

AWS: Static Website Host

A study into cloud solutions

Henry Lam

Objective

Demonstrate cloud administration expertise by hosting a static HTML website on Amazon Web Services (AWS).

Introduction

Hosting a static website provides an excellent foundation for showcasing cloud administration skills. This report details the process of deploying a static HTML website using AWS. By utilizing AWS tools like S3 Buckets, Route 53, and CloudFront, I successfully configured a secure, scalable, and functional website. This project highlights my ability to design and manage efficient cloud-based solutions.

S3 Buckets

To host the front-end components of the static website, I created an S3 bucket named `henryreallywantsajob.com`. This bucket houses the `index.html` file, which is available in my GitHub repository. The following configurations ensured the website's public accessibility:

- Enable static website hosting.
 - I activated static website hosting for the bucket, generating the endpoint:
`http://henryreallywantsajob.com.s3-website-us-east-1.amazonaws.com`

Static website hosting

Use this bucket to host a website or redirect requests. [Learn more](#)

We recommend using AWS Amplify Hosting for static website hosting

Deploy a fast, secure, and reliable website quickly with AWS Amplify Hosting. Learn more about [Amplify Hosting](#) or [View your existing Amplify apps](#)

Create Amplify app

S3 static website hosting

Enabled

Hosting type

Bucket hosting

Bucket website endpoint

When you configure your bucket as a static website, the website is available at the AWS Region-specific website endpoint of the bucket. [Learn more](#)
<http://henryreallywantsajob.com.s3-website-us-east-1.amazonaws.com>

- Allow public access.
 - I modified the bucket’s public access settings to enable access, ensuring the contents became reachable to users.

Block public access (bucket settings)

Edit

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to all your S3 buckets and objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to your buckets or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

Block all public access

Off

▼ Individual Block Public Access settings for this bucket

Block public access to buckets and objects granted through new access control lists (ACLs)

S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.

Block public access to buckets and objects granted through any access control lists (ACLs)

S3 will ignore all ACLs that grant public access to buckets and objects.

Block public access to buckets and objects granted through new public bucket or access point policies

S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.

Block public and cross-account access to buckets and objects through any public bucket or access point policies

S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

- Enable Access Control List (ACL)
 - By adjusting the ACL settings, I made individual objects (e.g., index.html) publicly accessible, resolving any access errors.

Access control list (ACL)

Edit

Grant basic read/write permissions to AWS accounts. [Learn more](#)

Grantee	Object	Object ACL
Object owner (your AWS account) Canonical ID: e4bfeb51e2d36c0330972bbce6c7d4b6fb1310070a826c73623d5e114693fd34	Read	Read, Write
Everyone (public access) Group: http://acs.amazonaws.com/groups/global/AllUsers	-	-
Authenticated users group (anyone with an AWS account) Group: http://acs.amazonaws.com/groups/global/AuthenticatedUsers	-	-

- Configuring the Bucket Policy
 - I implemented a bucket policy that granted public access to only the necessary objects, balancing accessibility with security.

Bucket policy

EditDelete

The bucket policy, written in JSON, provides access to the objects stored in the bucket. Bucket policies don't apply to objects owned by other accounts. [Learn more](#)

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PublicReadGetObject",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::henryreallywantsajob.com/*"
    }
  ]
}
```

Copy

Route 53

To link a custom domain to the S3 bucket, I utilized AWS Route 53 alongside CloudFront. Key steps included:

- Creating a public hosted zone.
 - I set up a public hosted zone for henryreallywantsajob.com, establishing a container for DNS records to manage domain routing.
- Allow public access
 - I created a CNAME record pointing henryreallywantsajob.com to the S3 bucket's static website endpoint. This made the domain accessible and routed traffic to the bucket efficiently.

Record details



Edit record


Record name

 _332563a62921d5c086d18b1f215d432b.henryreally

Record type

CNAME

Value

 _ad23e5f0e67aae9d2745e1bad29ca958.xlfgrmvvlj.ac
m-validations.aws.

Alias

No

TTL (seconds)

300

Routing policy

Simple

Cloudfront

To enhance security and performance, I configured AWS CloudFront as a content delivery network (CDN) with HTTPS support:

- Setting up a Cloudfront distribution

- I configured a CloudFront distribution with the S3 bucket as the origin, enabling secure and efficient content delivery.

Origin

Origin domain
Choose an AWS origin, or enter your origin's domain name. [Learn more](#)

Enter a valid DNS domain name, such as an S3 bucket, HTTP server, or VPC origin ID.

Protocol | [Info](#)

☒ HTTP only

☐ HTTPS only

☐ Match viewer

HTTP port
Enter your origin's HTTP port. The default is port 80.

- I enforced HTTPS for all connections to ensure secure communication by redirecting all HTTP connections to HTTPS.
- Requesting an SSL certificate.
 - Using AWS Certificate Manager (ACM), I obtained a public SSL certificate and verified domain ownership through Route 53 DNS validation records.
- Connecting the Certificate to CloudFront:
 - Once validated, I linked the SSL certificate to the CloudFront distribution, enabling secure HTTPS connections. I also added an alternate domain name (CNAME) for custom domain support.

Custom SSL certificate - *optional*

Associate a certificate from AWS Certificate Manager. The certificate must be in the US East (N. Virginia) Region (us-east-1).

www.henryreallywantsajob.com (3887d259-6fcb-45ff-a610-ed2be934b499) ▼



✓ www.henryreallywantsajob.com [Request certificate](#)

Legacy clients support - \$600/month prorated charge applies. Most customers do not need this.

CloudFront allocates dedicated IP addresses at each CloudFront edge location to serve your content over HTTPS.

☐ Enabled

Security policy

The security policy determines the SSL or TLS protocol and the specific ciphers that CloudFront uses for HTTPS connections with viewers (clients).

☒ TLSv1.2_2021 (recommended)

☐ TLSv1.2_2019

☐ TLSv1.2_2018


☐ TLSv1.1_2016

☐ TLSv1_2016

☐ TLSv1

- Updating Route 53 records
 - I updated the Route 53 hosted zone to route traffic to the CloudFront distribution. A new CNAME record was added to direct www.henryreallywantsajob.com requests to the CloudFront endpoint.


Record name

 www.henryreallywantsajob.com

Record type

CNAME

Value

 d3ap9l1bbe5iy1.cloudfront.net

Alias

No

TTL (seconds)

300

Routing policy

Simple

- Firewall Considerations
 - Given the static nature of the website, I chose not to enable a web application firewall (WAF), as the site lacks dynamic content or application logic.

Web Application Firewall (WAF) [Info](#)



Enable security protections

Keep your application secure from the most common web threats and security vulnerabilities using AWS WAF. Blocked requests are stopped before they reach your web servers.



Do not enable security protections

Select this option if your application does not need security protections from AWS WAF.

Result

The final outcome is a secure, scalable static website hosted on AWS with the custom domain [henryreallywantsajob.com](https://www.henryreallywantsajob.com). Accessible via HTTPS, the website guarantees trust and reliability for users while leveraging cost-effective and efficient cloud solutions.

