Daniel Butts

CMI – Rogers Final Report

Breakdown of work

September 2020 – October 2020:

* A large portion of my time was spent getting familiar with the Geo-Studies team’s problems and goals. I looked into the resources provided by Mitch to get familiar with how the ciphering of NAS messages are handled and how his team was attempting to decipher them.
* I did some minor investigations into trying to find more efficient methods to estimate the NAS count, but nothing came from it.

November 2020

* During the month of November, I focused more on getting myself familiar with 4G architecture and the different procedures that are carried out over the network.
* I then begun to develop flow charts that may be used to help determine the state of the devices during these different procedures (connection, release, location update, etc.). More work needs to be done to finish these.

December 2020

* Seeing as Rogers was unable to provide us with network data by this point, I begun familiarizing myself with Python libraries such as PyShark and Scapy to process PCAP files so that pertinent information may be stored as CSV files so that it can be used to train any future machine learning models.

January 2021

* Explored the different fields that existed in the different protocols used by 4G to determine which fields may be of interest.
* Continued to develop a prototype that would be able to parse PCAP files which contain 4G data into an appropriate CSV file.

February 2021

* Continued development of 4G PCAP-CSV script. I focused on how to map a current user to different network protocols using the corresponding tunnel ids.

March 2021

* Explored the paper mentioned by Khaled, set up the specified network, fixed the issues found in code, as well as explored D-ITG for traffic generation.

April 2021

* Determined that the work specified above was not applicable to our work due to it’s focus on open-flow traffic classification.
* Found a library called NFStream, seems like a promising way to explore and analyze network flows. Built a small script that used an hourly cron job, that gathered network traffic every hour and then used the NFStream tool to perform statistical analysis on the data.

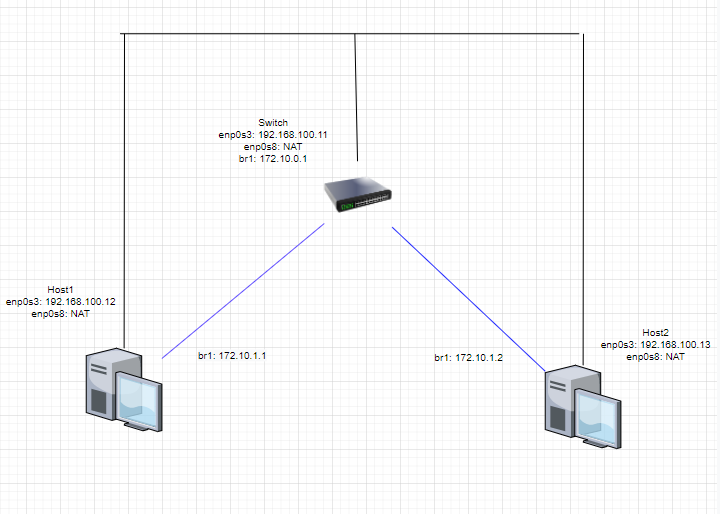
Environment Setup

I am currently using 3 Virtual Machines, 2 hosts to produce and receive network traffic, and 1 VM acting as a switch.

Specifications for each VM (note these are excessive, you don’t need as many resources allocated):

1. OS: Ubuntu 20.10
2. Each VM has 8GB of RAM
3. 1 CPU
4. 2 Network Adapters:
   1. NAT Adaptor
   2. Internal Network

This is what my network is currently looking like, to avoid needing to set up the network every time you turn your VM’s on, ensure that you update your netplan accordingly. Initially I had used Open vSwitch to forward flows to a controller, but it was determined to be unnecessary at this time.



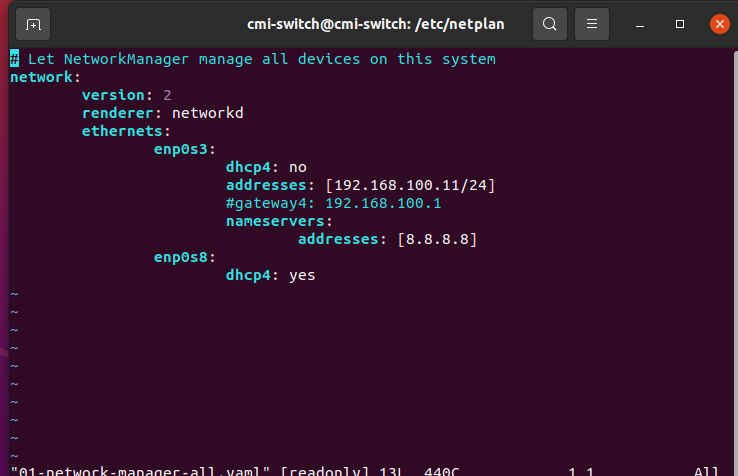


Figure 1 Example netplan for Switch

Open vSwitch Installation

1. Install Open vSwitch on all 3 machines
2. You only need to worry about setting up the VXLANs (shown in blue on the diagram).
   1. Run ovs-vsctl add-br br1 on all 3 VMs
   2. Perform ifconfig br1 [172…..] on all 3 VMs
   3. On the switch run:
      1. ovs-vsctl add-port br1 vx1 -- set interface vx1 type=vxlan options:remote\_ip=192.168.100.12
      2. ovs-vsctl add-port br1 vx2 -- set interface vx1 type=vxlan options:remote\_ip=192.168.100.13
   4. On both hosts run:
      1. ovs-vsctl add-port br1 vx1 -- set interface vx1 type=vxlan options:remote\_ip=192.168.100.11
3. To confirm that the vxlan’s are setup, try pinging one host from another using the VXLAN IP (172….)

After you confirm that the network is working properly, you need to install D-ITG from this link <http://traffic.comics.unina.it/software/ITG/>.

After installation, you will need to set up a receiver on one host, and a sender on another. To do so, move to your bin directory within the D-ITG directory, and run the command “./ITGRecv” on the receiver host. To send traffic from your sending host, you will need to do something similar, but the command will differ depending on the type of traffic that you want to send. For instance, to send DNS traffic, you could use the command “./ITGSend DNS -a 172.10.1.2 -t 30000” Refer to the documentation for more information.

Setting up the network gathering cronjob is straight forward. After ensuring that you have installed all necessary python libraries (in this case, you should only need to install nfstream and pyshark) and downloaded the appropriate modules from github, type “sudo crontab -e” and then add this one line at the bottom of the file (location of python file will depend on where you saved it):

