Nginx HTTPS Server Using Ansible

By: Er. Vikas Nehra (M. Tech, B. Tech), Experience: 15 + Years

Session - 44 Agenda:

Nginx HTTPS Server Using Ansible:

Here you will learn how to set up or enable Nginx HTTPS (SSL) Web server with self-signed SSL certificate on Linux RHEL 9 using Ansible. A self-signed certificate will not validate the identity of your server, since it is not signed by a trusted certificate authority, but it will allow you to encrypt communications between your server and your visitors.

Let's create an ansible playbook to configure Nginx HTTPS server at the managed node(s).

$ vim nginx-https.yml

```
---
- name: Nginx HTTPS Server Configuration Playbook
  hosts: node1
  become: true
  tasks:
    - name: Setting up the static hostname in the server machine.
      hostname:
        name: nginx.nehraclasses.in
        use: systemd

    - name: Installing Nginx & OpenSSL packages in the machine.
      dnf:
        name:
          - nginx
          - openssl
          - mod_ssl
        state: latest

    - name: Creating /etc/pki/nginx/private/ directory for the SSL private key file.
      file:
        path: /etc/pki/nginx/private/
        mode: '0755'
        state: directory

    - name: Copying the nginx configuration file to /etc/nginx directory.
      template:
        src: /home/vikasnehra/nginx.conf.j2
        dest: /etc/nginx/nginx.conf
        force: true

    - name: Generating the Private Key File with the name as server.key
      openssl_privatekey:
        path: /etc/pki/nginx/private/server.key
        size: 2048

    - name: Generating Certificate Sign Request (CSR)
      community.crypto.x509_certificate:
        path: '/etc/pki/nginx/server.crt'
        privatekey_path: '/etc/pki/nginx/private/server.key'
        force: true
        provider: selfsigned

    - name: Creating the website index file.
      copy:
        dest: "/usr/share/nginx/html/index.html"
```

Nginx HTTPS Server Using Ansible

By: Er. Vikas Nehra (M. Tech, B. Tech), Experience: 15 + Years

```
    content: |
      <h1>Nehra Classes Are Awesome.</h1>
      <i>This page is hosted on node1 machine using nginx.</i>
    force: true

  - name: Allowing HTTPS traffic in the firewall.
    firewalld:
      service: https
      zone: public
      permanent: true
      immediate: true
      state: enabled

  - name: Starting & enabling the nginx service.
    service:
      name: nginx
      state: started
      enabled: yes
...
```

Before we execute this playbook, we would require the following collections which need to be installed in the Ansible server.

$ ansible-galaxy collection install ansible.posix
$ ansible-galaxy collection install community.general
$ ansible-galaxy collection install community.crypto

We would require the python3-pip packages also, so that we can install the cryptography packages using pip command later.

$ sudo dnf install -y python3-pip-21.2.3-6.el9.noarch

cryptography is a package which provides cryptographic recipes and primitives to Python developers. Our goal is for it to be your “cryptographic standard library”.

You can install cryptography using pip command.

$ pip install cryptography==37.0.0

We would require the python3-pip packages on the managed node as well, so that we can install the cryptography packages using pip command later there. We can do it using ansible ad-hoc command (or we could have mentioned the same in the playbook as well).

$ ansible node1 -m command -a 'sudo dnf install -y python3-pip-21.2.3-6.el9.noarch'

Now, you can install cryptography using pip command in form of Ansible ad-hoc command (use sudo for the root).

$ ansible node1 -m command -a 'sudo pip install cryptography==37.0.0'

Now, we can execute the playbook to setup the Nginx HTTPS server on the managed node1.

$ ansible-playbook nginx-https.yml

Now, you can open any web browser and mention the IP Address (hostname if you have an authoritative DNS in your environment) with HTTPS.

https://192.168.229.129/

You can see the web page hosted on the manage node1.

Thank You