Complete Guide to Installing NGINX and Configuring a Self-Signed SSL Certificate on Ubuntu

This guide provides step-by-step instructions for installing NGINX on an Ubuntu system and setting up a self-signed SSL certificate to enable HTTPS. It is tailored for Ubuntu 20.04 LTS or later, but the steps are generally applicable to other recent Ubuntu versions.

Prerequisites

Before you begin, ensure you have:

- An Ubuntu server (20.04 LTS or later recommended)
- A user account with `sudo` privileges
- Terminal access
- Basic command-line knowledge
- An internet connection for downloading packages

Part 1: Installing NGINX

NGINX is a high-performance web server that can also serve as a reverse proxy, load balancer, or HTTP cache. Follow these steps to install it.

Step 1: Update the Package Index

Ensure you have the latest package information:

```
sudo apt update
```

Step 2: Install NGINX

Install NGINX and its dependencies:

```
sudo apt install nginx
```

Step 3: Verify NGINX Installation

Check if NGINX is running:

```
sudo systemctl status nginx
```

You should see `active (running)` in the output. If NGINX is not running, start it:

```
sudo systemctl start nginx
```

Enable NGINX to start on boot:

```
sudo systemctl enable nginx
```

Step 4: Test NGINX

Open a web browser and navigate to `http://your_server_ip`. To find your server's IP address, run:

```
ip addr show | grep inet
```

You should see the default NGINX welcome page, confirming the installation.

Part 2: Configuring a Self-Signed SSL Certificate

A self-signed SSL certificate enables HTTPS but will trigger browser warnings since it's not signed by a trusted Certificate Authority (CA). This is suitable for testing or internal use.

Step 1: Install OpenSSL

OpenSSL is required to generate the certificate. It's typically pre-installed, but verify with:

```
sudo apt install openssl
```

Step 2: Generate the Self-Signed Certificate

Create a directory for the certificate and key:

```
sudo mkdir /etc/nginx/ssl
```

Generate a self-signed certificate and private key (replace `example.com` with your domain or server IP):

```
sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout
/etc/nginx/ssl/nginx.key -out /etc/nginx/ssl/nginx.crt
```

You'll be prompted for certificate details (e.g., country, organization). For the **Common Name (CN)**, enter your domain (e.g., `example.com`) or server IP. Press `Enter` to skip optional fields.

- `-x509`: Creates a self-signed certificate

- `-nodes`: Skips passphrase for the private key
- `-days 365`: Sets certificate validity to 1 year
- `-newkey rsa:2048`: Generates a 2048-bit RSA key
- `-keyout`: Specifies the private key file
- `-out`: Specifies the certificate file

Step 3: Configure NGINX for HTTPS

Edit the NGINX configuration file (typically `/etc/nginx/sites-available/default`):

```
sudo nano /etc/nginx/sites-available/default
```

Replace the default server block with:

```
server {
    listen 80;
    listen [::]:80;
    server_name example.com www.example.com;

    # Redirect HTTP to HTTPS
    return 301 https://$host$request_uri;
}

server {
    listen 443 ssl;
    listen [::]:443 ssl;
    server_name example.com www.example.com;

    ssl_certificate /etc/nginx/ssl/nginx.crt;
    ssl_certificate_key /etc/nginx/ssl/nginx.key;

    root /var/www/html;
    index index.html index.htm;

    location / {
        try_files $uri $uri/ /index.html;
    }
}
```

- Replace `example.com www.example.com` with your domain or server IP.
- The first block redirects HTTP (port 80) to HTTPS (port 443).
- The second block enables HTTPS with the self-signed certificate.

Save and exit (`Ctrl+O`, `Enter`, `Ctrl+X` in `nano`).

Step 4: Test NGINX Configuration

Verify the configuration:

```
sudo nginx -t
```

Expected output:

```
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
```

Fix any errors reported before proceeding.

Step 5: Reload NGINX

Apply the changes:

```
sudo systemctl reload nginx
```

Step 6: Adjust Firewall (if applicable)

If using `ufw`, allow HTTPS traffic:

```
sudo ufw allow 'Nginx HTTPS'
```

Check the firewall status:

```
sudo ufw status
```

Step 7: Test HTTPS

Visit `https://your_server_ip` or `https://your_domain` in a browser. Expect a warning about the self-signed certificate (e.g., “Your connection is not private”). Click “Advanced” and proceed.

Verify the certificate from the terminal:

```
openssl s_client -connect your_server_ip:443
```

Press `Ctrl+C` to exit after viewing certificate details.

Part 3: Security Considerations

- **Self-Signed Certificate Limitations**: Browsers will display warnings. For production, use a trusted CA like Let’s Encrypt.

- **File Permissions**: Secure the certificate and key:

```
sudo chmod 600 /etc/nginx/ssl/nginx.key
sudo chmod 644 /etc/nginx/ssl/nginx.crt
```

- **Regular Updates**: Keep NGINX and OpenSSL updated:

```
sudo apt update && sudo apt upgrade
```

- **Backup Certificates**: Store `/etc/nginx/ssl/nginx.key` and `/etc/nginx/ssl/nginx.crt` securely.

Part 4: Troubleshooting

- **NGINX Fails to Start**: Check status (`sudo systemctl status nginx`) and logs (`sudo journalctl -u nginx`).
- **Certificate Issues**: Verify file paths in the NGINX configuration.
- **Browser Warnings**: Normal for self-signed certificates.
- **Port Conflicts**: Check for services using ports 80 or 443:

```
sudo netstat -tuln | grep ':80\|:443'
```

Part 5: Optional - Serving a Sample HTTPS Page

Create a test HTML page:

```
sudo nano /var/www/html/index.html
```

Add:

```
<!DOCTYPE html>
<html>
<head>
    <title>Welcome to NGINX with HTTPS</title>
</head>
<body>
    <h1>Success! Your NGINX server is running with HTTPS.</h1>
</body>
</html>
```

Save, exit, and reload NGINX:

```
sudo systemctl reload nginx
```

Visit `https://your_server_ip` to view the page.

Conclusion

You’ve installed NGINX on Ubuntu and configured a self-signed SSL certificate for HTTPS. This setup is ideal for testing or internal applications. For production, consider a trusted certificate to avoid browser warnings. Refer to the NGINX documentation for advanced configurations.