Using AWS to host secure, static website

This might be just one of the articles in already existing plethora of similar articles, teaching how to host a static website using any cloud provider services. However, I will try to make it different, by using simpler language, and simple examples. If this makes sense, please continue reading…

## Table of contents

* [Why host a static website?](#_Why_host_a)
* [Pre-requisites](#_Pre-requisites)
* [Registering a domain with Route53](#_Registering_domain_with)
* [Generating SSL certificate for domain](#_Generating_SSL_certificate)
* [Adding the static website assets to S3](#_Adding_the_static)
* [Configuring CloudFront for S3](#_Configuring_CloudFront_for)
* [The last action](#_The_last_action)
* [Target achieved](#_Target_achieved)

# Why host a static website?

Even before we begin, the first question that rings is, in the modern era of Web 3.0, why would somebody want a static site? It’s just plain useless, isn’t it? Well may be or maybe not. Not everyone in the world is super techie but having a website might be pretty cool for someone who just wants to share their wonderful trip photographs with friends, family, neighbors, colleagues. How about having a simple blog? What about small retailers who just wants to showcase their products in order to attract the customer to the brick-and-mortar shop? And there can be numerous other use cases where a simple static site can make sense. If you still agree with me, please continue reading.

So, why use so much of technicalities to host just a simple static site? Well, I would say the initial setup is only a one-time job and further will require little to no maintenance. Since we are focusing on AWS, here is the list of service that we will be utilizing to fulfil our purpose

* [Simple Storage Service (S3)](https://aws.amazon.com/s3/)
* [CloudFront](https://aws.amazon.com/cloudfront/)
* [Certificate Manager](https://aws.amazon.com/certificate-manager/)
* [Identity And Access Management](https://aws.amazon.com/iam/) (implicitly)
* [Route 53](https://aws.amazon.com/route53/)

The diagram below coarsely represents the components above and how the relate to each other.

A picture containing graphical user interface

Description automatically generated

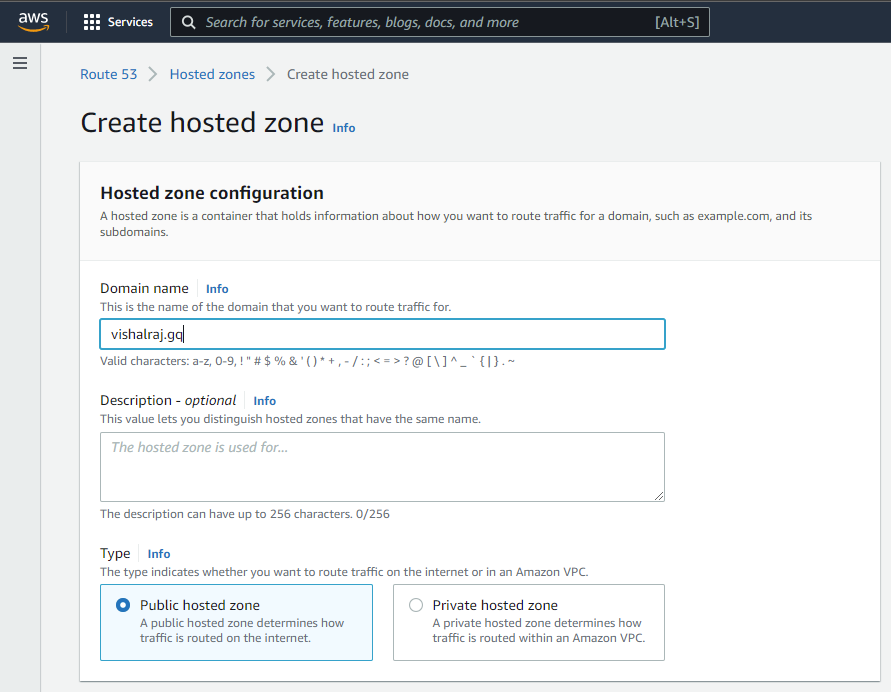
# Pre-requisites

Before we proceed ahead, we have two pre-requisites.

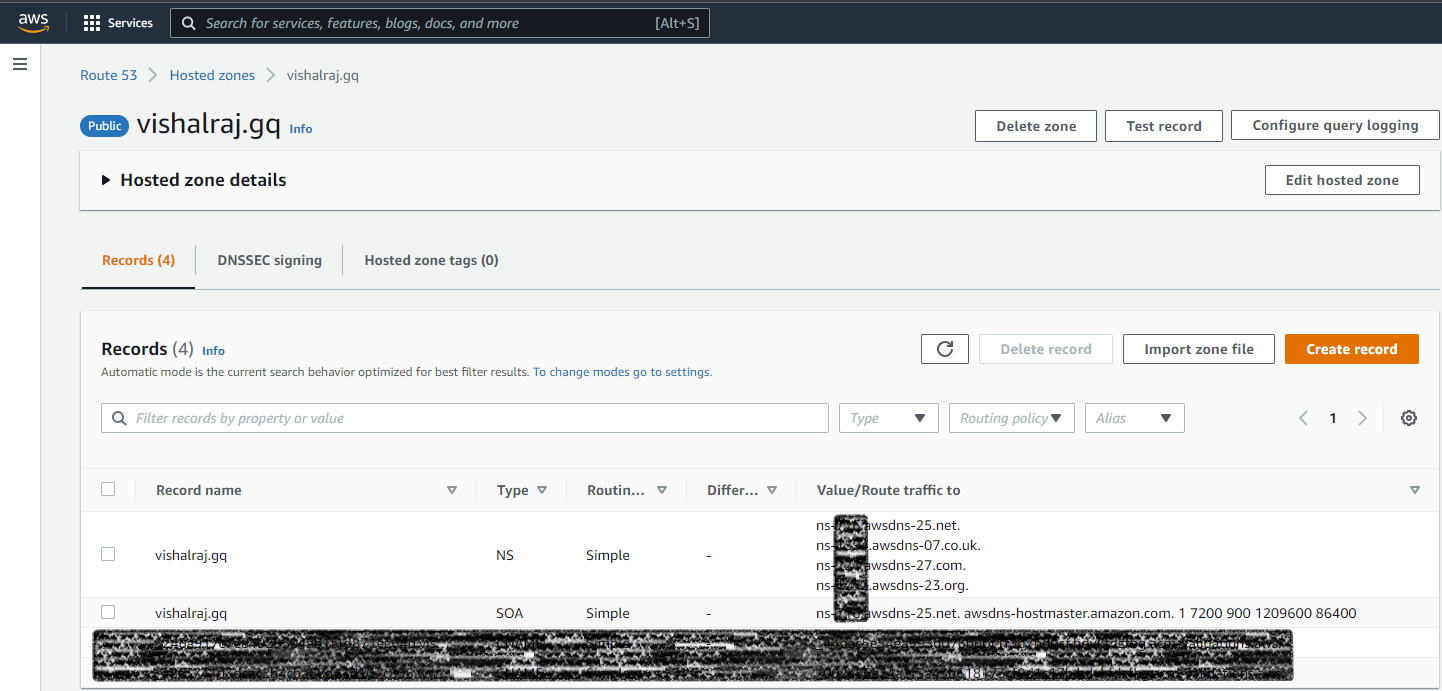
1. A registered domain name. If you just want to experiment, [freenom.com](https://freenom.com/) provides free domain name for up to 12 months, but only with certain TLD.
2. An AWS account. A fresh account is eligible for some free resources. See more [here](https://aws.amazon.com/free/).

# Registering domain with Route53

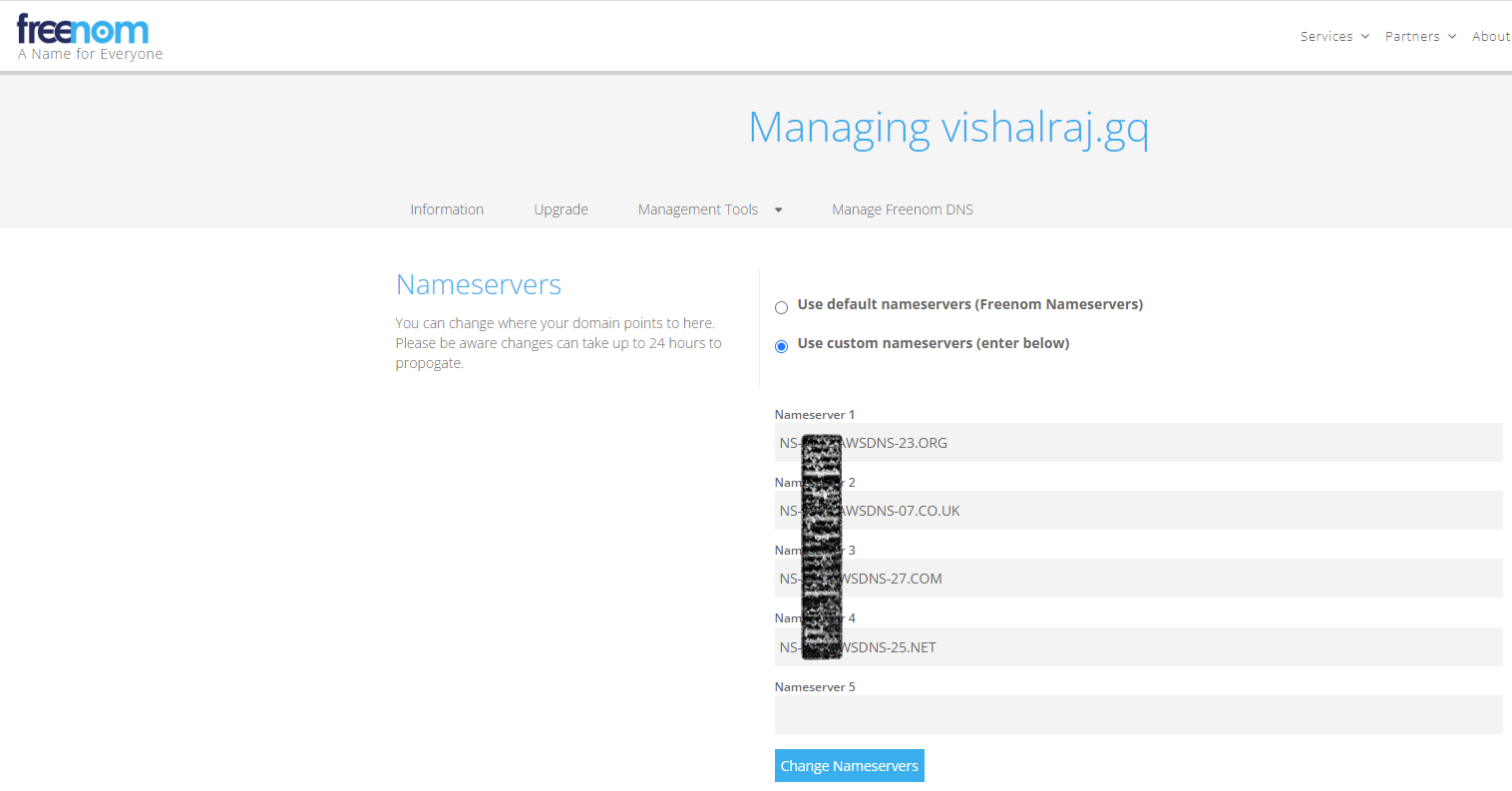
AWS Route53 is highly available and scalable cloud-based DNS system with 100% uptime guarantee. It provides an array of services for DNS management. We shall begin by adding our domain in Route53 by creating a new public hosted zone.



Once we register our domain with Route53, it will provide us with the list of name servers to be used for our domain.



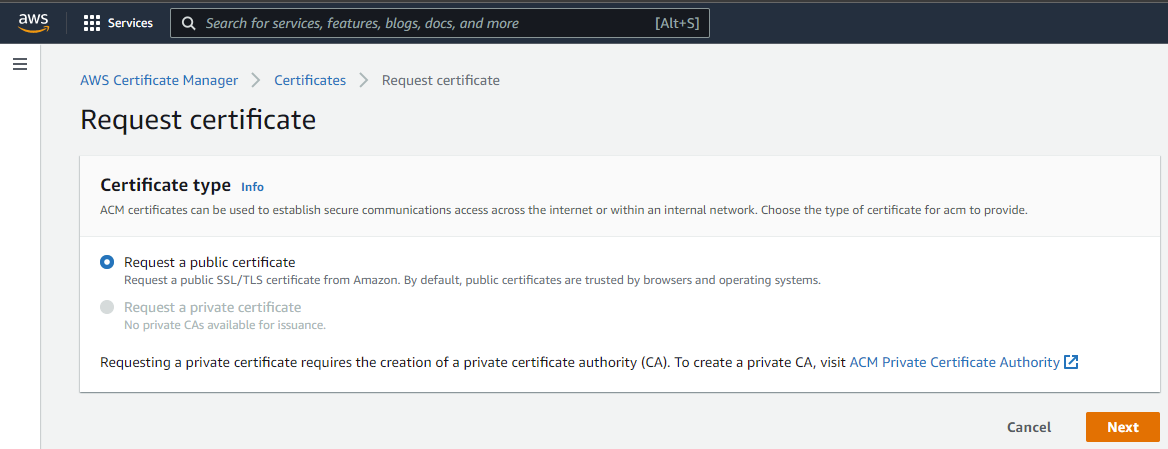
These name servers are managed by AWS, and we need to update this with our domain name registrar.



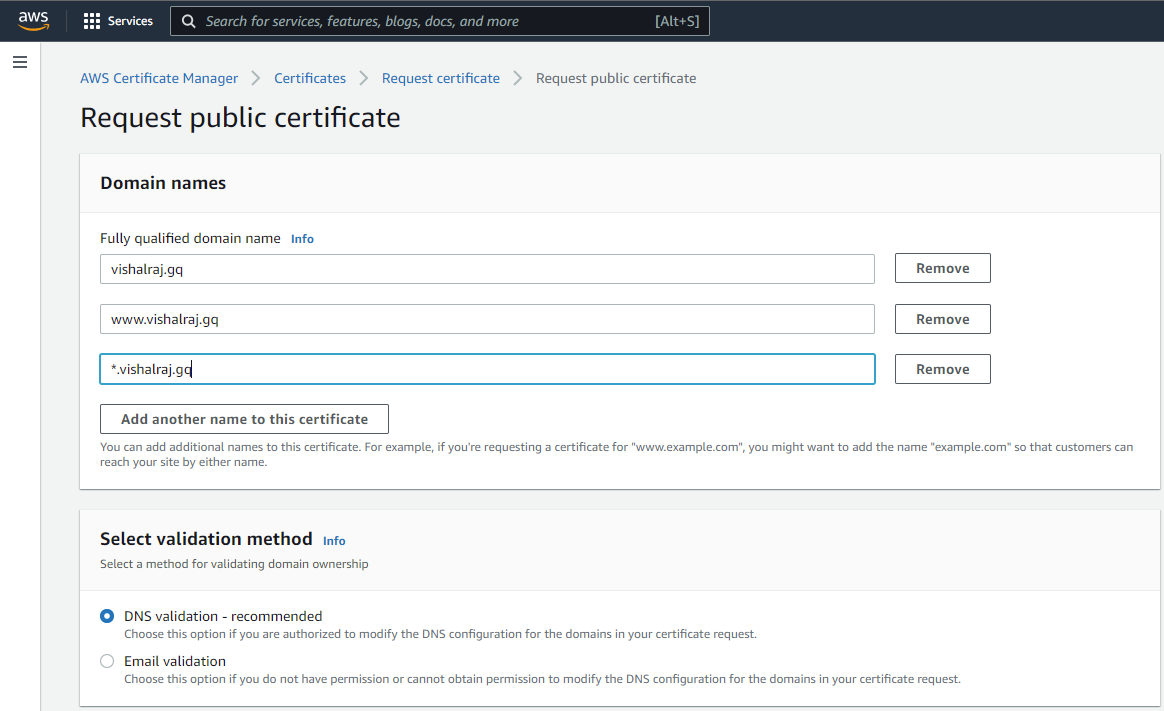
# 

# Generating SSL certificate for domain

The AWS Certificate Manager provides public SSL certificates for free of cost. We shall use it to generate the certificates for our domain, so that our domain can be access using the HTTPS protocol. This ensures that all communication between the user and the server is encrypted, and safe.



We will use the fully qualified domain name (FQDN), the www sub-domain and a \* for all future sub-domains, that we may need.



After we request the SSL certificate, AWS ensures that we are the actual owners of the domain before issuing the certificates. AWS recommends that we add certain CNAME records against our domain, which it can verify. Once the records are added, it takes some time for them to be propagated to AWS.

Graphical user interface, text, email, website

Description automatically generated

Once AWS verifies the domain with the CNAME entries, the certificate is issued and ready to be used.

Graphical user interface, application

Description automatically generated

# Adding the static website assets to S3

AWS Simple Storage Service seems to be the ideal choice when it comes to storing data. S3 supports practically unlimited storage. Each object in S3 can span up to 5 TB in size. AWS charges us for using S3 on the following two parameters

* Space consumed
* Size of data moving in and out of S3

Once we have our static site assets such as HTML, CSS, JavaScript, Images, Videos, PDFs etc. (whatever is required), we need to move it toS3. But before that we must create a bucket in S3, with a globally unique name. Because the bucket name is a part of URL which must globally unique.

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

# Configuring CloudFront for S3

AWS CloudFront is a globally distributes CDN service. Using CloudFront we can ensure that users from across the globe are able to access our contents with low latency. CloudFront can be integrated with several other AWS services include S3. Let’s go ahead and create a CloudFront distribution for our S3 bucket.

Graphical user interface, text, application

Description automatically generated

Although we could have enabled public access for S3 bucket, but rather we kept it private (by default). With OAI, we are permitting the CloudFront distribution to be able to access contents via the bucket policy.

A screenshot of a computer

Description automatically generated with medium confidence

Although AWS CloudFront has edge locations across the world, we can segregate the content distribution based on where are customer are located. Additionally, we are also informing the CloudFront distribution about the domain via which it will be accessed and providing the corresponding SSL certificates. Once the distribution is created, it will take some time before it is ready to be consumed.

A screenshot of a computer

Description automatically generated

# The last action

Once our CloudFront distribution is ready to be used, we must make another A record entry against the domain which would be a proxy to the CloudFront distribution.

Graphical user interface, text, application, email

Description automatically generated

# Target achieved

After all this, we need to be patient for all the DNS information to be propagated. Once it is done, we can access our domain and see the static site in action.

A snowy road with trees on either side of it

Description automatically generated with medium confidence