# Lab: Web Server Deployment

## Overview

Infrastructure deployment is an important role for the systems administrator. Public-facing infrastructure such as web servers present unique security and operational challenges compared to endpoints. Configuring a web server requires an understanding of firewall, networking design, network protocols, and security practices.

## Scenario

GlobeX HR division wants to launch a new employee information website. This website will be public-facing, as it does not contain sensitive information, but rather helpful information about the various services available to employees. You've joined the project team and are tasked with standing up some prototype infrastructure for the web developer to work with. "All I know is Wordpress!" the web developer exclaims. "If you can get me a web server and remote access to it, I'll be all set to begin front-end development."

## Prerequisites

* A pfSense VM in VirtualBox, free from configuration settings from previous labs
* A user endpoint VM in VirtualBox (any existing Windows 10 or Linux VM)

## Objectives

* Deploy an NGINX web server on Linux Server behind pfSense
* On pfSense, setup port forwarding to route traffic on ports 8000 to the web server
* Create firewall rule exceptions for port 8000 and 22
* Add SSH and open up port 22 for administration purposes
* Deploy UFW firewall and activate it
* Allow correct traffic through pfSense
* Test and validate access using a computer on the other side of the pfSense firewall
* Submit an updated network topology

## Resources

* [Ubuntu Server Download](https://ubuntu.com/download/server){:target="\_blank"}
* [Ubuntu Official Documentation - Install NGINX](https://ubuntu.com/tutorials/install-and-configure-nginx#1-overview){:target="\_blank"}
* [PfSense Port Forwarding](https://docs.netgate.com/pfsense/en/latest/nat/port-forwards.html){:target="\_blank"}

## Tasks

### Part 1: Topology 1/2

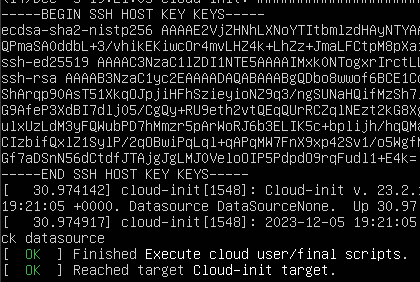
Read through the entire lab and use Draw.io to create an appropriate topology of the network you expect to construct. Include as many details as you can such as computer names, OS types, IP addresses, etc. Include a screenshot of this initial topology.

### 

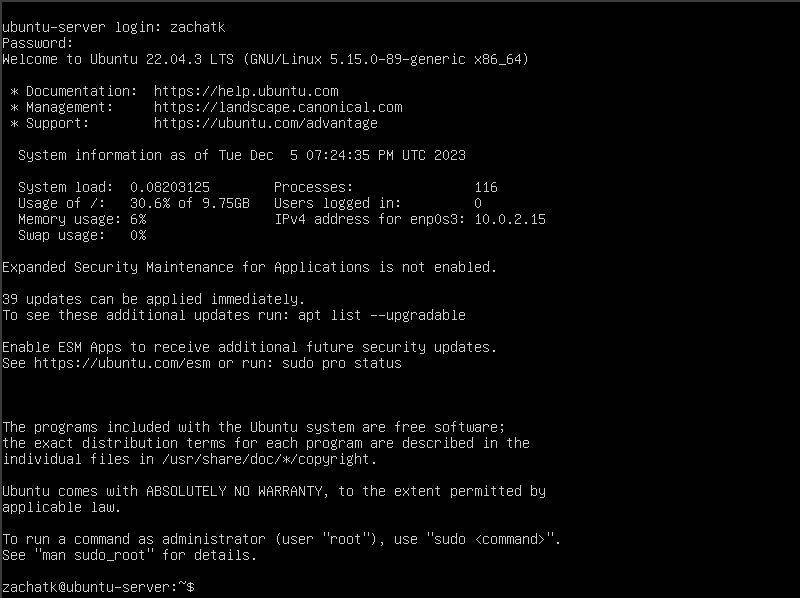
### Part 2: Staging

Submit detailed documentation regarding all of the configurations in this section.

* ~~First you will need a fresh pfSense VM, free from configuration settings from previous labs. You can reset an existing instance to factory settings (Diagnostics / Factory Defaults), revert to a baseline snapshot, import a fresh instance from a baseline OVA backup, or install pfSense on a new VM. However you achieve this, it is important to start from a clean baseline to avoid complications.  
  On the pfSense VM, configure the WAN network adapter to NAT Network and the LAN adapter to Internal Network.~~
* ~~Second you will need to create a new Ubuntu Server VM behind pfSense. This VM will host NGINX and be a webserver.~~
  + ~~Download the~~ [~~Ubuntu Server ISO~~](https://ubuntu.com/download/server)~~.~~
  + ~~Create a new VM in VirtualBox.~~
  + ~~Install the latest version of Ubuntu Server.~~
    - ~~Install OpenSSH during the OS installation process.~~



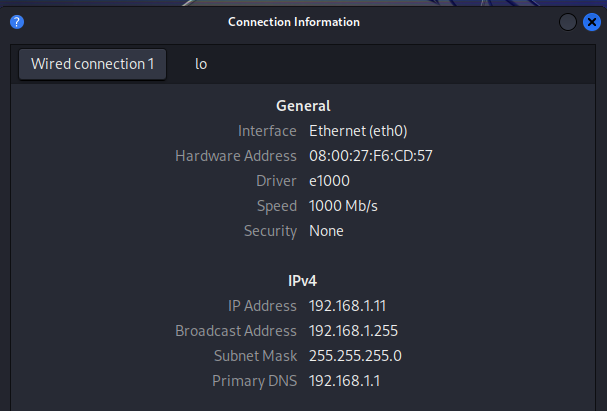
* + ~~Reboot when installation is complete.~~

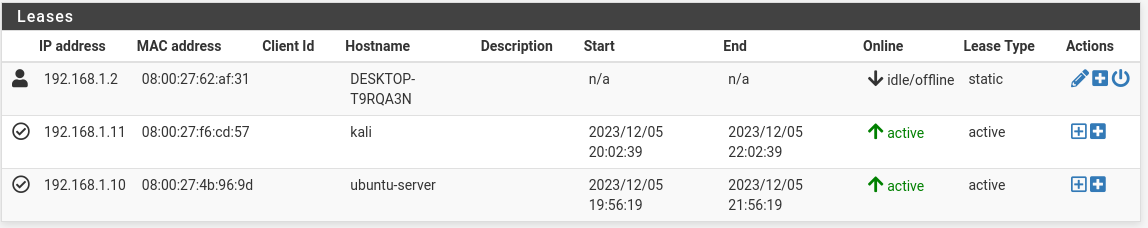


* ~~On the Ubuntu Server VM, configure the network adapter to match the LAN adapter of pfSense (should be set to the same Internal Network).~~



* ~~Finally, you will also need a user endpoint VM with a GUI (Windows 10 or Kali) for configuring pfSense and testing the webserver.  
  On the user endpoint VM, configure the network adapter to match the LAN adapter of pfSense and the Ubuntu Server (should be set to the same Internal Network).~~





**This part was as expected.**

### **Part 3: NGINX Web Server Setup**

In this step, you’ll deploy an NGINX web server on Linux Server behind pfSense.

Complete the following steps:

**I didn’t bother with SSH, its too much of a pain, I just did this manually.**

* ~~Install NGINX:~~

sudo apt update

sudo apt install nginx

* ~~Creating our own website:~~

cd /var/www

sudo mkdir globex

cd globex

sudo touch index.html

* ~~Code the new web page:~~
  + ~~Open the html file for editing using nano (~~**~~sudo nano index.html~~**~~) and copy the following to the~~ **~~index.html~~** ~~file:~~

<!doctype html>

<html>

<head>

<meta charset="utf-8">

<title>Hello via Nginx!</title>

</head>

<body>

<h1>Hello via Nginx!</h1>

<p>We have just configured our Nginx web server to serve HTML on Ubuntu Server!</p>

</body>

</html>

* ~~Set up virtual host:~~

cd /etc/nginx/sites-enabled

sudo touch globex

sudo nano globex

* + ~~Once you’ve opened the~~ **~~globex~~** ~~file, copy in the following:~~

server {

listen 8000;

listen [::]:8000;

server\_name employees.globex.com;

root /var/www/globex;

index index.html;

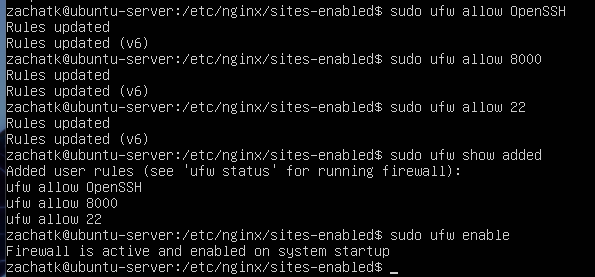
location / {

try\_files $uri $uri/ =404;

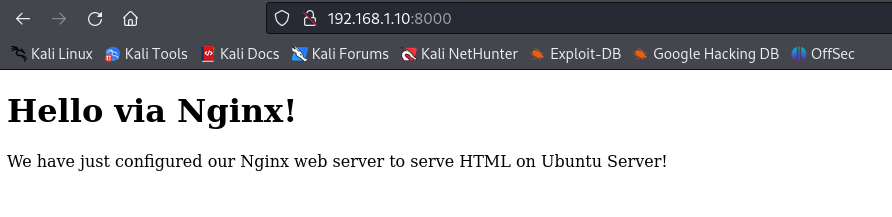
}

}

* ~~Activate virtual host and test results:~~
  + ~~Check to see that nginx is running~~ **~~sudo systemctl status nginx~~**
  + ~~If it is not running, start it with:~~ **~~sudo systemctl start nginx~~**
  + ~~If it is already running, you may need to restart it with:~~ **~~sudo systemctl restart nginx~~**
  + ~~Open up ports 22, 8000 on UFW.~~
    - ~~Activate the Ubuntu Server’s UFW.~~
    - ~~Include a screenshot of UFW settings while you’re shelled into Ubuntu Server.~~

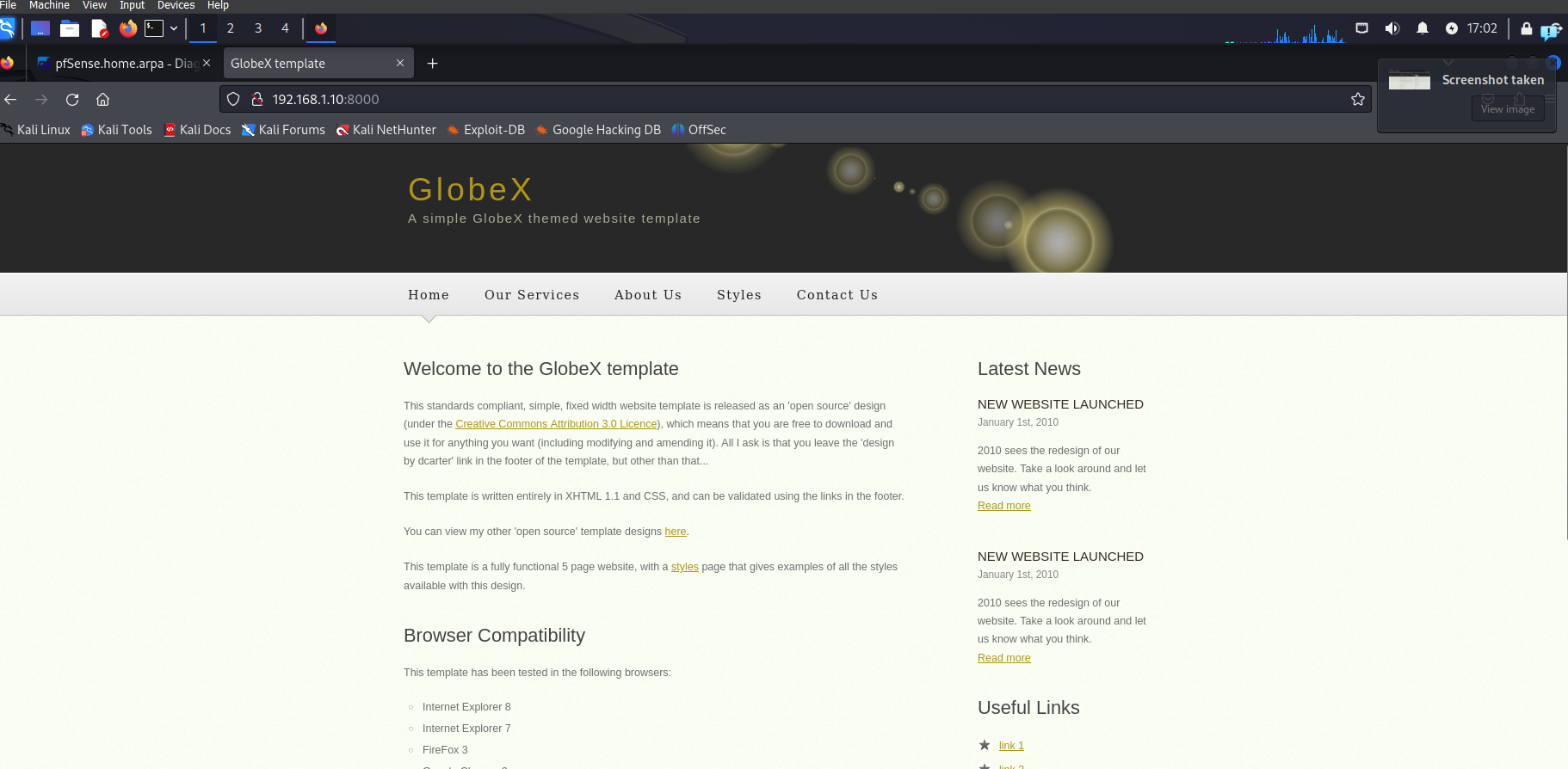


* + ~~Test that your webpage is available by navigating to~~ **~~http://<ip of NGINX server>:8000~~** ~~on a VM on the same subnet as the webserver.~~
    - ~~It should should show “Hello via Nginx!” as its headline. Include a screenshot.~~



**Like with Part 1, the instructions here were clear enough that I didn’t have to do anything crazy.**

* ~~Load the Globex website files:~~
  + ~~Clone the class repo to the home directory of your Ubuntu Server.~~
  + ~~Recursively copy the assets from class-07/lab/starter-code folder to~~ **~~/var/www/globex~~**~~.~~
    - ~~This will overwrite the~~ **~~index.html~~** ~~file you created in step 3.~~
  + ~~Access the web page using a browser (remember we used port 8000) and include a screenshot.~~



**To get this to work, I copied the entire repo (Seattle-ops-301d10.git) instead of trying to just download the folder. It was easier, and having access to the repo from my VM might help in the future.**

**I then simply renamed the previous globex folder, and mv’d the and renamed the starter-code. Again, lets just make this simple.**

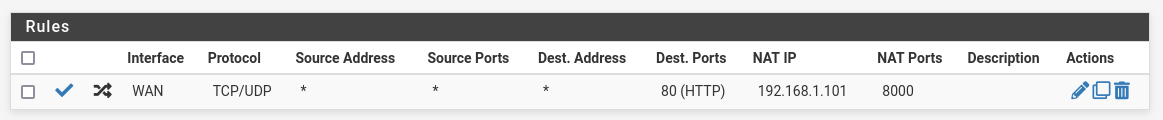
### **Part 4: Configuring pfSense**

Now that the web server is deployed, you’ll need to configure the pfSense perimeter to redirect inbound HTTP requests to the new VM’s IP address.

* ~~Set your pfSense’s WAN interface to NAT Network on VirtualBox.~~
* ~~Login to pfSense web UI and navigate to Interfaces > WAN. Scroll to the bottom and set~~ *~~Block private networks and loopback addresses~~* ~~and~~ *~~Block bogon networks~~* ~~to disabled. These default firewall rules clash with our VirtualBox NAT Network configuration.~~
* ~~Determine the web server’s MAC address and add a static DHCP mapping to pfSense for the NGINX web server.~~

**I used the pfsense UI for this**

* ~~In Firewall > NAT, configure pfSense to route incoming HTTP requests on WAN port 80 to the NGINX web server port 8000.~~



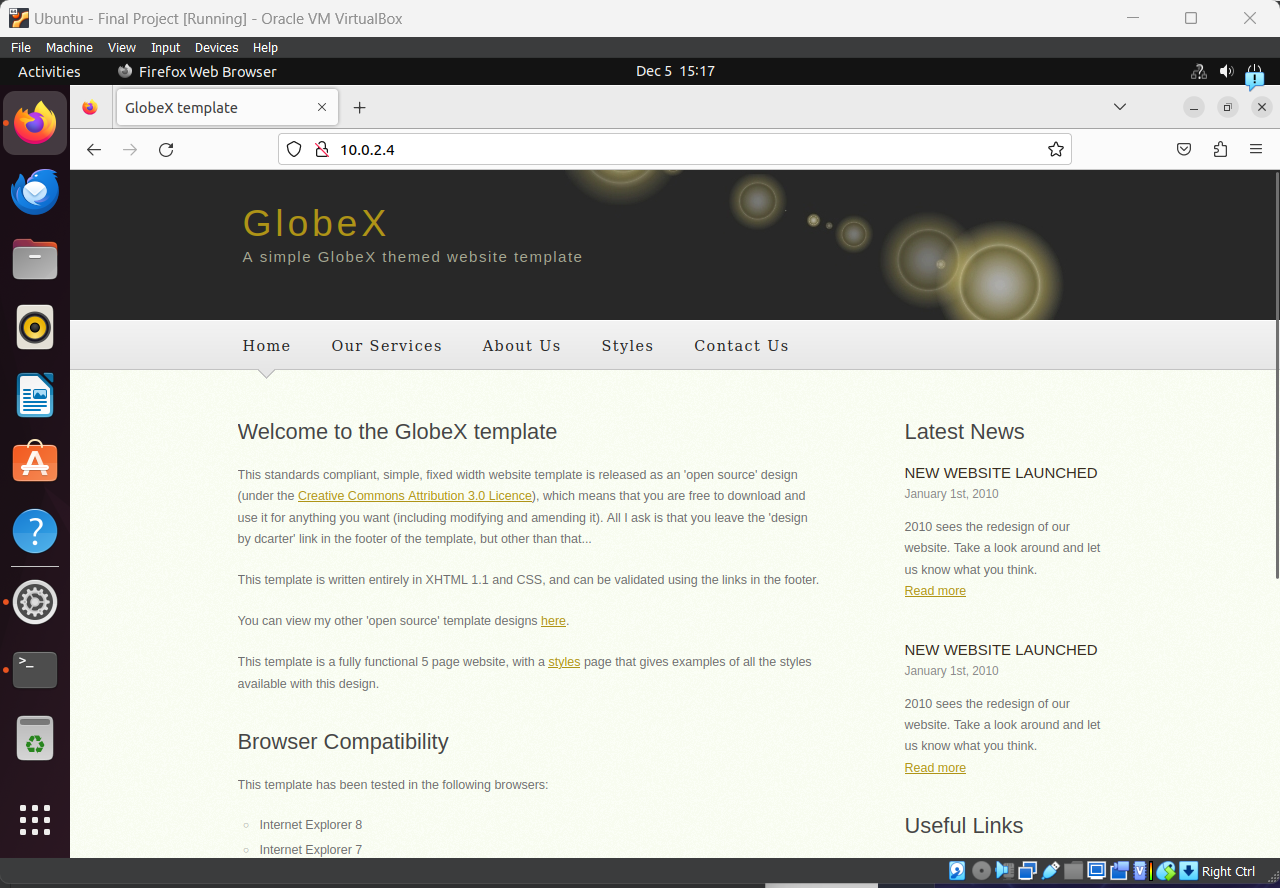
* ~~Now that the web server has been deployed we can access the web server from outside the network on a device that’s also set to NAT Network.~~

### **Part 5: Testing**

* ~~Identify your pfSense’s WAN address on its main dashboard screen towards the bottom right. You’ll need it for the upcoming step.~~
* ~~Move Kali Linux outside the LAN by setting its network adapter to NAT Network.~~

**Here is where I diverted. Instead of moving kali (and if anything didn’t work losing access to pfsense UI) I just spun up another VM. Also, had to tell multiple people this, set a static IP from the VM for this step. Pfsense cannot set a WAN IP address.**

* ~~Test and validate that your web page is accessible from outside the LAN by querying pfSense’s WAN interface using a browser.~~
  + *~~You can explicitly specify the port using~~* ***~~http://<WAN ip of pfSense>:80~~*** *~~or, because http traffic uses port 80 by default, you can use~~* ***~~http://<WAN ip of pfSense>~~*** *~~and your browser should automatically connect to port 80.~~*
* ~~Include a screenshot of the successful browser test.~~



Congratulations! You’ve set up your first web server properly behind a firewall.

### **Part 6: Topology 2/2**

When the other tasks are complete, review the topology and update, revise, extend, or add details as necessary.

Was your initial topology accurate to the finished product? Why or why not?

**No my initial topography was wrong: because I didn’t initially spin up another VM.**

