Final

The overall logic of my controller is represented via the guidelines set in this assignment. We will be connection hosts via switches to another while disallowing an untrusted host from pinging the rest of the lot. We can test the conclusiveness of what I'm attempting to create via the following commands. I Accomplished this by identifying the arp packets flooding them then identifying icmp packets, finding the src and dst ip addresses and if they were the ones I wanted id send them through if not id drop them. I did the same for tcp packets.

Pingall hypothesis-
We will be testing the connections through the utilization of ping a ll. This will check to see if by pinging the connections between hosts via the topology we created will work in accordance to the guideline that was presented. We will not drop packets that go to or from the untrusted host. With this in mind when we run ping all we will expect the untrusted host which I have named evil to drop all packets and look like xxxx. For the rest of the hosts we will expect them to successfully ping the rest however drop the packets that will be sent to the untrusted host. This will then look like x s2 s3 trust etc.

Iperf hypothesis - x x
Here we are testing the tcp bandwidth connection between a variety of the hosts. I will be inserting the base command as well as a variety of different variables to make sure all connections hold true. I will be expecting, regardless of the trusted or untrusted host that all tcp connections are secure and will result in bandwidth.

Ping all test results

```
mininet> pingall
*** Ping: testing ping reachability
evil -> X X X X
h1 -> X h2 h3 srvr
h2 -> X h1 h3 srvr
h3 -> X h1 h2 srvr
srvr -> X h1 h2 h3
*** Results: 40% dropped (12/20 received)
mininet>
```

Here we see what is to be expected and what I predicted would happen based of the code I created.
Srvr is the trusted and evil is the untrusted.

Iperf-
Here see we a variety of iperf tests

```
mininet> iperf
*** Iperf: testing TCP bandwidth between evil and srvr
*** Results: ['23.1 Gbits/sec', '23.1 Gbits/sec']
mininet> iperf h1 h2
*** Iperf: testing TCP bandwidth between h1 and h2
*** Results: ['14.1 Gbits/sec', '14.1 Gbits/sec']
mininet> iperf h1 evil
*** Iperf: testing TCP bandwidth between h1 and evil
*** Results: ['14.0 Gbits/sec', '14.0 Gbits/sec']
mininet> iperf evil h1
*** Iperf: testing TCP bandwidth between evil and h1
*** Results: ['14.5 Gbits/sec', '14.5 Gbits/sec']
mininet> iperf srvr evil
*** Iperf: testing TCP bandwidth between srvr and evil
*** Results: ['16.9 Gbits/sec', '16.9 Gbits/sec']
mininet>
```

We can see  the results are positive in their conclusiveness.

Dump ctl

-After we have done both these tests we can conclude by utilizing dump ctl to represent glimmers of the topology and connections henceforth.

```
mininet> dump ctl
<Host evil: evil-eth0:128.114.50.10 pid=4133>
<Host h1: h1-eth0:10.0.1.101 pid=4137>
<Host h2: h2-eth0:10.0.2.102 pid=4139>
<Host h3: h3-eth0:10.0.3.103 pid=4141>
<Host srvr: srvr-eth0:10.0.4.104 pid=4143>
<OVSSwitch s1: lo:127.0.0.1,s1-eth8:None,s1-eth9:None pid=4148>
<OVSSwitch s2: lo:127.0.0.1,s2-eth8:None,s2-eth9:None pid=4151>
<OVSSwitch s3: lo:127.0.0.1,s3-eth8:None,s3-eth9:None pid=4154>
<OVSSwitch s4: lo:127.0.0.1,s4-eth1:None,s4-eth3:None,s4-eth4:None,s4-eth5:None,
s4-eth9:None pid=4157>
<OVSSwitch s5: lo:127.0.0.1,s5-eth8:None,s5-eth9:None pid=4160>
<RemoteController c0: 127.0.0.1:6633 pid=4127>
mininet>
```