Hosting a Website with Amazon S3, CloudFront, and Custom Domain

This guide provides step-by-step instructions for hosting a static website using Amazon S3, CloudFront, and a custom domain with SSL/TLS certificate from AWS Certificate Manager (ACM).

Table of Contents

S3 Static Website Hosting Basics

What is S3 Static Website Hosting?

Amazon S3 provides a simple and cost-effective way to host static websites. A static website uses HTML, CSS, JavaScript, images, and other client-side files that don't require server-side processing.

Key Features of S3 Website Hosting

- **Highly Available**: Built on Amazon's reliable infrastructure
- **Scalable**: Automatically handles traffic spikes
- **Cost-Effective**: Pay only for storage used and data transferred
- **Secure**: Multiple security options including bucket policies and access control
- **Simple**: No servers to manage or maintain

Limitations of S3 Website Hosting

- Only supports static content (no server-side processing)
- Only supports HTTP by default (HTTPS requires CloudFront)
- S3 website endpoint URLs are not customizable
- Limited caching capabilities

CloudFront Basics

What is Amazon CloudFront?

Amazon CloudFront is a fast content delivery network (CDN) service that securely delivers data, videos, applications, and APIs to customers globally with low latency and high transfer speeds.

Key Features of CloudFront

- **Global Edge Network**: Content is cached at edge locations worldwide
- **Low Latency**: Delivers content from the nearest edge location
- **HTTPS Support**: Provides secure connections with SSL/TLS

- **Integration**: Works seamlessly with other AWS services
- **Customizable**: Various settings for caching behavior, origin selection, etc.

CloudFront Terminology

- **Distribution**: The main CloudFront configuration unit
- **Origin**: The source of content (e.g., S3 bucket, HTTP server)
- **Cache Behavior**: Rules that determine how content is cached and served
- **TTL (Time to Live)**: How long content remains in the cache
- **Invalidation**: Process to remove content from the cache before it expires
- **Edge Location**: Data center where content is cached

Types of CloudFront Distributions

1. **Web Distribution**: For static and dynamic web content
2. **RTMP Distribution**: For streaming media files (legacy, being deprecated)

Advanced CloudFront Concepts

Origin Access Identity (OAI)

- Secure way to allow CloudFront to access S3 content
- Prevents direct access to S3 objects
- Requires specific bucket policy configuration

Origin Access Control (OAC)

- Modern replacement for OAI with enhanced security
- Supports additional features and better integration with AWS services

Cache Behaviors

- Path pattern-based rules for different caching strategies
- Control over:
    - TTL settings
    - Compression
    - HTTP methods allowed
    - Query string forwarding
    - Cookie forwarding
    - Header forwarding

Edge Functions

1. **CloudFront Functions**: Lightweight JavaScript functions for:
    - URL rewrites/redirects
    - HTTP header manipulation
    - Simple request authentication
2. **Lambda@Edge**: More powerful functions for:
    - Content generation
    - A/B testing
    - Complex authentication
    - Server-side rendering

Security Features

- **Field-level encryption**: Protect sensitive data throughout the system
- **WAF integration**: Web Application Firewall for protection against common exploits
- **Geo-restriction**: Block access from specific geographic locations
- **Signed URLs/Cookies**: Provide time-limited access to content

Performance Optimization

- **Origin Shield**: Additional caching layer to reduce load on origins
- **Real-time logs**: Monitor and analyze viewer behavior
- **Origin failover**: Automatic switching to backup origins
- **Compression**: Automatic GZIP/Brotli compression

Custom Domain Configuration with ACM

AWS Certificate Manager (ACM)

- Provides free SSL/TLS certificates for AWS services
- Manages certificate renewal automatically
- Integrates seamlessly with CloudFront and other AWS services

DNS Configuration Options

1. **Amazon Route 53**: AWS's DNS service
    - Easiest integration with other AWS services
    - Health checks and routing policies
    - Automatic record creation with ACM
2. **Third-party DNS providers**:
    - Requires manual CNAME record creation
    - May need additional verification steps for ACM

Domain Validation Methods

1. **DNS validation**: Add CNAME records to prove domain ownership
2. **Email validation**: Receive and respond to verification emails

Step-by-Step Implementation Guide

Step 1: Prepare Your Website Files

1. Organize your HTML, CSS, JavaScript, images, and other assets
2. Ensure you have an `index.html` file as your main entry point
3. Test your website locally to ensure everything works correctly

Step 2: Create and Configure an S3 Bucket

1. Sign in to the AWS Management Console
2. Navigate to the S3 service
3. Click "Create bucket"
4. Choose a globally unique bucket name (doesn't need to match your domain)

5. Select the AWS Region closest to your primary audience
6. Configure bucket settings:
    - Block all public access (we'll use CloudFront for access)
    - Enable bucket versioning (optional but recommended)
    - Enable default encryption (recommended)
7. Create the bucket
8. Upload your website files to the bucket
9. Note: Do NOT enable static website hosting in bucket properties (we'll use CloudFront instead)

Step 3: Request an SSL/TLS Certificate

1. Navigate to AWS Certificate Manager (ACM)
2. Ensure you're in the US East (N. Virginia) region, as CloudFront requires certificates from this region
3. Click "Request a certificate"
4. Select "Request a public certificate"
5. Enter your domain names:
    - Main domain: example.com
    - Include subdomain: www.example.com (http://www.example.com) (optional)
6. Choose a validation method (DNS validation recommended)
7. Follow the validation steps provided
8. Wait for the certificate to be issued (status: "Issued")

Step 4: Create a CloudFront Distribution

1. Navigate to CloudFront in the AWS Management Console
2. Click "Create Distribution"
3. Configure origin settings:
    - Origin domain: Select your S3 bucket
    - Origin access: Select "Origin access control settings (recommended)"
    - Create a new OAC and apply it
4. Configure default cache behavior:
    - Viewer protocol policy: "Redirect HTTP to HTTPS"
    - Allowed HTTP methods: "GET, HEAD" (for static websites)
    - Cache key and origin requests: Use recommended settings for static websites
5. Configure distribution settings:
    - Alternate domain names (CNAMEs): Enter your domain names (e.g., example.com, www.example.com (http://www.example.com))
    - SSL certificate: Select "Custom SSL Certificate" and choose your ACM certificate
    - Default root object: "index.html"
6. Create the distribution
7. Wait for the distribution to deploy (Status: "Deployed")

Step 5: Update S3 Bucket Policy

1. After creating the CloudFront distribution, go back to your S3 bucket
2. Select the "Permissions" tab
3. Edit the bucket policy to allow access from your CloudFront distribution
4. The console should suggest the appropriate policy, which will look similar to:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowCloudFrontServicePrincipal",
            "Effect": "Allow",
            "Principal": {
                "Service": "cloudfront.amazonaws.com"
            },
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::your-bucket-name/*",
            "Condition": {
                "StringEquals": {
                    "AWS:SourceArn": "arn:aws:cloudfront::your-account-id:distribution/your-distribution-id"
                }
            }
        }
    ]
}
```

Step 6: Configure DNS Records

If using Route 53:

1. Navigate to Route 53 in the AWS Management Console
2. Select your hosted zone
3. Click "Create record"
4. Create an A record:
    - Name: @ (for root domain) or www (for subdomain)
    - Record type: A - IPv4 address
    - Alias: Yes
    - Route traffic to: "Alias to CloudFront distribution"
    - Select your CloudFront distribution
5. Create the record
6. Repeat for additional subdomains if needed

If using a third-party DNS provider:

1. Log in to your DNS provider's management console
2. Create a CNAME record:
    - Name: @ or www (depending on provider's syntax)
    - Value: Your CloudFront distribution domain name (e.g., d1234abcdef.cloudfront.net)
    - TTL: 3600 (or as recommended by your provider)
3. Save the record
4. Note: Some DNS providers don't support CNAME at the root domain. In this case, check if they offer ANAME, ALIAS, or similar record types

Step 7: Test Your Website

1. Wait for DNS propagation (can take up to 48 hours, but often much faster)
2. Open a web browser and navigate to your domain
3. Verify that:
    - The website loads correctly
    - HTTPS is working (lock icon in browser)
    - All links and resources load properly

Step 8: Optional Optimizations

1. **Configure error pages**:
    - In CloudFront, edit your distribution
    - Under "Error Pages", add custom error responses
    - Common setup: redirect 404 errors to a custom error page
2. **Set up redirects**:
    - Redirect www to non-www (or vice versa)
    - Implement using Lambda@Edge or S3 website redirect rules
3. **Enable CloudFront logs**:
    - Create an S3 bucket for logs
    - In your CloudFront distribution settings, enable logging
    - Specify the log bucket and prefix
4. **Configure cache invalidation**:
    - After updating website content, create invalidations in CloudFront
    - Use the CloudFront console or AWS CLI
    - Example CLI command: `aws cloudfront create-invalidation --distribution-id YOUR_DISTRIBUTION_ID --paths "/*"`

Troubleshooting Common Issues

Website Not Loading

- Verify DNS records are correctly configured
- Check that the CloudFront distribution is deployed
- Ensure S3 bucket policy allows CloudFront access
- Confirm index.html exists at the root of your bucket

SSL/TLS Certificate Issues

- Ensure certificate is fully validated and in "Issued" state
- Verify certificate is in the US East (N. Virginia) region
- Check that all domain names in use are covered by the certificate

Content Not Updating

- CloudFront caches content based on TTL settings
- Create an invalidation to force refresh of cached content
- Check that you're updating the correct S3 bucket

Access Denied Errors

- Verify the S3 bucket policy is correctly configured
- Check that CloudFront OAC/OAI is properly set up
- Ensure the files in S3 have appropriate permissions

Best Practices

1. **Security**:
    - Always use HTTPS (redirect HTTP to HTTPS)
    - Implement appropriate bucket policies
    - Consider using AWS WAF with CloudFront for additional protection
    - Regularly rotate any access credentials
2. **Performance**:
    - Optimize images and other assets
    - Use appropriate cache TTL values
    - Consider implementing compression
    - Minimize the use of third-party scripts
3. **Cost Optimization**:
    - Monitor CloudFront and S3 usage
    - Use the appropriate price class for CloudFront
    - Consider implementing S3 lifecycle policies for logs and backups
4. **Maintenance**:
    - Implement a CI/CD pipeline for website updates
    - Set up monitoring and alerts
    - Document your configuration for future reference
    - Regularly test your website's performance and security

By following this guide, you'll have a secure, scalable, and high-performance static website hosted on AWS infrastructure with your custom domain name and HTTPS support.