**Prerequisites**

* A remote server with a fresh installation of Ubuntu (for Debian-based systems) or CentOS (for RedHat-based systems).
* Ansible installed on your local machine.
* SSH access to the remote server using a key pair.

**Step 1: Define Your Hosts**

In the hosts file, specify the IP address or hostname of your remote server. You should also provide the SSH user and the private key file for authentication.

**[remote-mc]**

**18.216.20.248 ansible\_ssh\_user=ubuntu ansible\_ssh\_private\_key\_file=~/.ssh/k8s-poc-demo.pem**

## Step 2: Create Apache Configuration Templates

Create two Apache configuration templates, one for HTTPS (default-ssl.conf.j2) and one for HTTP to HTTPS redirection (default.conf.j2). These templates define how Apache should handle SSL and redirection.

### default-ssl.conf.j2 (For HTTPS Configuration)

**<IfModule mod\_ssl.c>**

**<VirtualHost \_default\_:443>**

**ServerAdmin webmaster@localhost**

**DocumentRoot /var/www/html**

**SSLEngine on**

**SSLCertificateFile /etc/ssl/certs/apache-selfsigned.crt**

**SSLCertificateKeyFile /etc/ssl/private/apache-selfsigned.key**

**ErrorLog ${APACHE\_LOG\_DIR}/error.log**

**CustomLog ${APACHE\_LOG\_DIR}/access.log combined**

**</VirtualHost>**

**</IfModule>**

### **default.conf.j2 (For HTTP to HTTPS Redirection)**

<VirtualHost \*:80>

ServerAdmin webmaster@localhost

DocumentRoot /var/www/html

Redirect permanent / https://your\_server\_domain/

</VirtualHost>

Replace your\_server\_domain with your server's domain name or IP address.

## Step 3: Create an Ansible Playbook

Create an Ansible playbook (e.g., install.yml) to automate the server setup and configuration. This playbook includes tasks to update the package cache, install Apache and OpenSSL, create SSL certificates, configure Apache for HTTPS, and set up HTTP to HTTPS redirection.

---

- name: Create and Deploy Web Server

hosts: remote-mc

gather\_facts: true

become: yes

tasks:

# ... (tasks from your playbook) ...

- name: Check Apache Configuration Syntax

command: apachectl configtest

ignore\_errors: yes

register: apache\_configtest\_result

- name: Restart Apache (if config test is successful)

systemd:

name: apache2

state: restarted

when: apache\_configtest\_result.rc == 0

handlers:

- name: restart apache

service:

name: "{{ 'apache2' if ansible\_os\_family == 'Debian' else 'httpd' }}"

state: restarted

The install.yml Ansible playbook automates the server setup and configuration. It includes tasks to:

* Update the package cache
* Install Apache and OpenSSL
* Create SSL certificates
* Configure Apache for HTTPS
* Set up HTTP to HTTPS redirection.

## Step 4: Run the Ansible Playbook

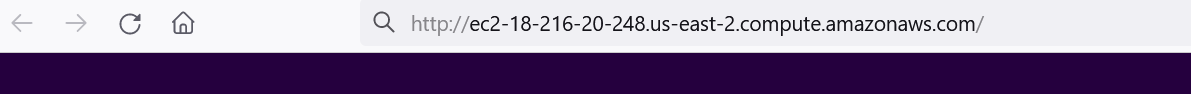
Run the Ansible playbook to configure the server for HTTPS and enable HTTP to HTTPS redirection.

ansible-playbook -i hosts install.yml

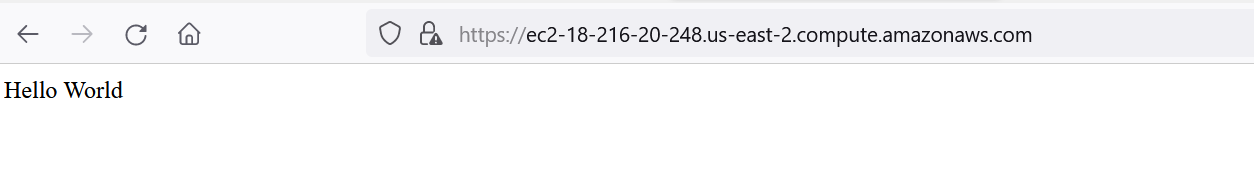
## Step 5: Test the Redirection

After running the playbook, access your website using HTTP (e.g., http://your\_server\_domain/). It should automatically redirect to HTTPS (e.g., https://your\_server\_domain/) using the self-signed SSL certificate.

**Desired Output => http to https redirection using web server**

****

**After hitting the above http link the site is redirected to https(self signed certificate)**

****

**Note:** Browser warnings about self-signed certificates are expected because they are not signed by a trusted certificate authority (CA). You can add a trusted SSL certificate for production use.

Designing a scalable and secure static web application in AWS involves several key steps.

1. S3 Bucket for Static Content:

Create an Amazon S3 bucket to store your static content (HTML, CSS, JavaScript, images, etc.). Enable versioning for backup and recovery.

2. CloudFront for Content Delivery:

Use Amazon CloudFront as a Content Delivery Network (CDN) to cache and serve content globally. This enhances performance and reduces latency for end-users.

3. Route 53 for DNS:

Register a domain name (e.g., example.com) and configure DNS using Amazon Route 53. Set up an alias record pointing to the CloudFront distribution.

4. SSL/TLS Certificates:

Obtain and install an SSL/TLS certificate from AWS Certificate Manager (ACM) for secure HTTPS connections. Configure the certificate with the CloudFront distribution.