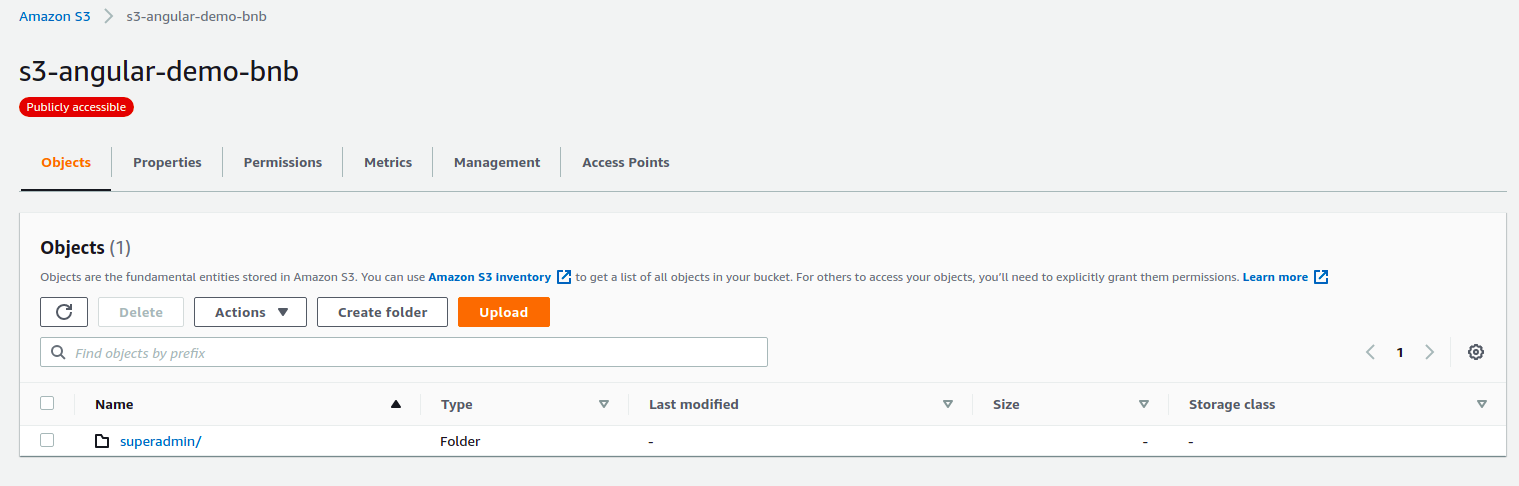
Deploying the angular front end using s3 bucket and the cloudfront.

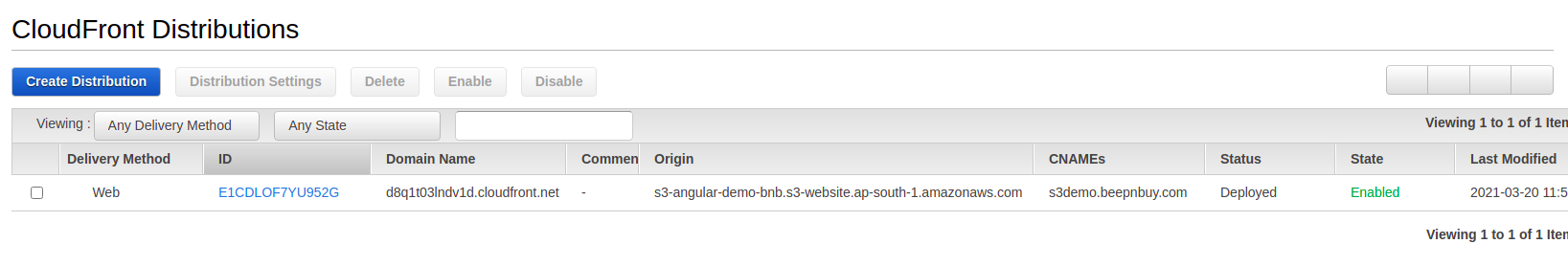
Here we have copied all the contents from the dist directory to s3 bucuckt at below path:



Also we have set to have static website option so that It could be served as front end.

Creating the cloud front distrinbution .

We have created the CDN distribution where for the origin we have given the s3 bucket url.

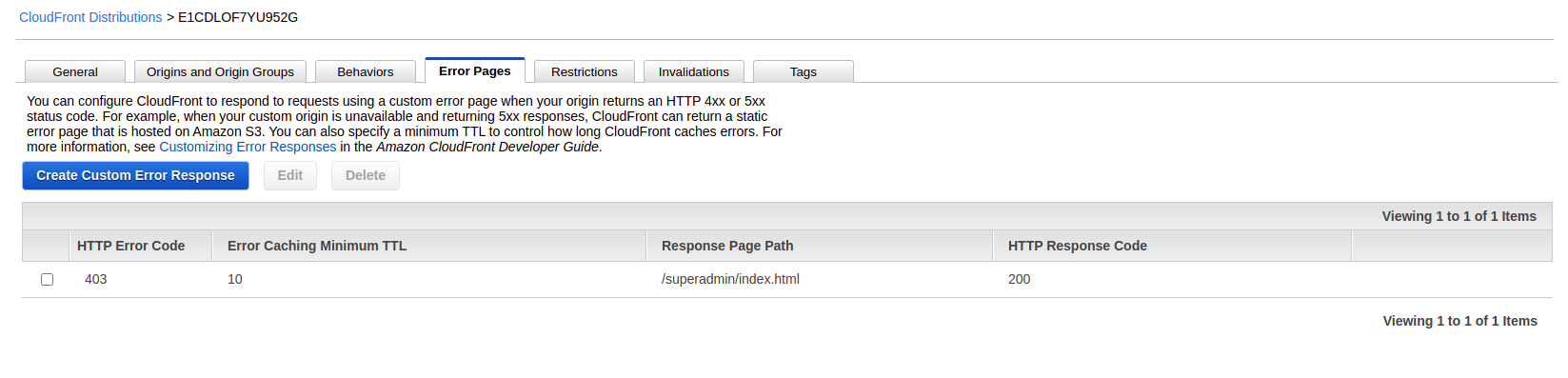
 Please note that to use CNAME in cloudfront we need to deploy the certificate in the

“us-east-1” region.

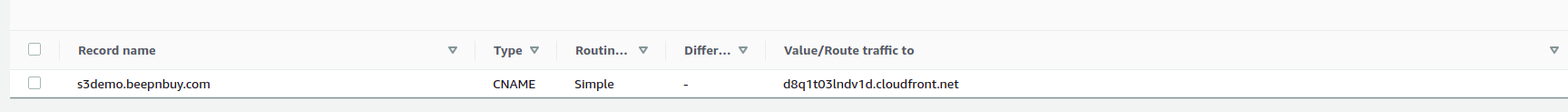
Also while creating the distribution we have given CNAME as “s3demo.beepnbuy.com”

**403 redirection:**

following 403 redirection we have configured according to current configuration.



Once the distributions status becomes deployed we need to add CNAME record in route53 for beepnbuy.com as below:



**Managing the headers for CORS:**

Following CORS policy has been set at the bucket level for CORS allow headers:

[

{

"AllowedHeaders": [

"\*"

],

"AllowedMethods": [

"GET",

"PUT",

"POST",

"DELETE"

],

"AllowedOrigins": [

"\*.beepnbuy.com"

],

"ExposeHeaders": []

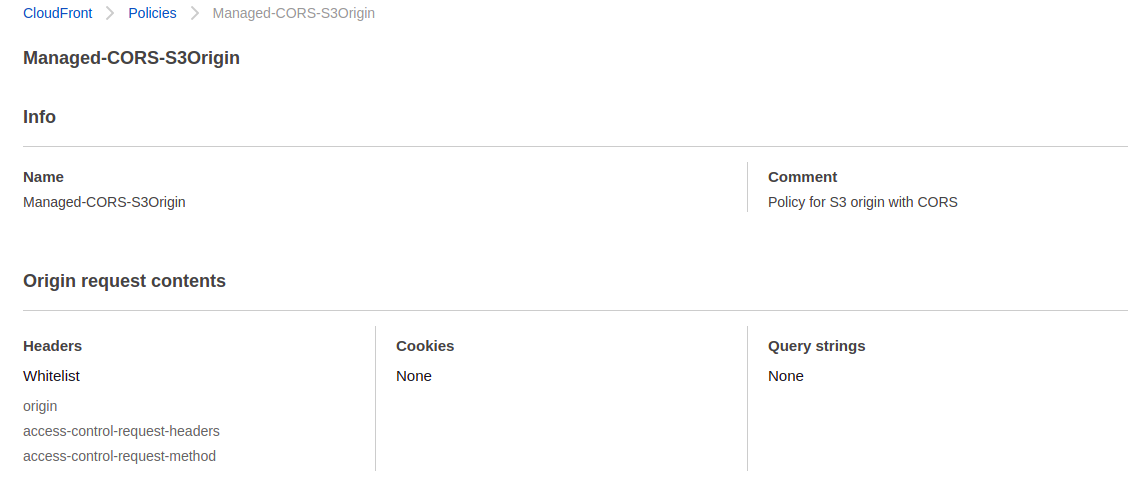
}

]

Also in the cloudfront following policy has been set for CORS:

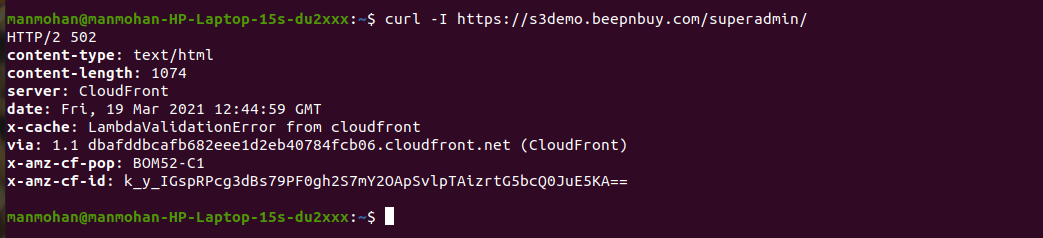


Details for the same policy is as below



**Deploying the lambda edge for setting up the headers.**

Now if you see the headers currently there are no headers have been configured



Now in order to to set header we are using lambda endge in the north virgina region.

Following nodsjs code we have used in lamda edge:

If you see in the above SS its whitelisting the headers for CORS.

'use strict';

exports.handler = (event, context, callback) => {

//Get contents of response

const response = event.Records[0].cf.response;

const headers = response.headers;

//Set new headers

headers['strict-transport-security'] = [{key: 'Strict-Transport-Security', value: 'max-age=31536000; includeSubdomains'}];

headers['x-content-type-options'] = [{key: 'X-Content-Type-Options', value: 'nosniff'}];

headers['x-frame-options'] = [{key: 'X-Frame-Options', value: 'SAMEORIGIN'}];

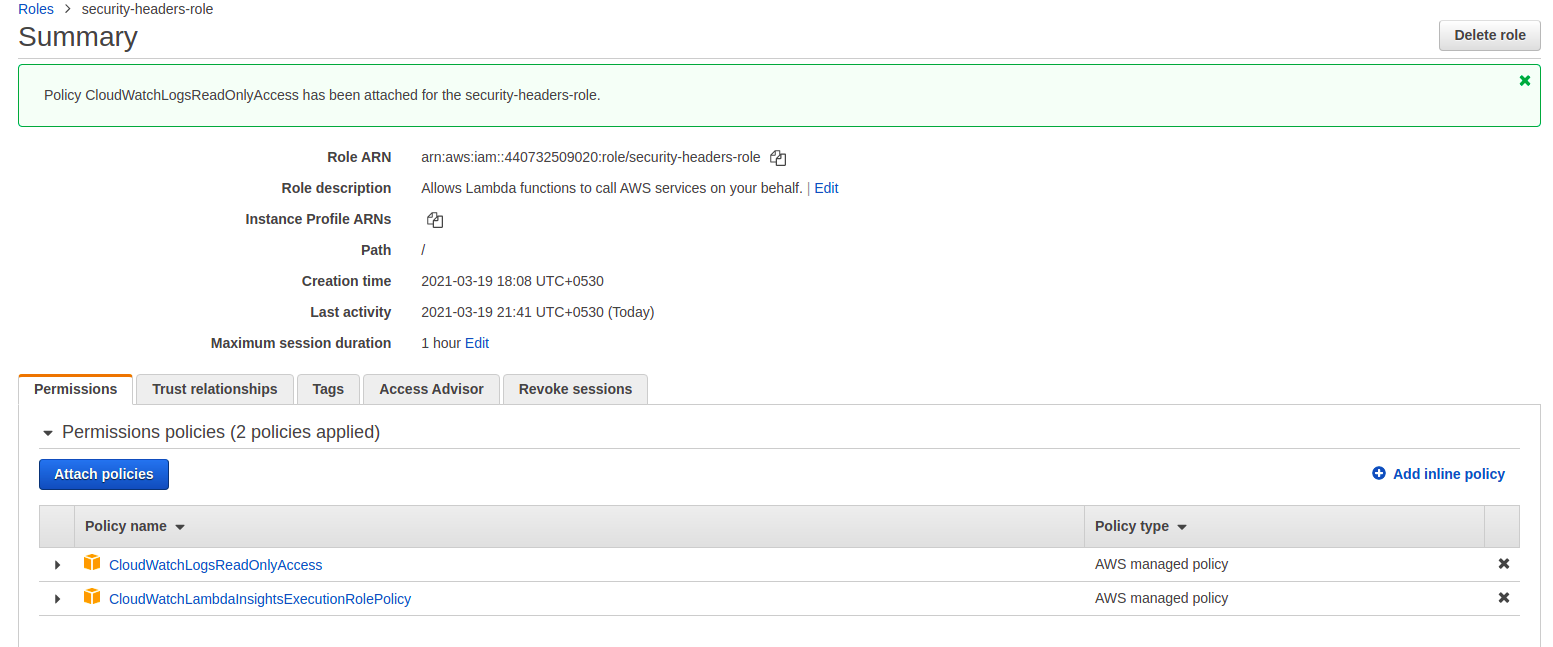
headers['x-xss-protection'] = [{key: 'X-XSS-Protection', value: '1; mode=block'}];

//Return modified response

callback(null, response);

};

Also to work this lambdaedge funtion following permissions are required for IAM user:



Also we need to configure below trust relationship in the IAM user using below code:

{

"Version": "2012-10-17",

"Statement": [

{

"Effect": "Allow",

"Principal": {

"Service": [

"lambda.amazonaws.com",

"edgelambda.amazonaws.com"

]

},

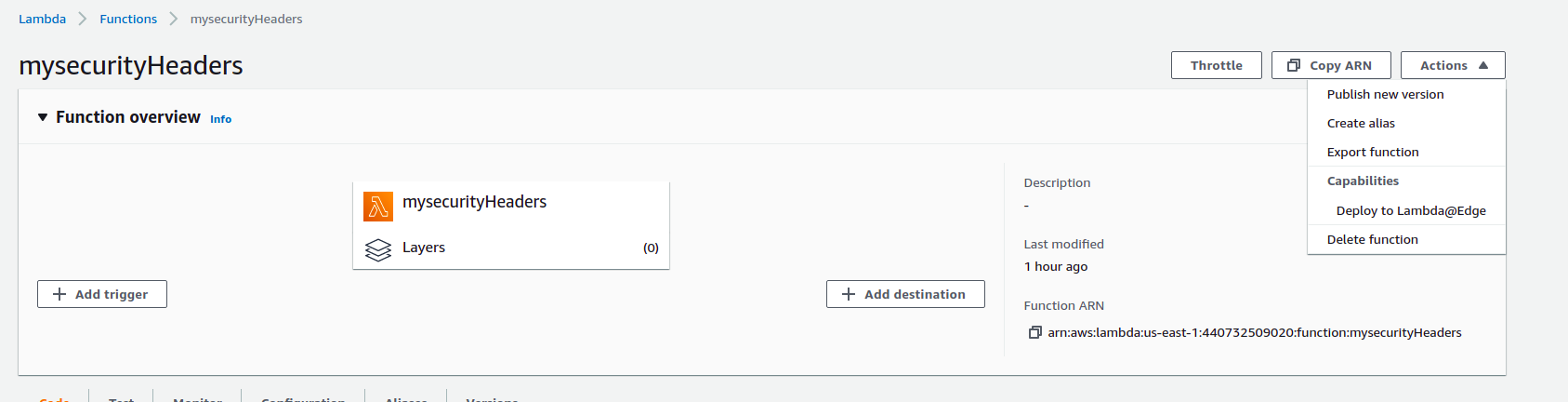
"Action": "sts:AssumeRole"

}

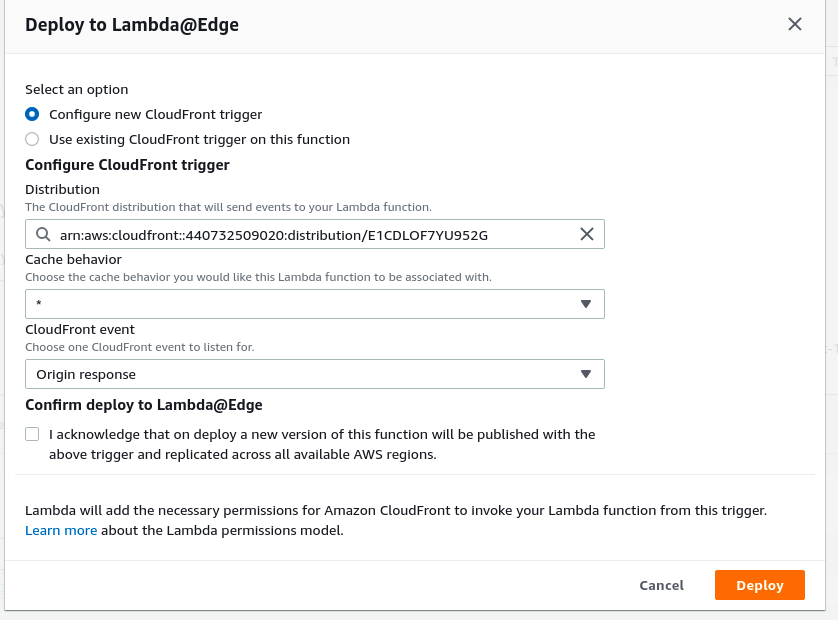
]

}

To deploy this code click on the actions and choose deploy to lambda edge option.



Following options we have selected while deploying this in the next popup.



After EdgeLambda execution now if you see the headers , following headers now have been set :



The same header setting we are using in the current production url as well.