Lesson Plan  
VPN Lab

short line

# Summary

1. Goals
2. Preparation
3. What is VPN?
4. How is it different from HTTPS and SSL/TLS
5. AWS Lab

# Goals

* Learn about VPN
* Understand use cases of VPN
* Make your own VPN server, and connect to your private network

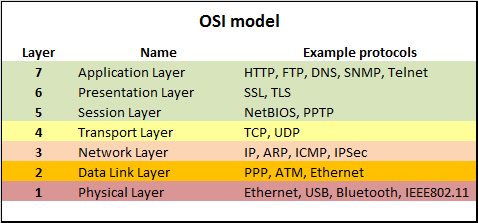
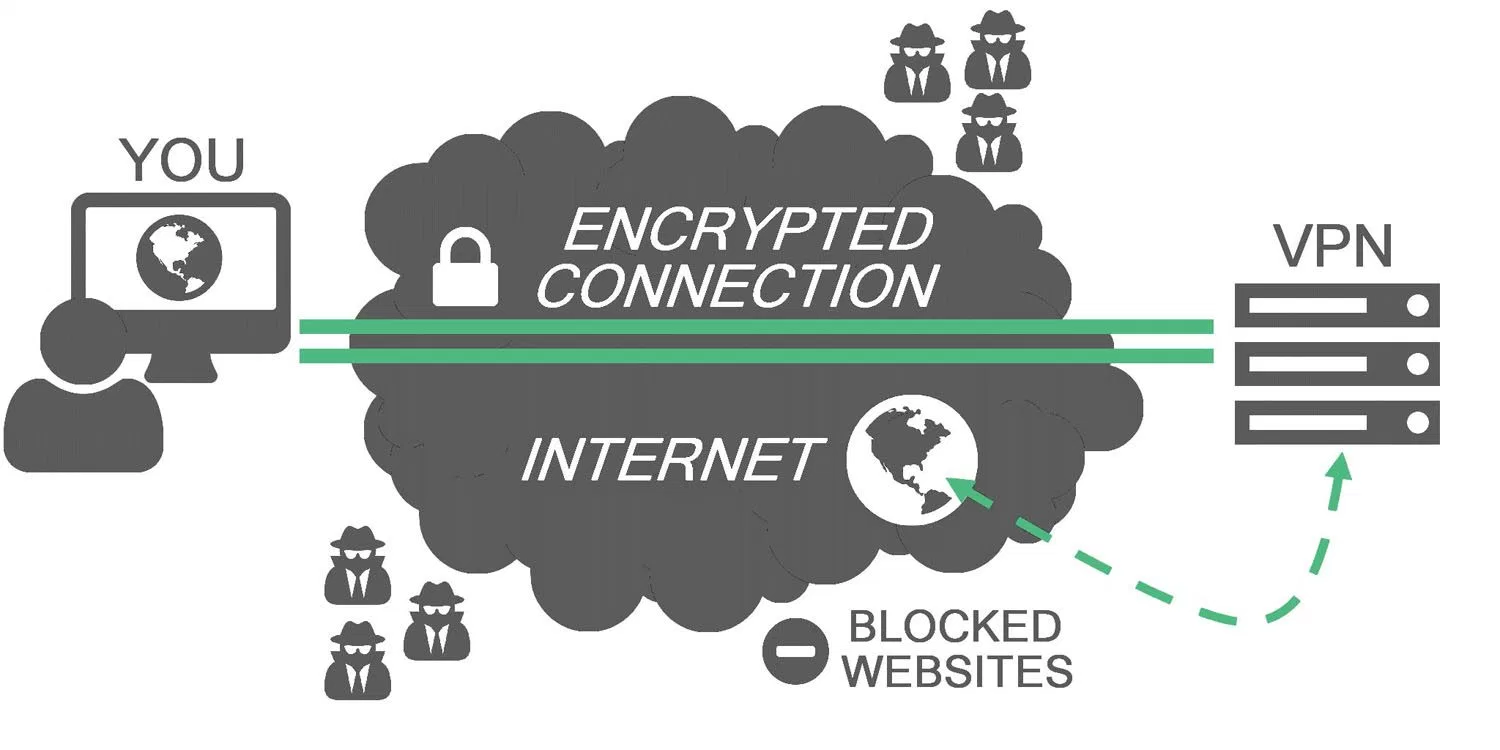
# Preparation

* AWS Account
* AWS VPC is created
* Kali Linux running as VM

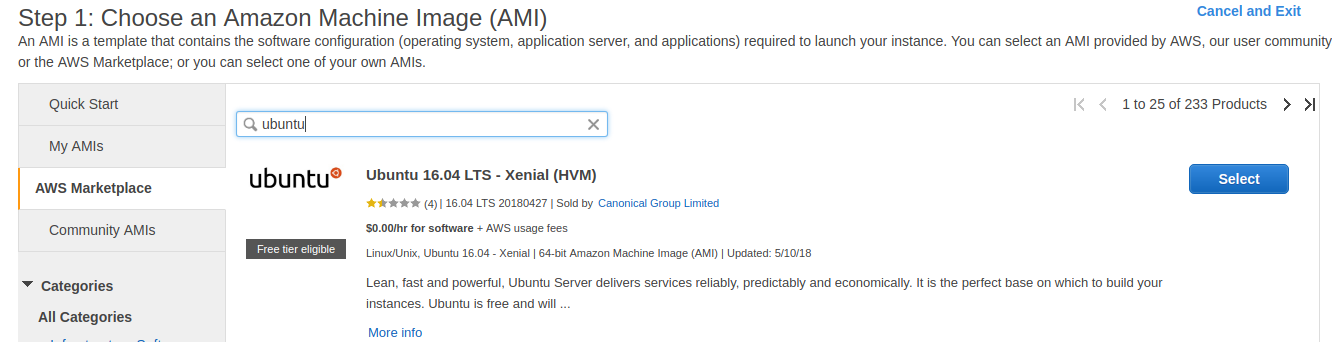
# What is VPN?

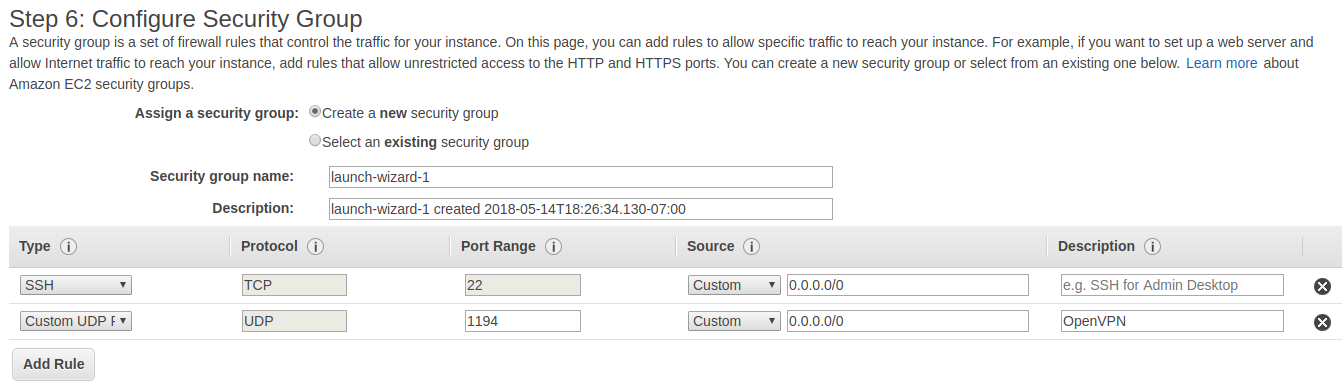
* A VPN, or Virtual Private Network, allows you to create a secure connection to another network over the Internet. VPNs can be used to access region-restricted websites, shield your browsing activity from prying eyes on public Wi-Fi, and more.

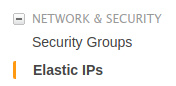
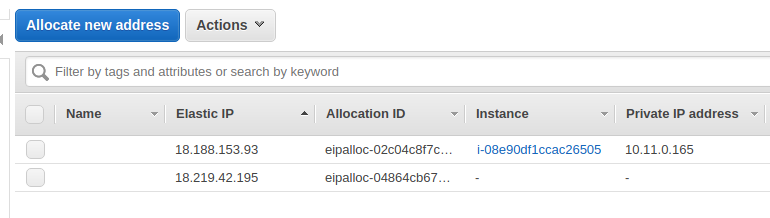
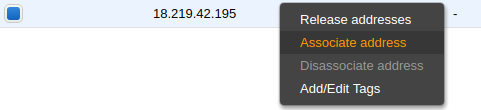
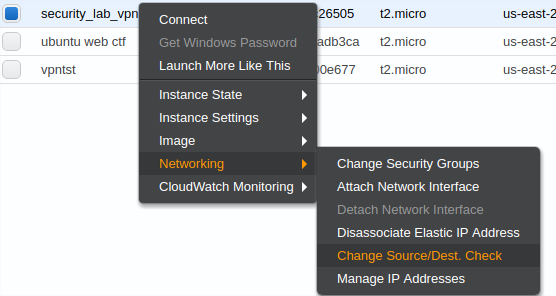
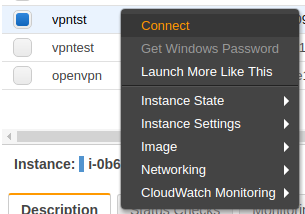
# How is it different from SSL/TLS or HTTPS?

* VPN, SSL/TLS, HTTPS are all used for confidentiality of information sent over the internet. However, there are subtle differences.
* SSL/TLS is a standard security protocol for establishing encrypted links between a web server and a browser in an online communication.
* HTTPS is simply running HTTP on top of SSL/TLS. Meaning SSL/TLS is doing the encryption work for HTTP making it HTTPS.
  + 
  + This may explain how HTTP is running on top of SSL/TLS
* This means that HTTPS alone is very secure because it would encrypt all browsing data as it leaves your computer. Only privacy concern is that if you were browsing Facebook or youtube, they would know your IP address. Despite the fact that data is encrypted, does not mean that the entire packet is encrypted. I still has your source IP and Destination IP in order to get to destination and respond back.
* VPN is in a way stronger security mechanism because it does not matter which protocol you are using. (Telnet, HTTP, FTP) VPN will encrypt everything and forward it to the designated VPN server, and from VPN server, it will forward traffic to the destination. This means that if you cannot access netflix from certain part of country, you can have VPN in United States, and access that website, because the website will think that you are browsing from US!
  + 

# AWS VPN Lab

* **Setup**
  + We will be using a setup-script from a github repo
    - <https://github.com/Angristan/OpenVPN-install>
  + Thankfully someone created a full installation script for us.
  + All we need to do is to launch a ubuntu 16.04 instance and run this script.
  + **Launch Instance : search ubuntu in AWS Marketplace**
  + 
  + **Launch Ubuntu 16.04 LTS - Xenial** 
    - **Free tier**
    - **Network - select your VPC**
    - **Auto-Assign Public IP - Disable**
      * **We will associate an elastic IP address later.**
    - Click Next until Security Group
    - **Add rule to open port for openvpn service to public interface**
      * **Type: Custom UDP Rule**
      * **Protocol: UDP**
      * **Port Range: 1194**
      * **Source: 0.0.0.0/0**

****

* + - **Launch!**
* **Elastic IP**
  + **Make sure you attach an elastic IP address to this instance so that its public IP address does not change every time you reboot the instance.**
  + ****
  + **Associate New address if you don’t have one yet.**
  + ****
  + **Right click an elastic IP and click ‘Associate address’**
  + ****
    - * **Then pick your VPN server instance to be associated with this public IP.**
  + **SSH** 
    - Before you ssh into the instance make sure you ***Disable Source/Dest Check***
    - 
    - To SSH into the instance, Instructions can be found in **Connect** if you right click your instance
    - 
  + In order to ssh in, you must have a private / public key pair associated with your instance.
    - If you are unsure of this step Check the tutorial below
      * <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-key-pairs.html>
* **VPN Setup**
  + ***Wget*** [***https://raw.githubusercontent.com/Angristan/OpenVPN-install/master/openvpn-install.sh***](https://raw.githubusercontent.com/Angristan/OpenVPN-install/master/openvpn-install.sh)
  + ***Chmod +x openvpn-install.sh***
  + ***Sudo ./openvpn-install.sh***
* Once the script finishes running, you should have openvpn server running with client.ovpn file in your home directory.
* If you plan to create a lot of users add the lines below to /etc/openvpn/client-common.txt
  + Otherwise just add these lines to your .ovpn file
  + Script-security 2
  + up /etc/openvpn/update-resolv-conf
  + down /etc/openvpn/update-resolv-conf
* Client.ovpn file is the config file for your clients to use your vpn service, thus if you want to use vpn from your machine, you need to transfer this config file to your own machine.
  + ***scp -i [key pair location] [username]@[ip address]:[client.ovpn full path on remote machine] [copy location in your own computer]***
  + Ex) ***scp -i ~/.ssh/keys/sugo\_public.pem ubuntu@18.220.74.21:/home/ubuntu/client.ovpn /home/sugo/Documents/***
* Once you’ve transferred ovpn file, you can simply connect to VPN with this command
  + ***Sudo openvpn client.ovpn***

Now you are inside VPC

Check your ip address with ifconfig command

All your traffic should be forwarded through a virtual interface called **tun0**

**Tun0** interface ip address is your ip address in VPC

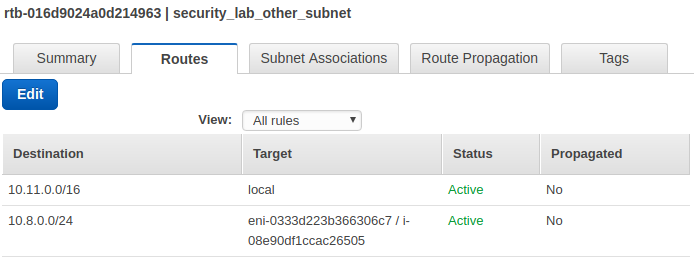
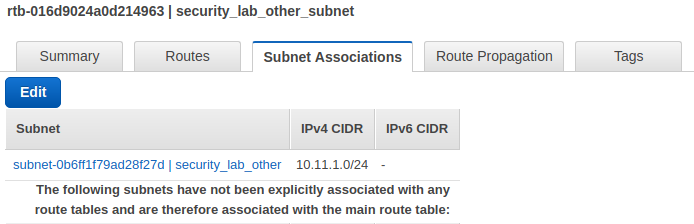
Google “what is my ip” and you should get the public ip address of your VPN server instance.

If you have other instances running in this VPC, you should be able to ping them with its private IP addresses.

UPDATE ------------

VPN clients are essentially running behind a VPN server acting as a router connecting two different networks. In order for other VPC instances to ping VPN clients with its private IP addresses, you must add a route to **Route Table** specifying that your VPN client subnet (default 10.8.0.0/24) to be routed to your VPN server.

Here’s what I did

* I created a separate subnet for the other instances that I want to communicate with through VPN. (ex. 10.11.0.0/24)
  + Main subnet - 10.11.0.0/24 (contains VPN server)
  + 
  + Other subnet - 10.11.1.0/24 (contains lab machines)
  + 
* For the ‘Other’ subnet. We will create our custom route table so that anything directed to 10.8.0.0/24 will be forwarded to our VPN server instance.
  + 
  + Make sure you associate the ‘other’ subnet with this route table
  + 
* This should solve the problem where VPC instances cannot ping 10.8.0.0 network.

# Troubleshoot

* If you cannot resolve DNS
  + Make sure your Kali VM is in NAT mode not Bridged or other network mode.