**Project Report**

**Purpose:**

Built a multi-purpose Raspberry Pi Cluster that serves as a webserver to deploy an educational website for the client for teaching, as well as a Virtual Private Network (VPN) server, to safely connect into the local network and the local network devices from anywhere in the world.

**Requirements:**

* Website availability to the students of Saint Martin’s University.
* Website needs to include 13 weeks of study materials
* Easily understandable documentation
* Website needs to be easily navigable
* Professor needs access to the server remotely

**Organization:**

Team:

* Sanju Byanjankar
* Joshua Schulz

Stakeholder:

* Dr. Xuguang Chen

**Engineering:**

Research Specifications:

* C Programming, HTML/CSS
* Parallel Programming with MPI and OpenMP
* Linux scripting in Raspbian Operating System
* Networking with Raspberry Pi
* Virtual Network Computing
* Virtual Private Network

Tools and Resources to be used for design:

Hardware:

* 3 Raspberry Pi 3 Model B+
* 16GB SD card per Raspberry Pi
* Power supply
* 3 Layer raspberry pi case
* HDMI cables
* Ethernet cables
* 5-port Network Switch
* USB keyboard
* USB mouse
* Monitor or TV with HDMI output

Software:

* HTML/CSS languages are used for front-end software design
* Putty used for remote desktop connection in Local Area Network
* Virtual Network Computing (VNC) Connect is used for remotely connecting with servers.
* piVPN is installed on the server and OpenVPN is installed on the remote client, to create an encrypted connection between the remote device and the Local Area Network that the server is on.
* Haproxy is used as a load balancer

**Starting up Raspberry Pi Cluster:**

1. Downloading the Raspberry pi Operating system

* Browse <https://www.raspberrypi.org/downloads/>
* Download the Latest version of NOOBS on to your computer

1. Installing the operating system on a micro SD card

* Before installing the Operating system, we need to format the SD card first to use FAT32 filesystem.
* Insert SD card into SD card adapter and plug it into the computer.
* Browse <https://www.sdcard.org/downloads/formatter/> . Download and install a free software for SD card formatting compatible with your device.
* Run the SD card formatter software, and choose the correct drive to be formatted. I repeat. Be careful and choose the right drive. Do a complete format.
* Locate the NOOBS zip file you downloaded, right click and extract it to the SD card. Make sure it’s not a single folder. If it is then, copy and paste all the files and folders from the NOOBS folder into the SD card.

1. Starting it up

* Plug in all the SD cards into each of the raspberry pis.
* Power up one of your raspberry pis. Plug in keyboard and mouse into the USB port, and plug in the monitor via HDMI cable. We have to configure each of the raspberry pis individually since we probably have only one keyboard, mouse and monitor. Feel free to do it all at once if you have enough equipment.
* I used 5 port Network Switch to connect to the internet, which is connected to my router on the other side. You can directly connect your raspberry pis to the router using network cable.
* There is no on and off power button on a raspberry pi, therefore as soon as you plug in the main power, it starts booting.
* Check to see if the pi has internet access by opening up the browser and going to your favorite website. From terminal, you can try “curl <http://www.google.com>
* Configuring a Raspberry pi is very easy. Make sure to change the Pi name and password for security reasons.

**Setting up Raspberry Pi Web Server:**

Configuring each raspberry pi to have a static IP address is a necessity for it to work as a web server. When a user browses a website, your router should be able to direct the traffic to the correct server. For this reason, the IP address has to be static, and dynamic IP address may cause issues with browsing the web page. In short, assigning static IP addresses to your raspberry pis means that you’ll always know exactly where to find them on your network.

First of all, make sure you have the latest version of Raspbian. You can do that by opening your terminal and typing “cat /etc/network/interfaces”. You’ see the following output.

# interfaces(5) file used by ifup(8) and ifdown(8)

# Please note that this file is written to be used with dhcpcd

# For static IP, consult /etc/dhcpcd.conf and 'man dhcpcd.conf'

# Include files from /etc/network/interfaces.d:

source-directory /etc/network/interfaces.d

auto lo

iface lo inet loopback

iface eth0 inet manual

allow-hotplug wlan0

iface wlan0 inet manual

wpa-conf /etc/wpa\_supplicant/wpa\_supplicant.conf

allow-hotplug wlan1

iface wlan1 inet manual

wpa-conf /etc/wpa\_supplicant/wpa\_supplicant.conf

As long as “iface etho0 inet manual” is there, you are good to go. If not upgrade to a newer version of Raspbian. Now, to configure static ip addresses we need to get the ip addresses of the raspberry pis from the network.

In terminal, type “netstat -nr” and write down the Gateway address. In my case, my gateway address or router’s address is 192.168.0.1. Use your favorite text editor to open the dhcpcd.conf file. I used nano which is very simple and it does come pre-installed in raspberry pi, but you also can use vim, Gedit and many more. Type “nano /etc/dhcpcd.conf” to open up the file and add the following lines at the bottom of the file. I used 192.168.0.200 for the first pi, 192.168.0.201 for the second and 192.168.0.202 for the third, so that it is very easy to remember.

interface eth0

static ip\_address=192.168.0.200

static routers=192.168.0.1

static domain\_name\_servers=192.168.0.254

Restart the pi by unplugging and replugging the main power or the best way is to go to the terminal and type “sudo reboot”.

If everything went well then you should be able to ssh into your pis from any computers on the same network. To verify, try “ssh pi@192.168.0.200 and it should prompt you to enter password for your raspberry pi. Now the question is, how to make secure shell connection to the raspberry pis without having to type in the password? We can accomplish this task by generating using rsa keys.

Steps to secure shell from one raspberry pi to another using rsa keys.

* In terminal, make sure you are in root directory. Type “ssh-keygen” to generate a new key and save in the default “.ssh” folder with no password.
* Jump into that directory by using “cd .ssh” command and type “cp id\_rsa.pub pi01” to copy the id\_rsa.pub into a new file “pi01”. This will keep things organized.
* Now, ssh into the second raspberry pi, i.e. 192.168.201, and do the same thing. Generate a key and save it in a new file pi02. While on /.ssh directory, type “scp 192.168.0.200:/home/pi/.ssh/pi01” to copy pi01 file where the key is saved in first raspberry pi into the raspberry pi you currently in i.e. 192.168.0.201. Add pi01 to the authorized keys list. Type “cat pi01>>authorized\_keys. Exit out of current raspberry pi and switch into the first raspberry pi.
* At this point ssh into the third pi i.e. 192.168.0.201 and repeat the steps from the beginning.
* Final step is to jump to the first pi and copy files pi02 and pi03 into .ssh folder of the current pi. Use “scp 192.168.0.201:/home/pi/.ssh/pi02” and “scp 192.168.0.201:/home/pi/.ssh/pi03” to copy. Use “cat pi02>>authorized\_keys” and “cat pi03>>authorized\_keys” to add them into the authorized keys list of the first raspberry pi.

If you did everything correctly up till here, you should be able to ssh between the pis without any authentication using “ssh pi@192.168.0.201” command.

**Start a web server on Raspberry Pi.**

Create an html file using nano editor and give it a name of your choice. Save it with an extension .html, so that the server knows it’s a web file. I created a home page named index.html which is saved in the root folder of the pi. All the links from inside the home page is also saved in the same directory. For example: I have files named, chapter1.html, chapter2.html and so on.

Raspberry pi has very limited space. Therefore, next we are going to get a simple node server and make it run in all our raspberry pis. You can go to  <http://howtonode.org/hello-node> and copy the “hello world” example and custom configure it. I did it in the following way.

var http = require('http');

var express = require('express');

var app = express();

//express pointed to the root directory of the pi

app.use(express.static('/home/pi'));

var server = http.createServer(app);

//all raspberry pi servers must be listening to port 8000

server.listen(8000);

app.get('/', function (req, res) {

//making sure the server knows the html file format.

res.header('Content-type', 'text/html');

res.sendFile('index.html');

});

console.log("Server running at http://127.0.0.1:8000/");

Node webserver doesn’t have similar features as apache web server, and therefore I had to download “express” package to navigate through the files in the root directory.

Use nano text editor and copy the code into a file named basic\_node\_webserver.js

Type “nano basic\_node\_webserver.js” and paste it into the editor and save it. After that, starting uup is very easy. To run the node server use the command “node basic\_node\_server.js” and you should see the following.

$ node basic\_node\_webserver.js

Server running at http://127.0.0.1:8000/

Repeat the same for all other pis. Make sure they are running well before moving further.

You should be able to browse the server from all other devices that are on the same network. Go to your favorite browser, and browse “192.168.0.200:8000”, “192.168.0.201:8000”., “192.168.0.203:8000”. In that case, you should be able to open up the index.html file located in the root directory of the pi. In my case, all the files are just on the root directory. You can create a folder and put all the files in one for more organized work. Remember, all the files need to be on the same directory for easy navigation. This navigation is helped by the package we downloaded called express, which basically looks for files in the same directory on a user’s command. So, all three pis are responding to the requests to port 8000.

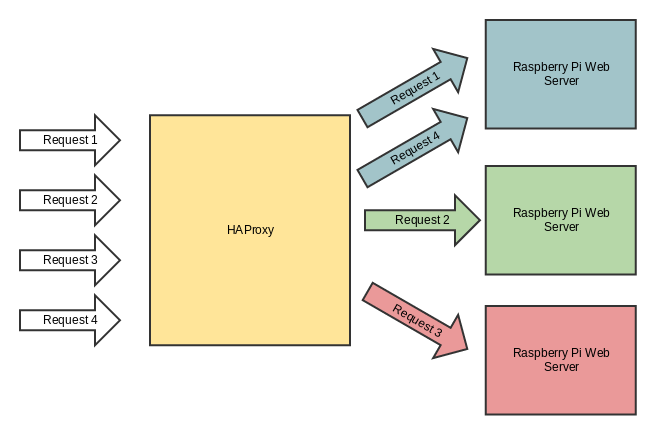
**Tying up all the raspberry pis with HAProxy:**

HAProxy stands for High Availability Proxy which is developed by HAProxy Technologies LLC. It is an open source load balancer proxy for TCP and HTTP applications.

Now, *why do we need raspberry pi cluster and haproxy necessary for website deployment?*

This is a legit question to ask. Why do we need multiple raspberry pis when a single raspberry pi is capable of listening and responding to the requests? Well, if this raspberry pi was actually a computer serving requests on the internet for your website, when your websites get popular you will encounter tremendous amount of traffic on your server and may not be able to handle. Therefore, it is better to have multiple computers serving the requests instead of one. It just like a division of labor in a certain company.

Adding more computers or raspberry pis to the network can improve or minimize the downtime of the website. Even if one or two servers goes down the other servers can still handle the requests. Also, HAProxy improves the performance of servers by distributing their workloads to multiple raspberry pis by minimizing response times and increasing throughput.



We will be using HAProxy to load balance incoming requests to a single IP address (website address), so that each request can go to one of the several Raspberry pi’s behind the router. Next step is to download and install HAProxy into one of our Raspberry Pi

Use command “sudo apt-get install haproxy” to download and install.

Use command “node basic\_node\_webserver.js” in all the raspberry pis.

Check to see if all the servers are running by browsing the ip addresses from another device on the same network. Don’t for get to include the port number 8000 after the IP addresses.

Configuring HAProxy:

You can choose any of the raspberry pis to install the HAProxy or even choose any computer on the same network at home. Follow the steps below to configure properly.

* Go to the raspberry pi with HAProxy installed
* We are going to edit the config file. So, let’s use nano editor to open haproxy config file.

Type “sudo nano /etc/haproxy/haproxy.cfg”

* Let’s edit the front-end and the back-end portion of the code.

frontend raspberrypinodes

bind \*:80

mode http

default\_backend nodes

backend nodes

mode http

balance leastconn

option forwardfor

http-request set-header X-Forwarded-Port %[dst\_port]

http-request add-header X-Forwarded-Proto https if { ssl\_fc }

option httpchk HEAD / HTTP/1.1\r\nHost:localhost

server raspberrypi01 192.168.0.200:8000 check

server raspberrypi02 192.168.0.201:8000 check

server raspberrypi03 192.168.0.202:8000 check

listen stats

bind \*:1936

stats enable

stats uri /

stats hide-version

stats auth someuser:password4321

Finally, it’s time to see it in action. Save the updates in the config file and get onto the pi with HAProxy installed. Type “sudo service haproxy restart”.

Browse the IP address of the raspberry pi in your favorite browser from a different device in same network. Notice that it doesn’t show the port 8000 anymore. That is a default for port 80, which basically means that HAProxy is receiving request on port 80 and proxies it to port 8000 on one of the raspberry pis that is listening to port 8000.

If you try to refresh the page, it switches between the servers and that is because HAProxy is load balancing between the raspberry pis. In the back-end portion, instead of “leastconn” if I use “roundrobin”, the servers will basically take turns to respond to the request. The reason I chose “leastconn” is because it helps us to choose the pi with the least connection which will better optimize connections.

If you reached here and accomplished every thing till now then you deserve a Pie. We have built a Raspberry Pi Cluster Webserver. Staying focused again.

This is how it works in the large scale. When you browse a popular website, it hits the central web address and your request gets routed to one of the many servers.

The last portion in the config file is “listen stats” and is a stats tracker. It is optional but since we added it in the config file, we should be able to go to the port we specified and see the stats on each of our servers.

**Domain Name and Domain Name Server**

“www.Google.com”, “www.Amazon.com.us”, “www.stmartin.edu”, etc. are some examples of the domain names. Domain names are used to name the server’s IP addresses so that you do not have to remember the numbers to browse them on internet. It is easy to remember [www.Google.com](http://www.Google.com) than it is to remember an IP address. Therefore, I have created a free account in Godaddy.com to get a free subscription of domain name for a year.

Steps to get a domain name and make your website browsable using it:

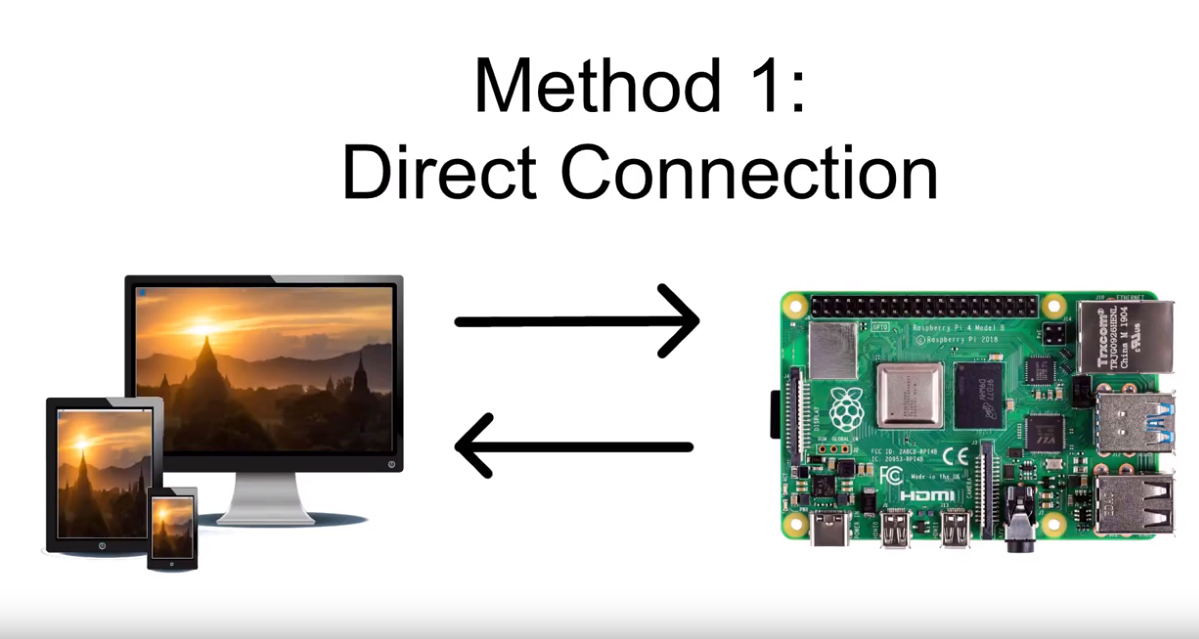
* Go to www.godaddy.com page and sign up for your free account.
* It will ask you some basic information like name, address and email address.
* As soon as you create an account you will be asked to verify the email address by clicking the link emailed to you.
* Now, go to the home page. You will see a search box where you can type the name you want in your domain name. I typed in “benzan”, and it showed me all the available lists of domain names and their prices.
* Your goal is to find some kind of deal like for instance, the first year free, $1 for two years subscription etc.
* After you find the domain name you wanted, click “Add to cart” button to proceed to checkout.
* Here now you have a domain name. In my case, it is [www.benzanusa.com](http://www.benzanusa.com)
* Make sure to go to the homepage. On the top right corner of the page find your name and click it. From the drop-down list, choose My Products; which is listed below your customer number.
* Next, find Domains and click to see your domain name you just purchased.
* To the right end of the domain name, click on the DNS which will open DNS Management.
* In DNS Management we will be editing records to point to our home server’s address.
* Before we edit anything in this page, it is important to get the public IP for your home network, which is the address of your router assigned by your service provider for the Wide Area Network. Remember, it is totally different than the LAN IP address 192.168.0.1 that we used for router.
* To get your public IP address you have to be in your home or Local Area Network that your servers are in and browse whatsmyip.com.
* This will provide you with the public IP address of your network and this is the IP address we are going to use to configure the record in DNS Management.
* Go to the DNS Management page where we left and click on the pencil icon to the right of the Type A inside the Record.
* Leave Host as it is, type your public IP address we just retrieved into the box where it says “Points to” and TTL (Time To Live) to 1 hour.
* Save it and check the name servers.
* If you do not have your native name servers use the default name servers provided by godaddy.com and it will make your life easier.
* Go to your favorite browser and type your domain name, in my case it is benzanusa.com. Assuming that you have turned on the HAProxy and node servers in the raspberry pis, it will populate your website from the servers.

**Virtual Network Computing (VNC)**

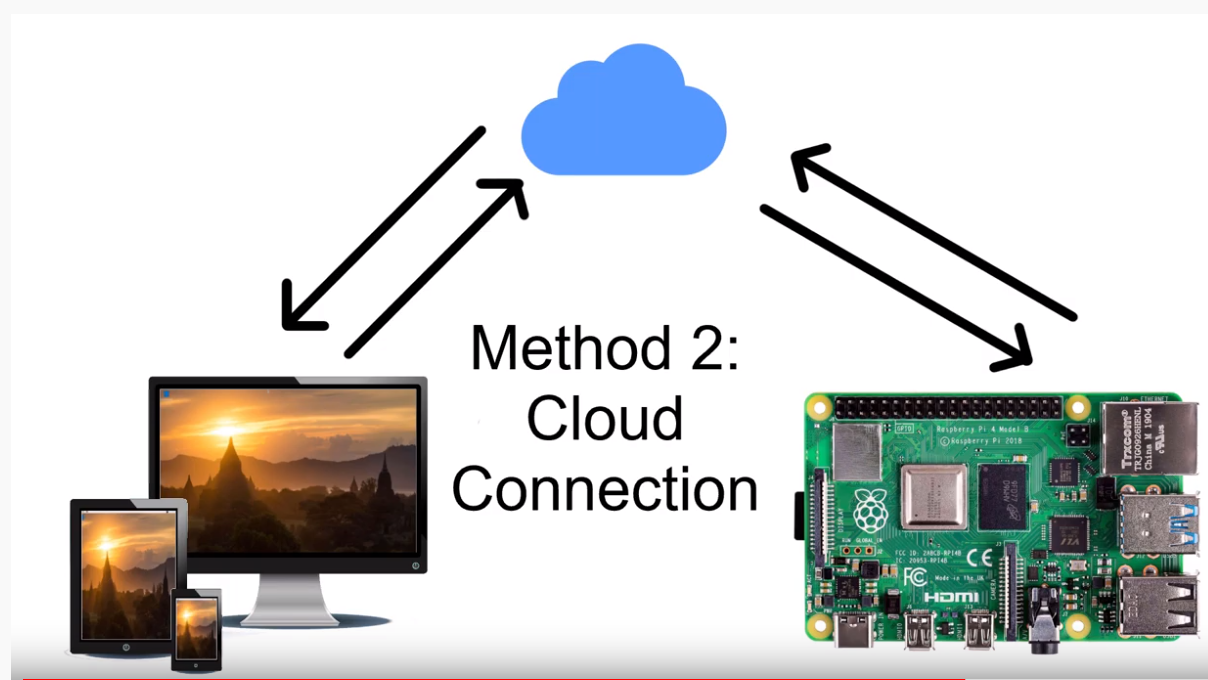
VNC was originally developed at the [Olivetti & Oracle Research Lab](https://en.wikipedia.org/wiki/Olivetti_%26_Oracle_Research_Lab) in Cambridge, United Kingdom. VNC is a graphical desktop-sharing system that uses the Remote Frame Buffer protocol (RFB) to remotely control another computer. It transmits the keyboard and mouse events from one computer to another, relaying the graphical-screen updates back in the other direction, over a network. Multiple clients may connect to the VNC server at the same time.

There are two different method to connect to the raspberry pi remotely.

* Direct Connection
* This is effective when you are on a same network as that of your raspberry pi. It is similar to putty application in windows where you can secure shell into a device on the local area network. What if you want to access your raspberry pi form anywhere in the world? That’s exactly when the cloud connection comes in.



* Cloud Connection
* As you can tell by its name, the whole connection is via cloud. We actually connect two end nodes via an account we create in realvnc.com. This connection will give us access to our raspberry pi from anywhere in the world.



Let’s go step by step how to make this connection work:

* Secure shell into your raspberry pi and update and upgrade it into a newest version of Raspbian. Use: sudo apt-get update and sudo apt-get upgrade. Do this every time you want to install something on the raspberry pi.
* The for raspberry pi we need to install VNC server not VNC viewer. Viewer is for the clients so I did not install it in the raspberry pi. Type: sudo apt-get install -y realvnc-vnc-server.
* Next, go to Preferences > Raspberry Pi Configuration >Interfaces and Enable VNC.
* Now you should see a VNC logo on the top right corner of the raspberry pi.
* Leave the raspberry pi alone and jump into you client computer, and install VNC viewer. Go to realvnc.com and choose the compatible viewer for your machine.
* Now comes the fun part. We will now create an account in Real VNC.
* After you gain a username and password, now that is the key to the connection.
* Go back to your raspberry pi and click on the VNC logo on the top right.
* Click on Menu > Licensing.
* Login with the same username and password for your Real VNC account.
* Go back to the client computer and do the same thing. Login to your VNC Viewer, and login using the same username and password.
* Now, you should be able to see your computer on the viewer. When you click on it will ask your raspberry pi’s username and password. Connection is created between the two.
* VNC viewer is even available in apple store and google play, which is even more handy and convenient.
* Direct connection is simple. Get your raspberry pi’s IP address and type it on the VNC viewer search box. Make sure you are on the local area network as that of raspberry pi. You will be prompted to enter username and password. Simply login.

Virtual Network Computing is very easy to install and use. There is no port forwarding involved which is why I chose this in the first place. I was using window’s remote desktop connection but it requires me to type in my public IP address, which is not very secure. My two ends are connected via cloud and is very secured.

**Virtual Private Network (VPN)**

VPN sounds almost like VNC, but don’t be fooled. These are totally different tools. For instance, VPN is the tool you can use to be in two different places at the same time. VPN extends a private network (or Local Area Network) across a public network, and enables users to send and receive data across shared or public networks as if their computing devices were directly connected to the private network. Therefore, we can still be secure and use the secure, trusted connection to get jobs done even on the unsecure connection. What that means is, VPN creates an end-to end encrypted tunnel between two nodes to provide the safest connection.

Installing VPN has multiple step and once again we are going step by step.

* First stop is going to be on www.pivpn.io website. You will see the instructions on how to install and get VPN running in your raspberry pi.
* Use the curl command provided on the website to install pivpn on the raspberry pi. I am installing VPN server on my second raspberry pi i.e. 192.168.0.201. So, get on the terminal and type the following.
* curl -L https://install.pivpn.io | bash
* After the process finishes, hit OK on the PiVPN Automated Installer.
* On the Static IP Needed step, it will remind you that this process requires a static IP address, which we already have. If you don’t have it done yet, jump back to the previous notes of my raspberry pi configuration notes. Hit OK.
* On the static IP address window, note down the IP addresses of your router and the raspberry pi you are working on. We will need this while we port forward. Hit Yes.

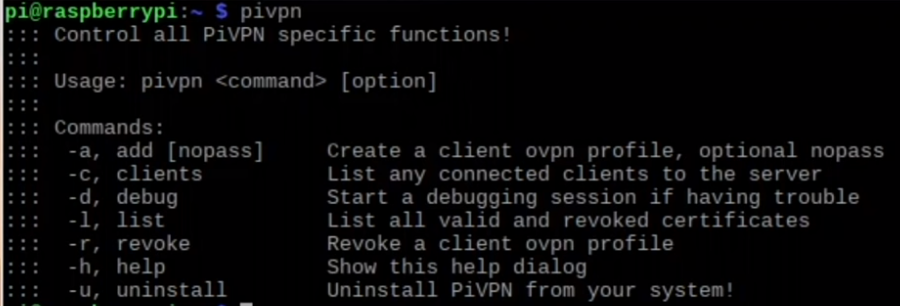
Mine is: IP address : 192.168.0.201

Gateway : 192.168.0.1

* Hit OK ono IP conflict window.
* Choose the user pi, that’ll whatever you named your raspberry pi.
* Next window Hit Yes again for the automatic upgrades for minimal security breaches.
* I used default protocol i.e. UDP on the next window.
* Next step is very important and is about assigning the port. It’s like the main entrance to your home or might as well call it the port of entry. 1194 is the default port and I used the same. Keep note of that number as we are going to need it for port forwarding.
* Used 256-bit encryption and you can make your own choice.
* On the Public IP or DNS page I choose public IP to connect. You can get your public IP by going into your favorite browser and typing Whatsmyip.org. Then, hit OK.
* Choose Google DNS because it is the default. Again, you can choose a DNS for your client and hit OK.
* Reboot the raspberry pi.
* Now PiVPN is successfully installed.

Adding user to the VPN server

* Get to the terminal and type pivpn.
* It will get you some help on what you can do with that command. You will see the following.



* Now as you can see, to add the user we have to use “-a”.

Type: pivpn -a and it prompt you to choose a client name and password of your wish

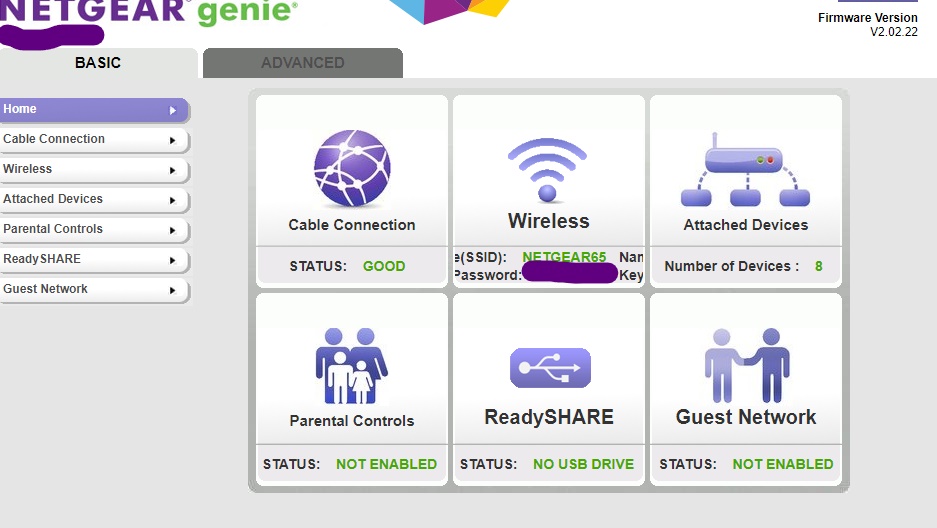
* Now, doing this you have created a “.ovpn file” in the ovpns folder. My user name was sanju, and therefore it created a file named “sanju.ovpn” (which is actually a certificate we are going to use to create end-to-end encryption) in “home/pi/ovpns” directory.
* “sanju.ovpn” is like the key, and it is not recommended to send it to your client devices via email because then, there will be no point of creating an encrypted connection. You just sent your key to the un-encrypted email which is a security risk.
* Easiest and safest way is to copy it to a flash drive or any other private storage device and transfer it manually to the client machines. That’s exactly what I did.
* Now, we will download the client version of VPN which is called OpenVPN.
* Go to openvpn.net and download and install the latest OpenVPN compatible to your machine.
* After the installation is done, you will have an icon on the desktop called OpenVPN GUI.
* Click on that and now you will get an error because no certificates has been imported to initialize it. Click OK.
* Now, have your external drive with the “.ovpn” file ready. Connect it to your client computer.
* Get on to the lower right taskbar where you can see a OpenVPN GUI icon.
* Right click on it and click Import file. Choose your external drive to choose the file from and double click on the “.ovpn” file. Now we are ready for port forwarding.
* Follow the port forwarding portion on the following topic.

**Port Forwarding**

Port forwarding is also known as port mapping. It is an application of network address translation (NAT) that redirects a communication request from one address and port number combination to another while the packets are traversing a network gateway, such as a router or firewall. Without port forwarding, your firewall will not allow any traffic in, therefore you have to configure your router to let traffic in at port 80. On the inside network you already have HAProxy listening to port 80 and will route the traffic to the servers.

Port Forwarding to HAProxy on Server: 192.168.0.200

* First of all, make sure you are in your LAN.
* Then, in your favorite browser, type in your router’s IP Address. Usually, it is 192.168.0.1, if you haven’t changed it. In my case, it is the default (192.168.0.1) assigned by my network provider.
* You will be then prompted for your username and password.
* After the authentication process, you will be directed to the router’s interface.

****

My router interface is shown in the above picture which is of NETGEAR genie. Yours might be different interface depending on the brand of the router you use.

* Go to Advanced >Advanced Setup>Port Forwarding.
* Click Add Customer Service button to add a port.
* Type the name you want the connection to be represented as.
* Leave the Service Type to TCP/UDP.
* Set the External Starting Port and External Ending Port to 80.
* Type your server IP address in Internal IP address box. In my case, it is 192.168.0.200 where my HAProxy is listening to port 80.
* Click Apply and you are all set.

Port Forwarding for Virtual Private Network (VPN) on Server: 192.168.0.201

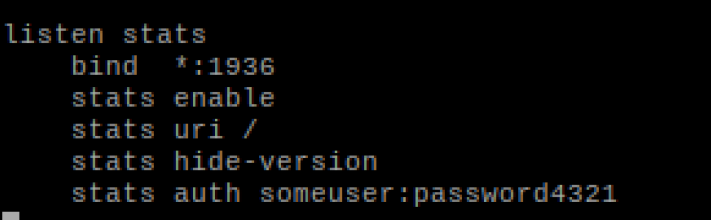
* We follow the same steps until we click Add Customer Service button to add a port.
* Now type the Service name of your choice.
* Leave Service type to TCP/UDP.
* Remember the port number we noted while we were configuring pivpn, i.e. 1194. Type that port number in both External Starting Port and External Ending Port.
* Next, Internal IP address is going to be the one you wrote down while configuring the pivpn. In my case, it is 192.168.0.201.
* Click Apply and here we are done.
* If every thing went well up until now, your VPN is ready to use.
* To check the VPN, make sure the client computer is not on the LAN. I use my phone’s hotspot to be in the different network. Make sure your phone is not connected to the Wi-Fi either, if you are going to use the hotspot.
* Now, let’s login to the VPN. Go to the tiny icon on the bottom left of the taskbar and right click on it.
* Click connect and enter the password that we set up while we added the user pivpn on our raspberry pi. Click OK.
* To make sure you are connected to the private network at home, go to your favorite browser and search 192.168.0.1 and see what happens.
* Exactly! It takes you to the authentication page for your router. This only happens when you are on your Local Area Network.

This completes our process to build a virtual private network. Now, you can connect to all your home devices from anywhere in the world and that’s awesome.

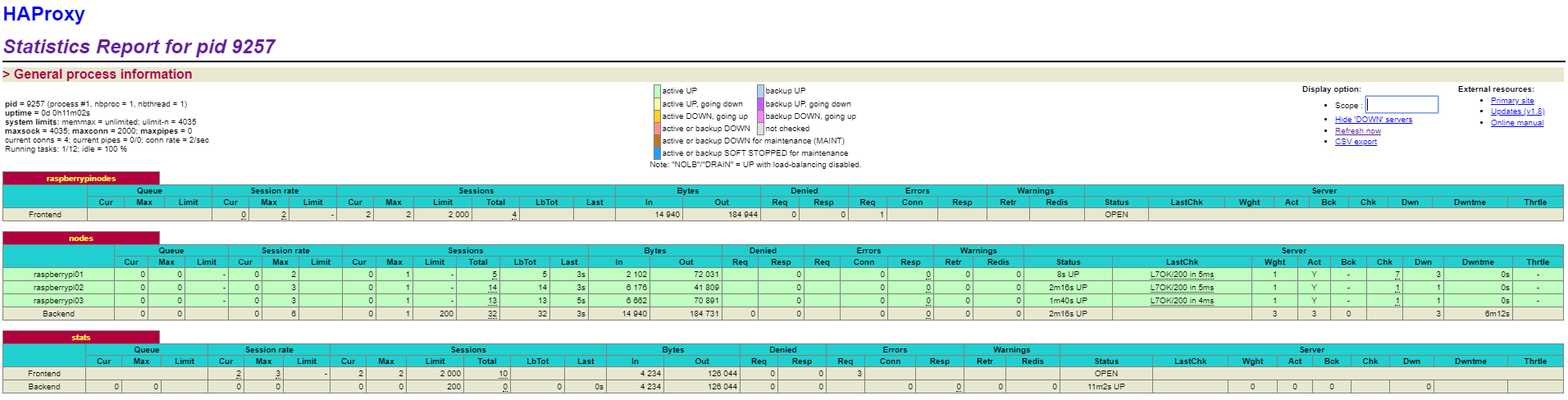
**Monitoring and controls**

* For monitoring the traffic, error and queues on the server, I configured the HAProxy to listen at port 1936 for the stats which will provide us the statistics on the server’s status in real time. This will give us on idea, if we need to scale up in future.

I added a simple block of text to the haproxy.cfg file

****

* Here my username: someuser, and password:password4321 comes in handy when I try to get through the port number 1936. It will take me to the authentication page and on successful identification, it will allow me to view the status of my servers. I have attached a clip of the page below.

****

* Servers need constant updates and upgrades for the latest operating systems and software to minimize the security breach and optimal traffic speed. Therefore, updates and upgrades will be scheduled every 3 months.
* VNC is used to control the servers anytime and from anywhere in the world. It is very easy to turn on and off the servers using VNC and I can even do it form my smart phone.
* VPN will give you extra slack to connect to any needed connections to your home devices.

**Next Steps**

* Build a client-side software, through which a client can easily create and modify the texts on the websites. This will save them from diving into the actual code and modify.
* Linux has no limit. We keep moving forward and try out new technologies using this simple device.
* Next thing in my mind is use raspberry pi to control the Energenie wireless sockets to automate my home devices and make them work for me in a controlled loop. VPN I just built will help me a lot to complete this task.
* Raspberry Pi can also be used as a caching server which will make your internet comparatively faster for the same amount of money you’ve been paying your network provider.

**Milestones:**



**Budget:**

* Saint Martin’s University allocated $150.00 of the budget to support the project, “Pi in the Sky”.
* 3 Raspberry Pi 3 Model B+
* 16GB SD card per Raspberry Pi
* Power supply3 Layer raspberry pi case
* HDMI cables
* Ethernet cables
* 5-port Network Switch
* Project owner’s expense
* USB keyboard - $30.00
* USB mouse - $25.00
* Monitor or TV with HDMI output - $105.00

**Risks:**

* There is always a security risk. These risks have made cyber security a vital organ of the tech world.
* Server might not be able to handle excessive traffic because of the low RAM on each of the servers.

Process for managing risks:

* Paying attention to simple things will eliminate most of the security risks. For example, custom creating passwords and usernames, creating a password easy to remember but hard to guess, making sure the software is up-to-date, making sure the port forwarding is done properly and many more.
* Since, this project is not the commercial version, it cannot handle too much traffic. Raspberry Pi 3 Model B+ has a RAM of 1GB and is completely embedded into the chip. It’s impossible to increase the RAM size in Raspberry Pi but it can be optimized using a software