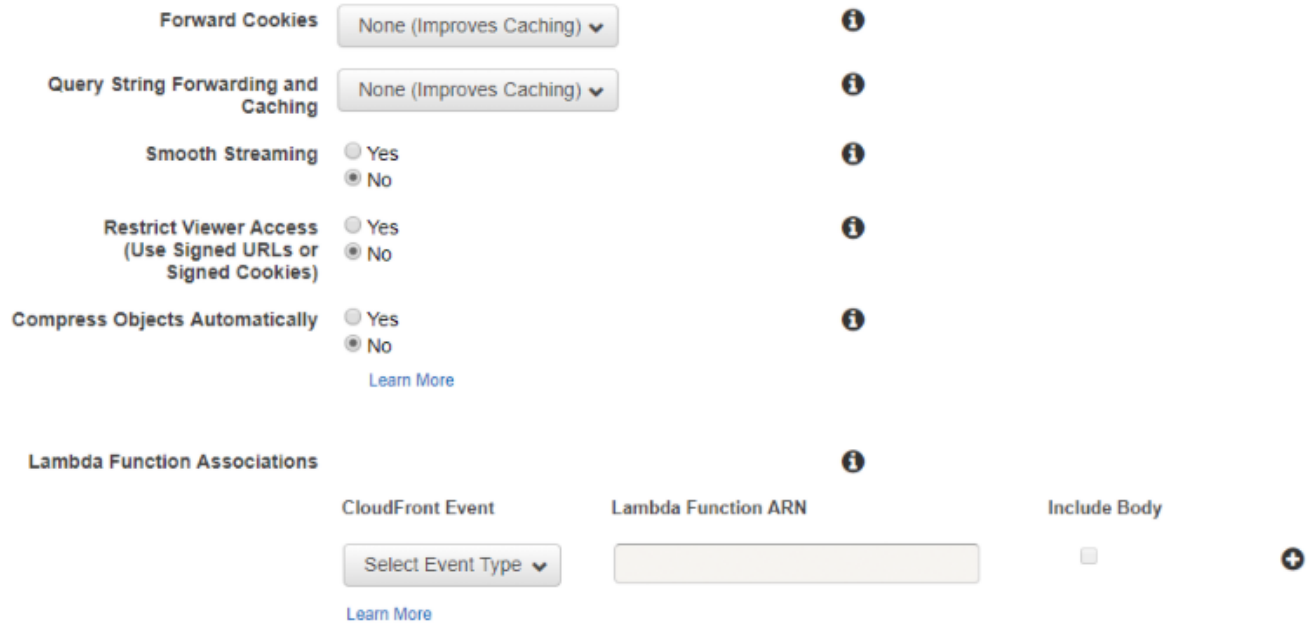


- For the **Default Cache Behavior Settings** section:
- Under **Viewer Protocol Policy** select **Redirect HTTP to HTTPS**.
- For **Allowed HTTP Methods** choose **GET, HEAD**.
- Leave **Field-level Encryption Config** blank.
- Leave **GET, HEAD (Cached by default)** for **Cached HTTP Methods**.
- For **Cache Based on Selected Request Headers** leave it as the default **None (Improves Caching)**.
- For **Object Caching** also leave it at the default **Use Origin Cache Headers**.

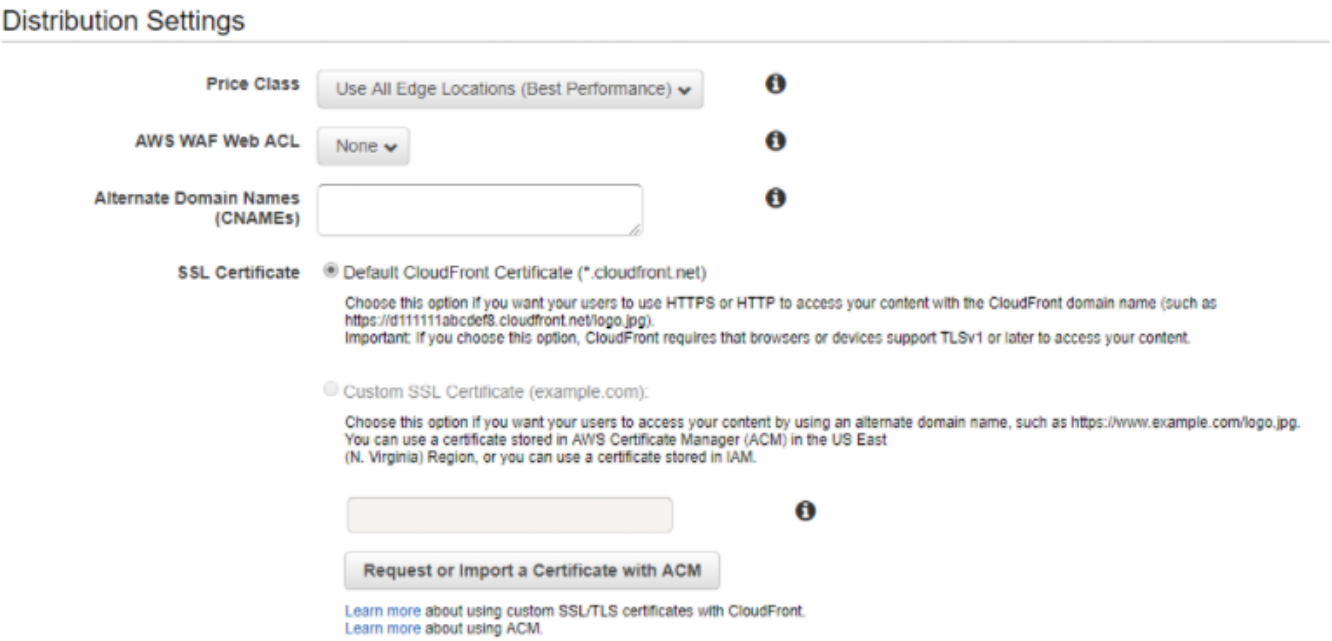


- Under **Forward Cookies** leave it as **None (Improves Caching)**.
- Also for **Query String Forwarding and Caching** leave as **None (Improves Caching)**.
- For **Smoothing Streaming** select **No**.
- For **Restrict Viewer Access (Use Signed URLs or Signed Cookies)** select **No**.
- Also leave **Compress Objects Automatically** as **No**.

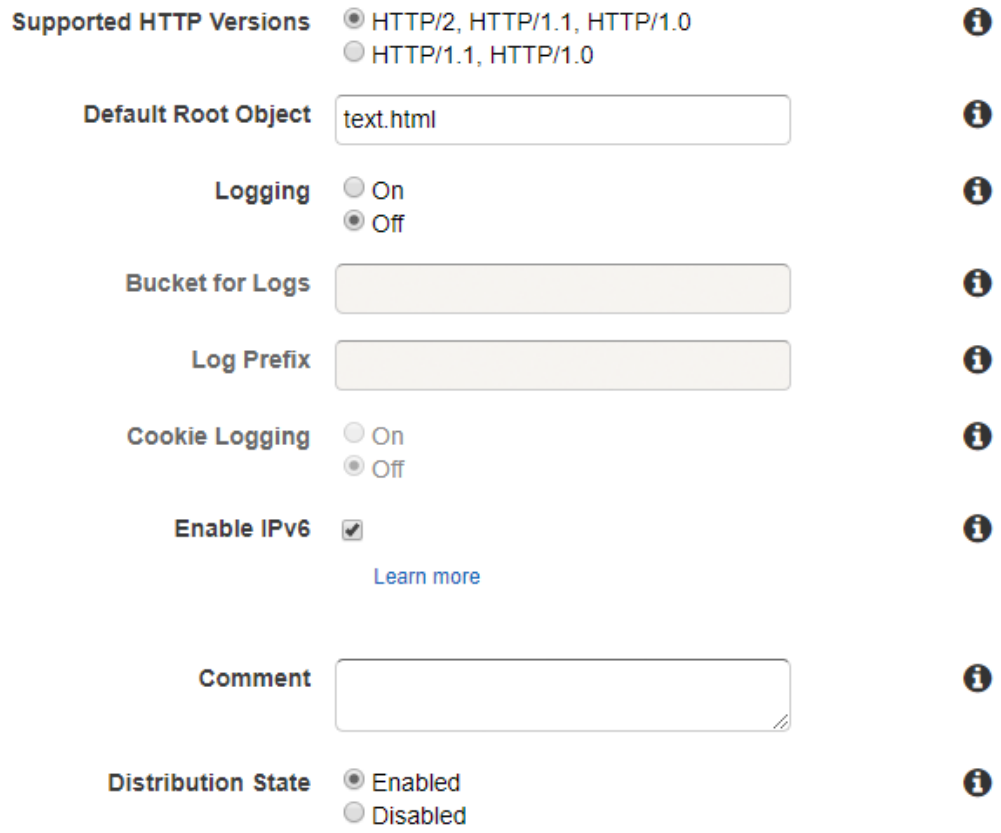
- We can also leave **Lambda Function Associations** as the default.



- Scroll down to **Distribution Settings**.
- For **Price Class** leave the default **Use All Edge Locations (Best Performance)**.
- We will not be using WAF so for **AWS WAF Web ACL** leave it as **None**.
- Also leave **Alternate Domain Names (CNAMEs)** blank.
- We will also use the **Default CloudFront Certificate** for **SSL Certificate**.



- For **Supported HTTP Versions** leave as **HTTP/2, HTTP/1.1, HTTP/1.0**.
- Under **Default Root Object** type in `text.html`.
- We can leave **Logging** set to **Off**.
- Leave **Enable IPv6** checked.
- Finally set **Distribution State** to **Enabled**.



- Click **Create Distribution**.
- Click on **Distributions** at the top left to see your CloudFront distribution being built.
- This can take 15-20 minutes to complete.

⏰ While we wait, we ill head over to S3 and lock down access to only allow calls from CloudFront.

2. Restrict our S3 bucket policy to CloudFront

- Click **Services** at the top left and type in S3 or select it from History.
- Click your bucket `2019-mm-dd-xx-website`. IMPORTANT: Your bucket will have a different name.
- Click **Permissions**.
- Select **Bucket Policy**.
- We can see that CloudFront has added what we call an "Origin Access Identity" to the policy.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AddPerm",
            "Effect": "Allow",
            "Principal": "*",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::2019-03-01-er-website/*"
        },
        {
            "Sid": "2",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::cloudfront:user/CloudFront Origin Access
                    Identity E1KO2GAPIWFF7X"
            },
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::2019-03-01-er-website/*"
        }
    ]
}
```

- Remove the public S3 access section so it looks more like the following:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "2",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::cloudfront:user/CloudFront Origin Access
                    Identity E1KO2GAPIWFF7X"
            },
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::2019-03-01-er-website/*"
        }
    ]
}
```

- This will only allow our specific CloudFront distribution access to our S3 bucket which is what we want.
- Click **Save** and grab a cup of coffee while we wait for the CloudFront Distribution to finish baking.

---

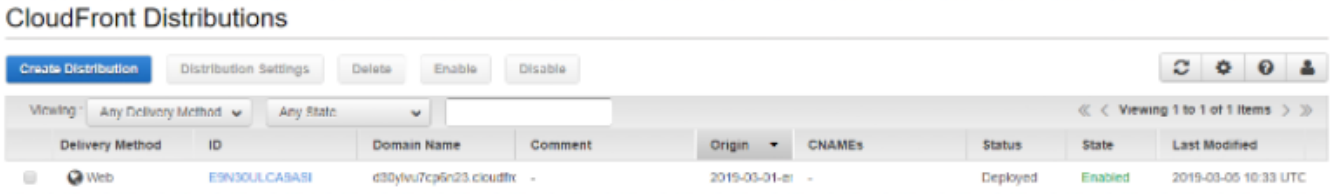
3. Steps for testing that we successfully locked down S3 from public view

- Browse to **your** S3 endpoint: *Example: http://2019-03-01-er-website.s3-website-us-east-1.amazonaws.com/*
- You will see a **403 Forbidden** as we effectively removed public access via the bucket policy.

403 Forbidden

- Code: AccessDenied
- Message: Access Denied

- Click on the CloudFront distribution ID. (The blue hyperlink)



- Copy the URL under **Domain Name**.
- Browse to that URL and you should now see the **text.html** page.

⚠️ Remeber the distribution may take up to 15 minutes to complete.

Next we will wire up our static website to a backend API.

Awesome, we are moving though our exercise goal list nicely.

Exercise goal checklist

1. ~~Create a simple chatbot using the lex console.~~
2. ~~Upload our website to S3.~~
3. ~~Create a content delivery network and lock down S3.~~
4. Build an API gateway mock with CORS.
5. Build a Lambda mock, use IAM, push logs to CloudWatch.
6. Create and seed a database with weather data.
7. Enhance the lambda, so it can query the database.
8. Play with your new text based data driven application.
9. Create a LEX proxy using Lambda.
10. Enhance API gateway to use the LEX proxy.
11. Play with your new voice web application.