Using the Raspberry Pi to establish a Virtual Private Network (VPN) Connection to a Home Network

Constadinos Lales
Computer Engineering Technology
New York City College of Technology, CUNY
186 Jay Street, Brooklyn, NY 11201
Costa.Lales@mail.citytech.cuny.edu

Aparicio Carranza, PhD
Computer Engineering Technology
New York City College of Technology, CUNY
186 Jay Street, Brooklyn, NY 11201
acarranza@citytech.cuny.edu

Abstract - Because of the advances in technology, people are able to bring computer devices such as laptops, tablets, and smart phones with them anywhere they go. In addition, they are able to connect to various networks out there in the public to obtain internet access. With this luxury, these people run into, serious, problems. One of these problems is security. When accessing public internet, the data transferred from one's computer is not encrypted and is available to anyone who has some knowledge of computer networking. In addition, these networks may take away from our freedom of web browsing by blocking different websites. They can also view every website that one goes on while connected to their internet. All of these problems can be solved by setting up a virtual private network (VPN). A VPN is a network that uses encryption to securely connect two different networks together using public telecommunication such as the internet. To establish a VPN connection, one needs to connect to a server. In this paper, we will be describing how we used the Raspberry Pi (A cheap microcomputer) as a VPN server to a home network; in order to create a VPN connection between a home network and the public internet.

Keywords - Raspberry Pi; VPN(*Virtual Private Network****); OpenVPN***

I. Introduction

Raspberry pi is a small credit card sized computer that includes ports such as HDMI, Ethernet, 2 USB's version 2.0, Audio, and RCA Video. In addition, Raspberry Pi includes a SanDisk card slot which is used as the Pi's storage and GPIO (*General Purpose Input/Output*) pins which can be programmed using python. There are two models of the Pi that is available to purchase in different electronic sites. Model A comes with a 256MB RAM and costs $25 and Model B comes with 512MB RAM and costs $35. There are several, Linux based, operating systems available for the Pi that can be downloaded online and written on the SanDisk card. Each operating system has its pros and cons. The one to choose from depends on what a user wants to use the raspberry pi for. The Pi operates at 700MHz by default, but can be overclock to 900MHz. Furthermore, the Pi is powerful enough to support videos in 1080p using OpenGL ES 2.0 and hardware-accelerated OpenVG [1].

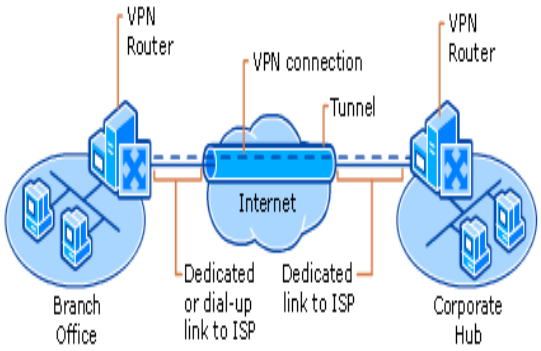
After writing one of the operating systems to the raspberry pi and placing the SanDisk in the Pi's slot, one is ready to utilize the Pi. The Pi can be accessed by attaching a HDMI cable from the Pi to the TV and a keyboard. However, if those items are unavailable to a person,

one can also connect to the Pi via SSH (Secure Shell) using putty. SSH is a secure way to remotely connect to a command line. This is done by either connecting the Pi to a PC and sharing you internet or connecting it to your router. This will provide the Pi with an IP address that is used to connect via SSH. Putty also offers X11 forwarding which allows you to use an xming server to open up the GUI (*Graphical User Interface*) of the Pi by using the "***lxsession***" command on the Pi's command line. Thus, one is able to fully access the Pi with just an Ethernet cable, a five volt micro USB charger, the computer application putty, and xming. When the Pi boots up for the first time it detects that there is no configurations made and tells the user to use the "***sudo raspi-config***" to configure it. The most important configuration to be done on the Raspberry Pi is expanding the root file system. By default the Pi uses only 2GB of memory and not all the memory available to the SanDisk.

II. Understanding VPN

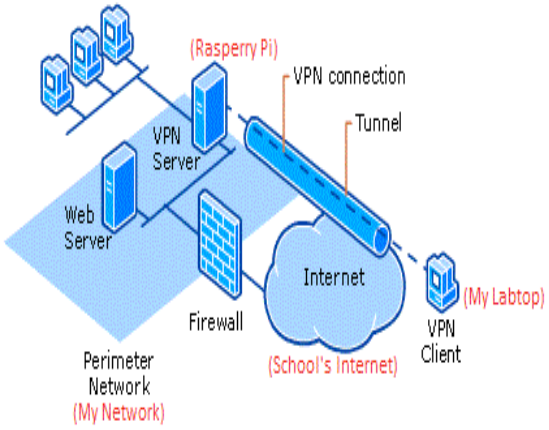
Now that the Raspberry Pi is setup, it is time to setup the VPN server. Before doing so, one must understand the tunnel protocols and types of VPN connection there are available in order to be able to choose the one that is right for them. Some of the most common VPN security technologies are Internet Protocol Security (IPSec), Secure Sockets Layer (SSL), Transport Layer Security (TLS), Point-to-Point Tunneling Protocol (PPTP), Layer 2 Forwarding (L2F), and Layer Two Tunneling Protocol (L2TP), The two basic types of Cisco VPNs one can have is Site-to-site and Remote access [2].

Large companies with offices all around the globe need a way to connect to their different offices and organizations together. The type of VPN that they would use to do that is Site-to-Site VPN. This allows organizations to have routed connections with separate offices, or with other organizations over the internet. This logically acts as a dedicated Wide Area Network link. When using this type of VPN, the protocol used for tunneling is IPSec. This is a security mechanism that encrypts any traffic supported by the IP protocol, such as Internet, e-mail, Telnet, and more. IPSec uses either digital certificates or pre-shared keys to provide authentication, data encryption and negotiation. To summarize, if one needs a network that holds multiple devices to another network with many devices on it, they would use Site-to-Site VPN and the protocol that would need to be configured is IPSec [3].



The type of VPN this paper is mostly interested in is Remote Access VPN that is when an individual host connects to a private network. Remote access VPN is a method of connecting one network with a single device to another network. This is usually used for travelers who need to access the company network securely over the internet. Though, we believe that it should be used not only by travelers, who want to work anywhere they go, but by anyone who wants to

access the internet securely in public without any restrictions. The protocols that can be setup with Remote access VPN is IPSec and SSL. SSL is a communication protocol that provides secure Internet-based on client to server interactions. Authentication is developed between the server to client by using public key cryptography and digital certificates. Once authentication is established the entire communication session is encrypted [3].



SSL VPN is a bit more challenging to set up compared to PPTP. PPTP is a Microsoft VPN technology that uses standard authentications protocols, such as Challenge Handshake Authentication Protocol (CHAP), or Password Authentication Protocol (PAP). PPTP does not encrypt data unless it is used along with other Microsoft encrypted mechanism. PPTP can be used by most operating systems and servers including android phones. Although PPTP offers an easy and fast way of connecting to a different network, the authentication protocols used are easier to crack. Thus, PPTP should not be used if one cares about security [3].

After discovering about all the security technologies out there, we had to make our decision on the tunneling protocols we were going to use to set up our VPN server. we were more interested in good security than fast setup and the type of VPN connection we wanted to implement was Remote access thus narrowing our option to SSL VPN. Next, we looked for software that supported SSL VPN connection and we stumbled upon OpenVPN.

III. OpenVPN Configurations

To create our VPN server on the Raspberry Pi, we decided to use OpenVPN. OpenVPN is a software application that establishes site-to-site or remote access connections through the custom use of security protocol SSL/Transport Layer Security (TLS). Additionally, OpenVPN is an open source application. To obtain the software OpenVPN on the Raspberry Pi, we used the command "*sudo apt-get install openvpn openssl*". The client we were using to connect to the Raspberry Pi server was Windows 7 Toshiba laptop. We went to the OpenVPN website and downloaded the software for our client.

Before building the keys and certifications, it is recommended to redirect the files in the Raspberry Pi "***/usr/share/doc/openvpn/examples/easy-rsa/2.0***" to the main directory "*etc/openvpn*" by using the command.

cp -r /usr/share/doc/openvpn/examples/easy-rsa/2.0 /etc/openvpn/easy-rsa

This is done to ensure that no modifications get overwritten when OpenVpn packages are upgraded. Next you enter the vars files located in the easy-rsa and export the top level easy-rsa tree by using the commands

nano easy-rsa/vars

and changing

export EASY_RSA=”pwd`” To *export EASY_RSA=”/etc/openvpn/easy-rsa”*

Then, we build the keys and certificates needed by using these commands on the Raspberry Pi.

./easy-rsa/clean-all
./easy-rsa/build-ca OpenVPN
./easy-rsa/build-key-server server
./easy-rsa/build-key client1
./easy-rsa/build-dh

Next, we needed to create the configurations for the server of the Raspberry Pi. In these configurations, we included the layer our device was connected in, the location of the server key and certificate, the protocol used to establish connection, the port number we wanted to use, the server’s ip address, how long we wanted the openvpn to attempt to establish connection, and the DNS servers we wanted to set.

nano openvpn.conf
dev tun
proto udp
port 1194
ca /etc/openvpn/easy-rsa/keys/ca.crt
cert /etc/openvpn/easy-rsa/keys/server.crt
key /etc/openvpn/easy-rsa/keys/server.key
dh /etc/openvpn/easy-rsa/keys/dh1024.pem
user nobody
group nogroup
server 10.8.0.0 255.255.255.0
persist-key
persist-tun
status /var/log/openvpn-status.log
verb 3
client-to-client
push “redirect-gateway def1"
#set the dns servers
push “dhcp-option DNS 8.8.8.8"
push “dhcp-option DNS 8.8.4.4"
log-append /var/log/openvpn
comp-lzo

we had to forward the Rasbperry Pi’s IP address to our public IP address and this is done by enabling IP forwarding and using the commands
echo 1 > /proc/sys/net/ipv4/ip_forward
nano sysctl.conf
edit “#net.ipv4.ip_forward=1” to “net.ipv4.ip_forward=1”

Next, We had to edit the Raspberry Pi’s IP tables to allow packets to be sent from the VPN server IP address, and the Raspberry Pi IP address along with the port we configured in the server configuration file. This is done by editing the Rasperry Pi’s ***rc.local*** file which runs when the Pi is booting up. In addition, the port used in the server configuration file needed to be port forward by the router that we were using.

sudo nano /etc/rc.local
then add to the end of the file before exit
iptables -t nat -A INPUT -i eth0 -p udp -m udp --dport 1194 -j ACCEPT
iptables -t nat -A POSTROUTING -s 10.8.0.0/24 -o eth0 -j SNAT --to-source Pi IP address

Lastly, we had to give our client its key and certificates that was built previously and we configured the settings for the clients OpenVPN using the following commands. The following files (ca.crt, client1.crt, client1.key, and vpnsettings.ovpn) needed to be placed in our clients “C:\Program Files\OpenVPN\config” directory folder.

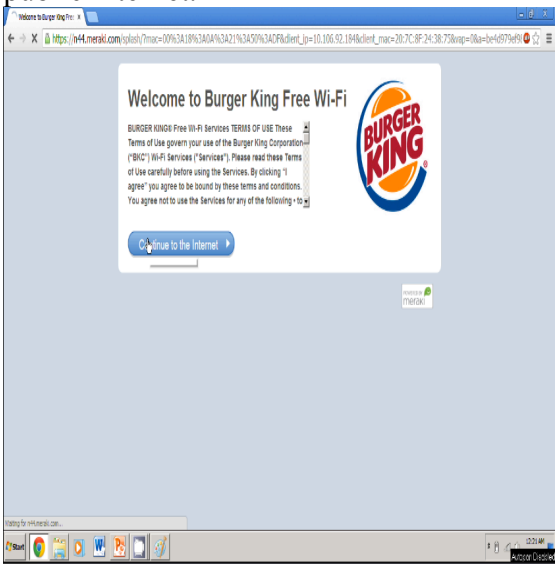
nano vpnsettings.ovpn
dev tun
client
proto udp
remote Pi’s IP address 1194
resolv-retry infinite
nobind
persist-key
persist-tun

ca ca.crt
cert client1.crt
key client1.key
comp-lzo
verb 3

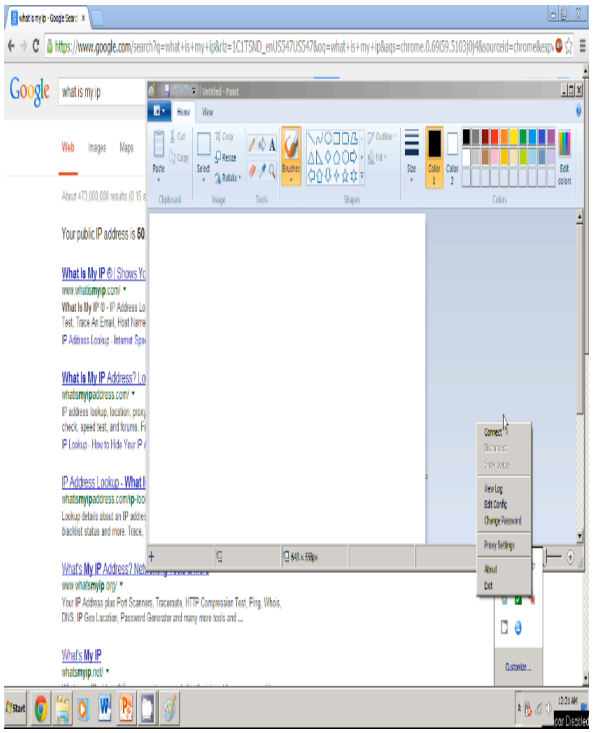
The guide we used to learn about these configurations for Openvpn are from references [4 - 5].

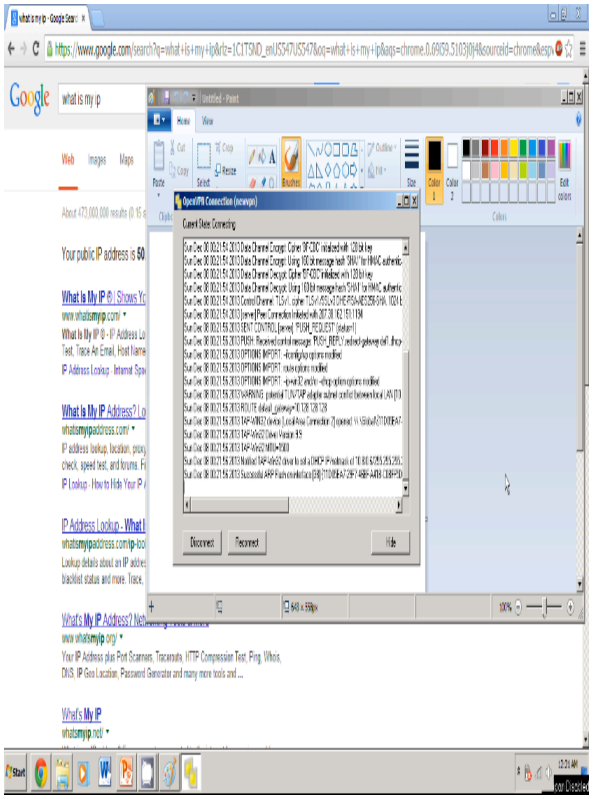
IV. Experimental Results

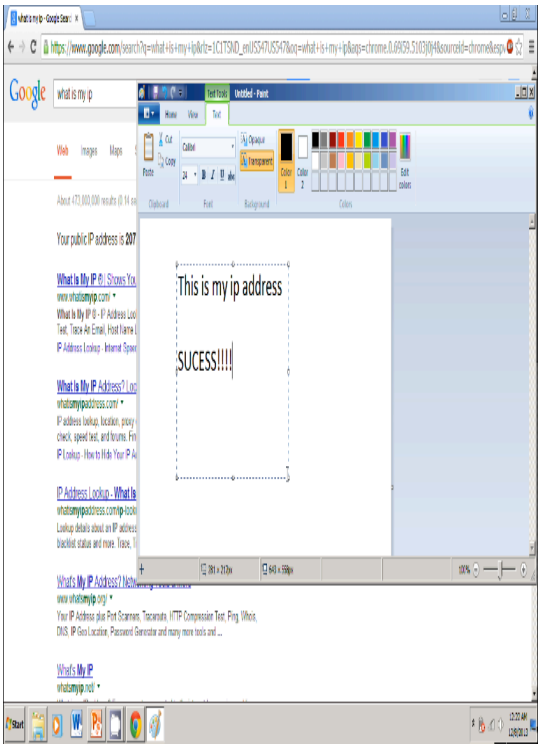
In order to test the VPN server we have created, we needed to leave the Pi in the router of the network we wanted to connect to (our home network) and travel outside looking hotspots that provide internet access. We brought our laptop with us which included all the client configurations files and the software OpenVPN. We brought our laptop to a local burger king and connected it to its public internet.



Afterwards, it was time to see if we successfully configure our VPN server. We did this by running OpenVPN as administrator and pressing connect.







As one can see, We were successful connecting to our home network from burger king through a full tunnel connection using the SSL/TLS protocol. This resolves the security issues one has when connecting to a public internet and we are now able to roam the web without any regulations. Moreover, We can access our Pi or any servers set up on our home network anywhere we go.

V. Conclusions

What makes the Raspberry Pi so special is its size and pricing. Raspberry Pi has given us the opportunity to create $35 value projects that are worth way more than its cost. We were inspired to create a VPN network when we found out how China was using the Pi to bypass its great firewall [6]. A tech-savvy Chinese man used VPN to connect to one of the many foreign VPN providers and was able to escape the government's censorship. On top of that, he was able to use his Pi as a hotspot giving more people access to this freedom. Using this idea, a person could also use the Pi's VPN server to enter foreign websites from another country. For example, from the US we are unable to enter websites for the UK, but if we were connected to a network in the UK we will be able to enter these website. To do this, one will need to obtain the IP from a service provider. Usually these services cost money, but we were able to find a website that provides an IP addresses from the UK for free [7].

References

[1] Upton, E.. "Raspberry Pi Faqs" N.p.. Web. 23 Nov 2013. *http://www.raspberrypi.org/faqs*

[2] Microsoft Technet, . "VPN Tunneling Protocols" N.p.. Web. 5 Dec 2013. *http://technet.microsoft.com/en-us/library/cc771298(v=WS.10).asp&xgt;*.

[3] Microsoft TechNet, . "How VPN Works" N.p.. Web. 22 Nov 2013. *http://technet.microsoft.com/en-us/library/cc779919(v=ws.10).asp&xgt;*.

[4] "Open vpn" N.p.. Web. 22 Nov 2013. *http://openvpn.net/index.php/open source/documentation/howto.html.*

[5] "Raspberry Pi Tutorials." Home. N.p., n.d. Web. 22 Nov. 2013.

[6] Muncaster, Phil. "Raspberry Pi Puts Holes in China's Great Firewall, The Register. N.p., 29 May 2013. Web. 12 Dec. 2013. *http://www.theregister.co.uk/2013/05/29/raspberry_pi_helps_hassle_free_circumvention_great_firewall/*

[7] VPNBOOK, . "Free openvpn and pptp" N.p.. Web. 15 Dec 2013. *http://www.vpnbook.com/*