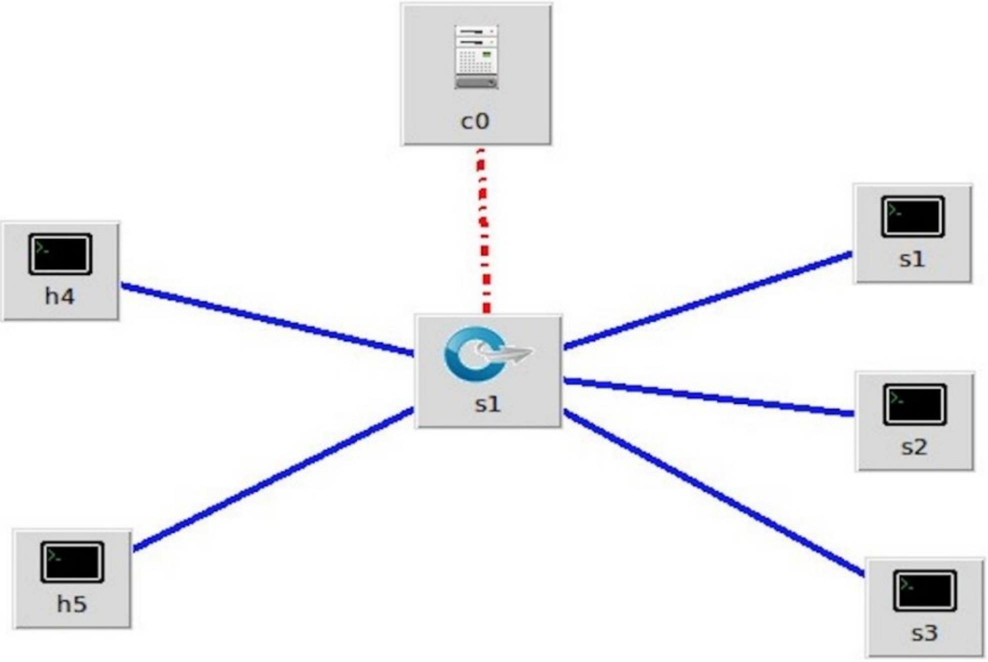
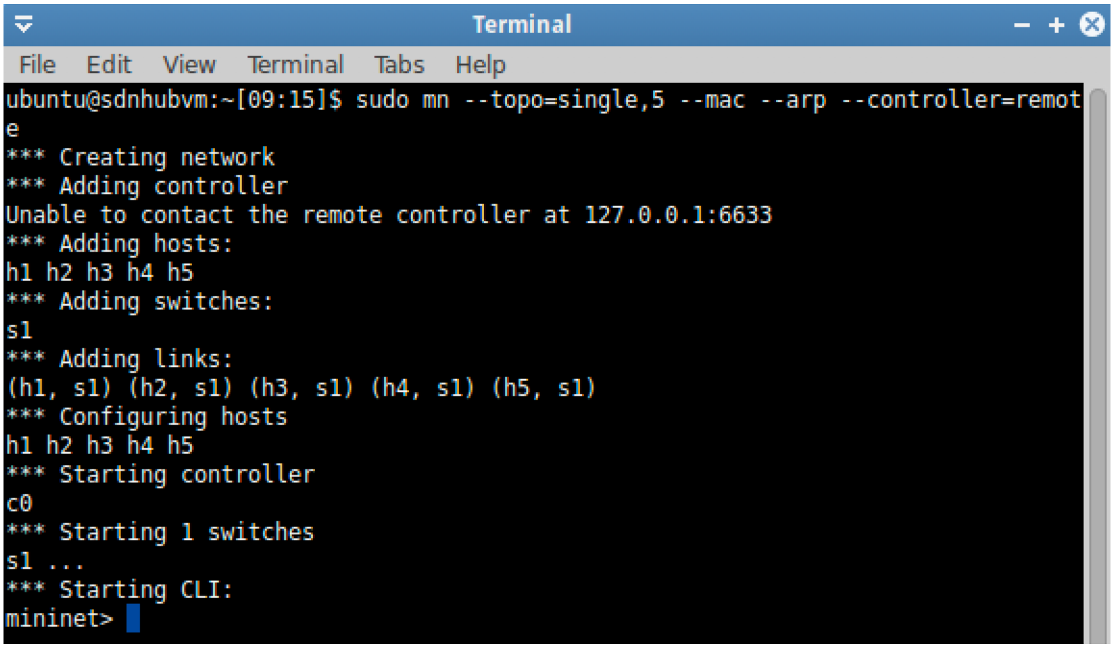
Flow Clustering (unsupervised learning) and Classification (supervised learning) for Load-Balanced SDN Controller Operation

Topology and Working



The following topology was created in Mininet using the command:
sudo mn —TOPO=SINGLE,5 —mac —arp —controller=remote
This command creates a topology with 5 hosts: 3 servers (h1, h2, H3) and 2 clients (h4, H5).



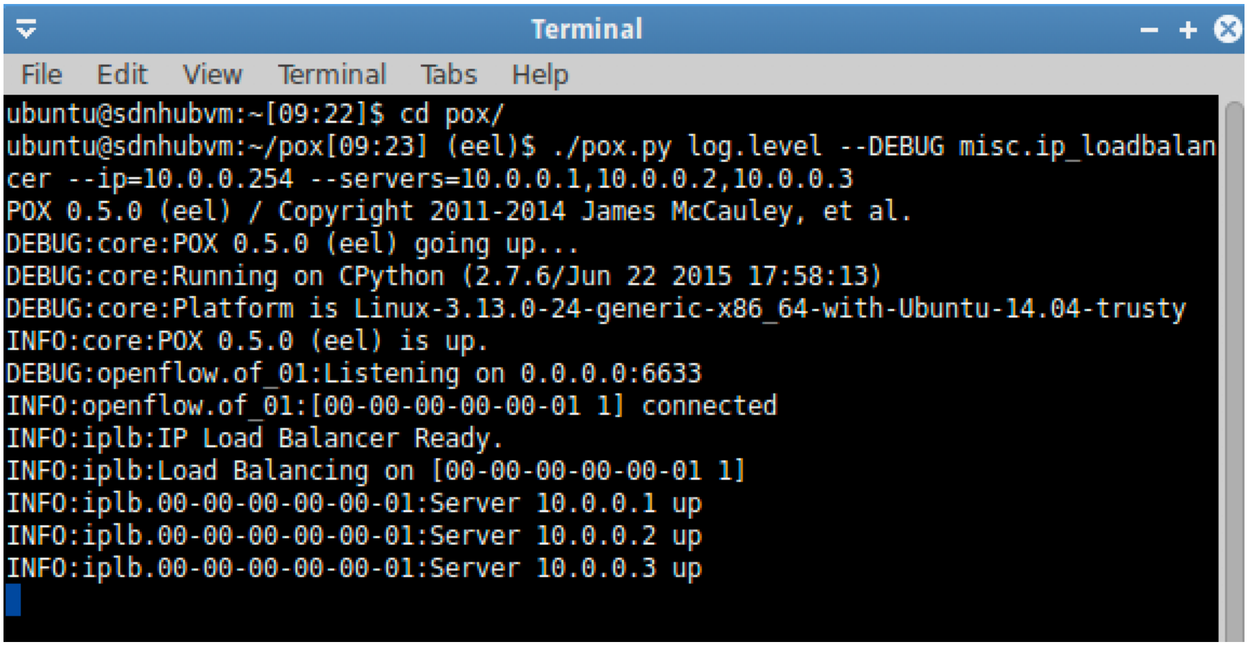
```
ubuntu@sdnhubvm:~[09:15]$ sudo mn --topo=single,5 --mac --arp --controller=remot
e
*** Creating network
*** Adding controller
Unable to contact the remote controller at 127.0.0.1:6633
*** Adding hosts:
h1 h2 h3 h4 h5
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1) (h3, s1) (h4, s1) (h5, s1)
*** Configuring hosts
h1 h2 h3 h4 h5
*** Starting controller
c0
*** Starting 1 switches
s1 ...
*** Starting CLI:
mininet>
```

POX controller is used to implement load balancing.

After this run the controller using the command:

```
./pox.py log.level –DEBUG misc.ip_loadbalancer –ip=10.0.0.254 –SERVERS=10.0.0.1,10.0.0.2,10.0.0.3
```

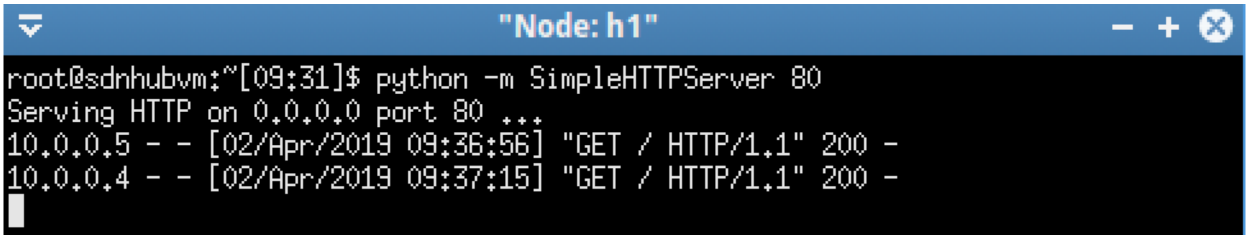
Here –IP=10.0.0.254 is the load balancer IP and –servers is the IP addresses of each of the servers.



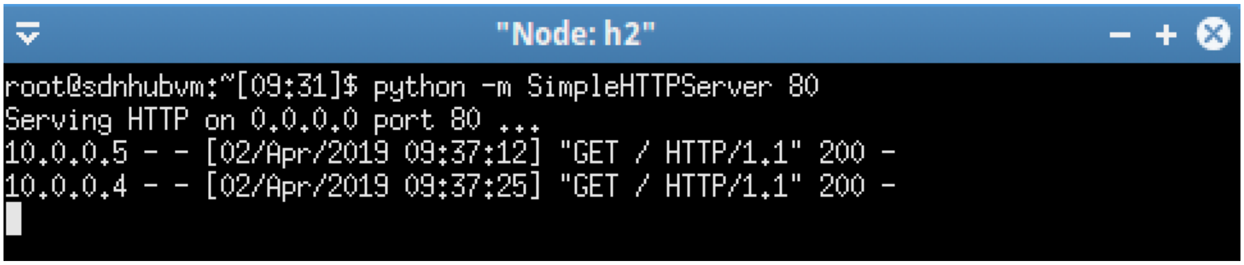
```
ubuntu@sdnhubvm:~[09:22]$ cd pox/
ubuntu@sdnhubvm:~/pox[09:23] (eel)$ ./pox.py log.level --DEBUG misc.ip_loadbalan
cer --ip=10.0.0.254 --servers=10.0.0.1,10.0.0.2,10.0.0.3
POX 0.5.0 (eel) / Copyright 2011-2014 James McCauley, et al.
DEBUG:core:POX 0.5.0 (eel) going up...
DEBUG:core:Running on CPython (2.7.6/Jun 22 2015 17:58:13)
DEBUG:core:Platform is Linux-3.13.0-24-generic-x86_64-with-Ubuntu-14.04-trusty
INFO:core:POX 0.5.0 (eel) is up.
DEBUG:openflow.of_01:Listening on 0.0.0.0:6633
INFO:openflow.of_01:[00-00-00-00-00-01 1] connected
INFO:iplb:IP Load Balancer Ready.
INFO:iplb:Load Balancing on [00-00-00-00-00-01 1]
INFO:iplb.00-00-00-00-00-01:Server 10.0.0.1 up
INFO:iplb.00-00-00-00-00-01:Server 10.0.0.2 up
INFO:iplb.00-00-00-00-00-01:Server 10.0.0.3 up
```

This screenshot shows that all the servers are up.

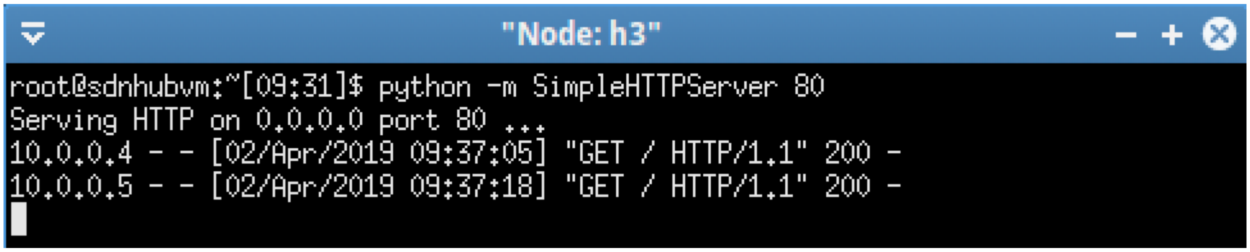
After creating the topology, the servers have to be set up. In node 1, 2, and 3 the SimpleHTTPServer is set up on port 80. The servers are set up with individual IP addresses. This has been shown in the screenshots below.



```
root@sdnhubvm:~[09:31]$ python -m SimpleHTTPServer 80
Serving HTTP on 0.0.0.0 port 80 ...
10.0.0.5 - - [02/Apr/2019 09:36:56] "GET / HTTP/1.1" 200 -
10.0.0.4 - - [02/Apr/2019 09:37:15] "GET / HTTP/1.1" 200 -
```

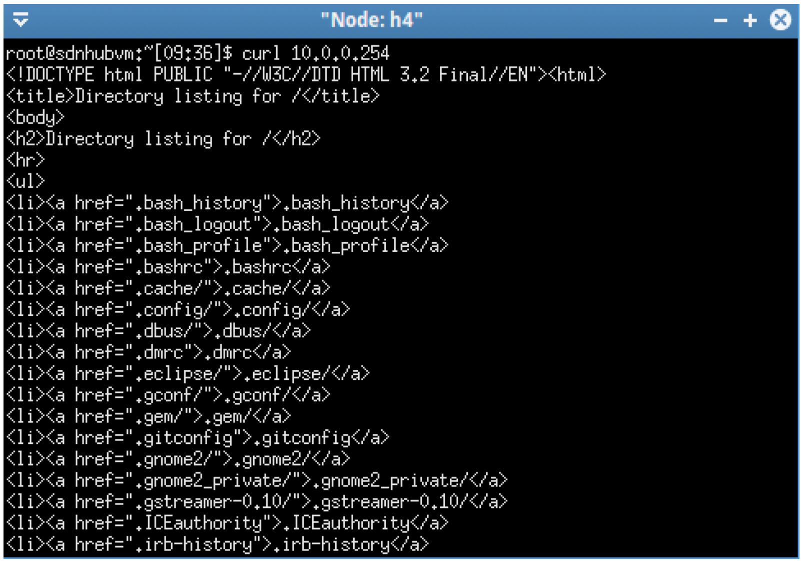


```
root@sdnhubvm:~[09:31]$ python -m SimpleHTTPServer 80
Serving HTTP on 0.0.0.0 port 80 ...
10.0.0.5 - - [02/Apr/2019 09:37:12] "GET / HTTP/1.1" 200 -
10.0.0.4 - - [02/Apr/2019 09:37:25] "GET / HTTP/1.1" 200 -
```

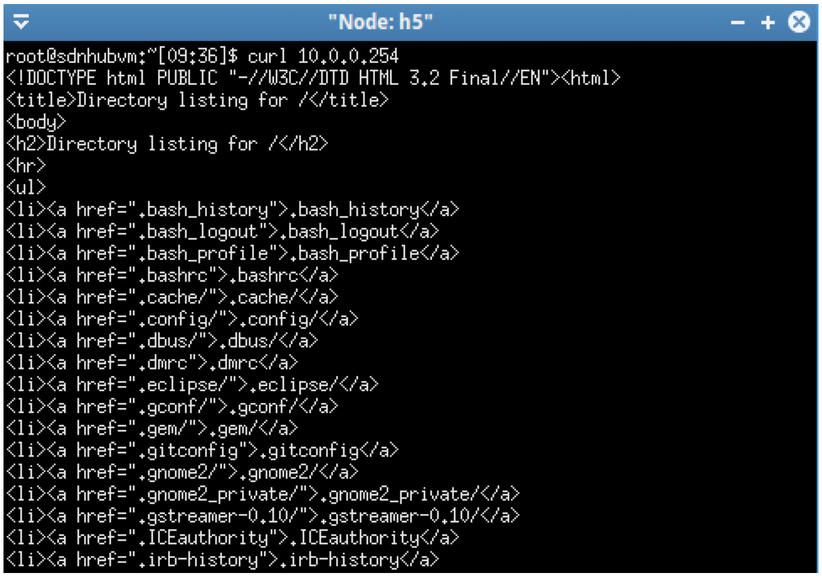


```
root@sdnhubvm:~[09:31]$ python -m SimpleHTTPServer 80
Serving HTTP on 0.0.0.0 port 80 ...
10.0.0.4 - - [02/Apr/2019 09:37:05] "GET / HTTP/1.1" 200 -
10.0.0.5 - - [02/Apr/2019 09:37:18] "GET / HTTP/1.1" 200 -
```

After this, the curl command is used to send the traffic to the server. The curl command is used from both the HTTPClient nodes. The below screenshot shows the clients sending traffic to the server.

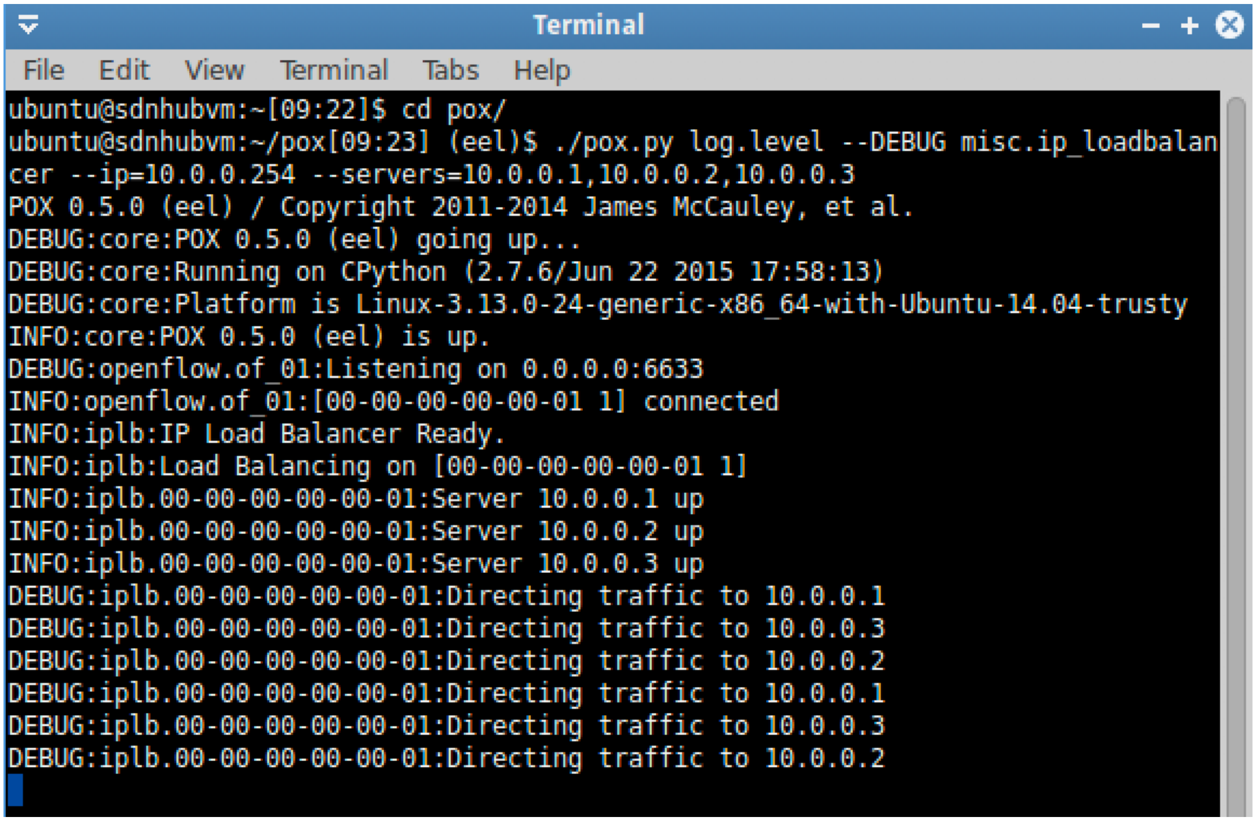


```
root@sdnhubvm:~[09:36]$ curl 10.0.0.254
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 3.2 Final//EN"><html>
<title>Directory listing for /</title>
<body>
<h2>Directory listing for /</h2>
<hr>
<ul>
<li><a href=".bash_history">.bash_history</a>
<li><a href=".bash_logout">.bash_logout</a>
<li><a href=".bash_profile">.bash_profile</a>
<li><a href=".bashrc">.bashrc</a>
<li><a href=".cache/">.cache/</a>
<li><a href=".config/">.config/</a>
<li><a href=".dbus/">.dbus/</a>
<li><a href=".dmrc">.dmrc</a>
<li><a href=".eclipse/">.eclipse/</a>
<li><a href=".gconf/">.gconf/</a>
<li><a href=".gem/">.gem/</a>
<li><a href=".gitconfig">.gitconfig</a>
<li><a href=".gnome2/">.gnome2/</a>
<li><a href=".gnome2_private/">.gnome2_private/</a>
<li><a href=".gstreamer-0.10/">.gstreamer-0.10/</a>
<li><a href=".ICEauthority">.ICEauthority</a>
<li><a href=".irb-history">.irb-history</a>
```



```
root@sdnhubvm:~[09:36]$ curl 10.0.0.254
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 3.2 Final//EN"><html>
<title>Directory listing for /</title>
<body>
<h2>Directory listing for /</h2>
<hr>
<ul>
<li><a href=".bash_history">.bash_history</a>
<li><a href=".bash_logout">.bash_logout</a>
<li><a href=".bash_profile">.bash_profile</a>
<li><a href=".bashrc">.bashrc</a>
<li><a href=".cache/">.cache/</a>
<li><a href=".config/">.config/</a>
<li><a href=".dbus/">.dbus/</a>
<li><a href=".dmrc">.dmrc</a>
<li><a href=".eclipse/">.eclipse/</a>
<li><a href=".gconf/">.gconf/</a>
<li><a href=".gem/">.gem/</a>
<li><a href=".gitconfig">.gitconfig</a>
<li><a href=".gnome2/">.gnome2/</a>
<li><a href=".gnome2_private/">.gnome2_private/</a>
<li><a href=".gstreamer-0.10/">.gstreamer-0.10/</a>
<li><a href=".ICEauthority">.ICEauthority</a>
<li><a href=".irb-history">.irb-history</a>
```

The below screenshot shows directing of different traffic from nodes to servers. The server starts to direct traffic alternatively. Thus in a round robin fashion the client gets its server.



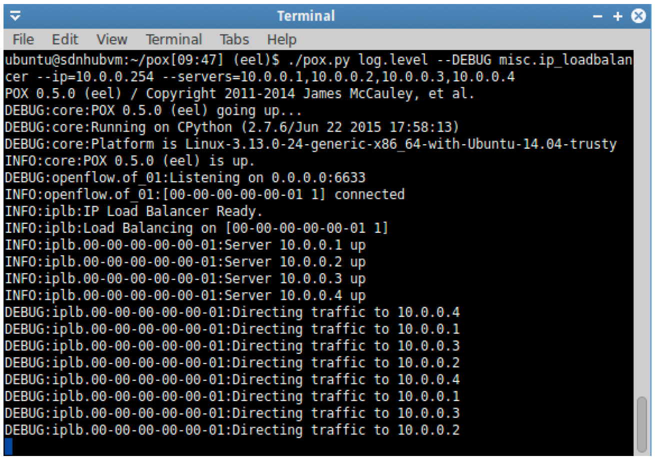
```
ubuntu@sdnhubvm:~[09:22]$ cd pox/
ubuntu@sdnhubvm:~/pox[09:23] (eel)$ ./pox.py log.level --DEBUG misc.ip_loadbalan
cer --ip=10.0.0.254 --servers=10.0.0.1,10.0.0.2,10.0.0.3
POX 0.5.0 (eel) / Copyright 2011-2014 James McCauley, et al.
DEBUG:core:POX 0.5.0 (eel) going up...
DEBUG:core:Running on CPython (2.7.6/Jun 22 2015 17:58:13)
DEBUG:core:Platform is Linux-3.13.0-24-generic-x86_64-with-Ubuntu-14.04-trusty
INFO:core:POX 0.5.0 (eel) is up.
DEBUG:openflow.of_01:Listening on 0.0.0.0:6633
INFO:openflow.of_01:[00-00-00-00-00-01 1] connected
INFO:iplb:IP Load Balancer Ready.
INFO:iplb:Load Balancing on [00-00-00-00-00-01 1]
INFO:iplb.00-00-00-00-00-01:Server 10.0.0.1 up
INFO:iplb.00-00-00-00-00-01:Server 10.0.0.2 up
INFO:iplb.00-00-00-00-00-01:Server 10.0.0.3 up
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.1
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.3
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.1
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.3
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
```

Now we need to add one more server and check if load is balanced between the 4 servers.

Create a topology using the command:
sudo mn —topo=single,6 —mac —arp —controller=remote
This command creates a topology with 4 servers (h1, h2, H3, h4) and 2 clients (H5, h6).

Now we repeat all of the steps above to check if load balancing works.

From the screenshot below we can see that load is balanced between all the servers (old and new).

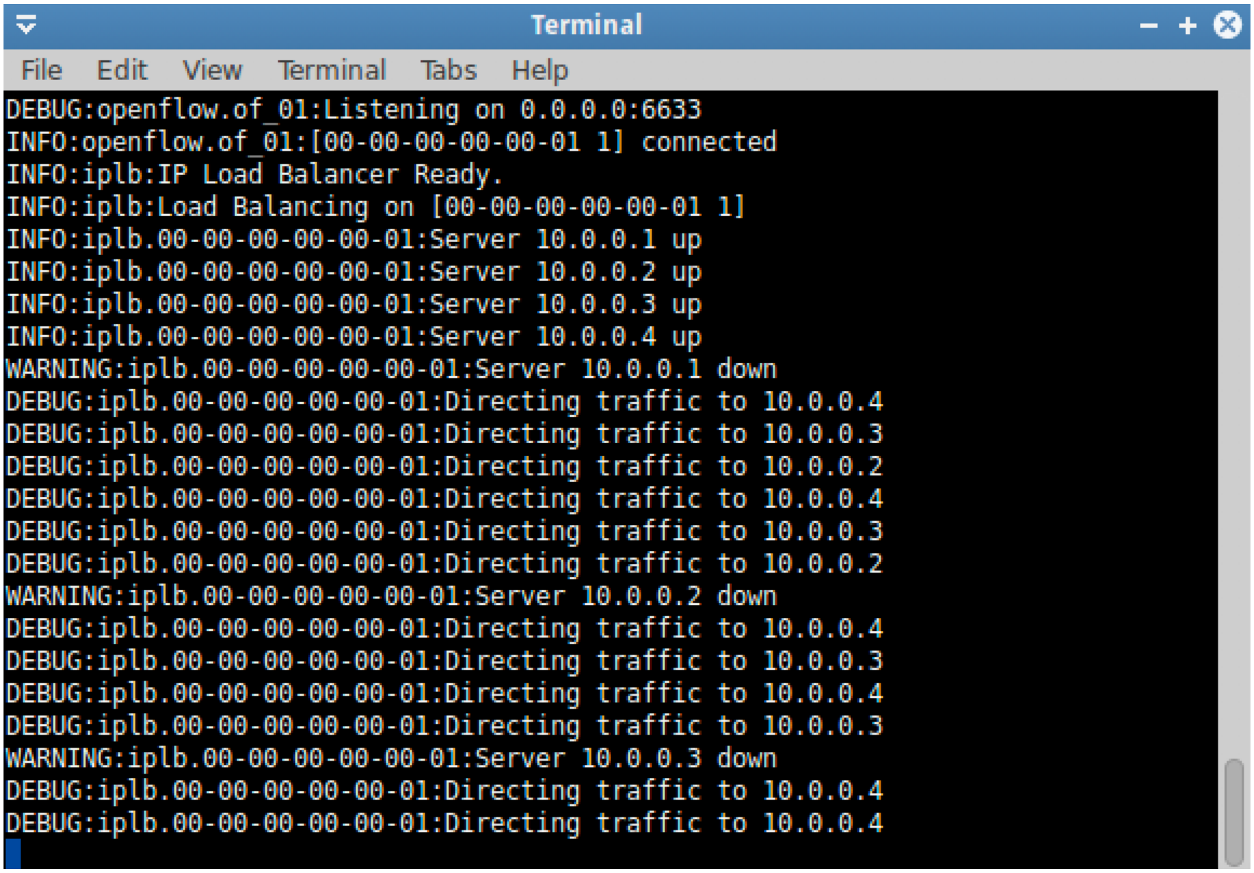


```
ubuntu@sdnhubvm:~/pox[09:47] (eel)$ ./pox.py log.level --DEBUG misc.ip_loadbalan
cer --ip=10.0.0.254 --servers=10.0.0.1,10.0.0.2,10.0.0.3,10.0.0.4
POX 0.5.0 (eel) / Copyright 2011-2014 James McCauley, et al.
DEBUG:core:POX 0.5.0 (eel) going up...
DEBUG:core:Running on CPython (2.7.6/Jun 22 2015 17:58:13)
DEBUG:core:Platform is Linux-3.13.0-24-generic-x86_64-with-Ubuntu-14.04-trusty
INFO:core:POX 0.5.0 (eel) is up.
DEBUG:openflow.of_01:Listening on 0.0.0.0:6633
INFO:openflow.of_01:[00-00-00-00-00-01 1] connected
INFO:iplb:IP Load Balancer Ready.
INFO:iplb:Load Balancing on [00-00-00-00-00-01 1]
INFO:iplb.00-00-00-00-00-01:Server 10.0.0.1 up
INFO:iplb.00-00-00-00-00-01:Server 10.0.0.2 up
INFO:iplb.00-00-00-00-00-01:Server 10.0.0.3 up
INFO:iplb.00-00-00-00-00-01:Server 10.0.0.4 up
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.4
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.1
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.3
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.4
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.1
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.3
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
```

Now we need to check that if one or more server is down the load should be balanced between the other live servers.

Use the following command in the mininet terminal to disable the link between switch and server:
link s1 hx down (where x is the server i.e h1, h2, H3, h4)

The screenshot below shows that even if servers are down load is balanced between the other live servers.



```
DEBUG:openflow.of_01:Listening on 0.0.0.0:6633
INFO:openflow.of_01:[00-00-00-00-00-01 1] connected
INFO:iplb:IP Load Balancer Ready.
INFO:iplb:Load Balancing on [00-00-00-00-00-01 1]
INFO:iplb.00-00-00-00-00-01:Server 10.0.0.1 up
INFO:iplb.00-00-00-00-00-01:Server 10.0.0.2 up
INFO:iplb.00-00-00-00-00-01:Server 10.0.0.3 up
INFO:iplb.00-00-00-00-00-01:Server 10.0.0.4 up
WARNING:iplb.00-00-00-00-00-01:Server 10.0.0.1 down
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.4
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.3
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.4
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.3
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.2
WARNING:iplb.00-00-00-00-00-01:Server 10.0.0.2 down
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.4
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.3
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.4
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.3
WARNING:iplb.00-00-00-00-00-01:Server 10.0.0.3 down
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.4
DEBUG:iplb.00-00-00-00-00-01:Directing traffic to 10.0.0.4
```

Hence even if one or more server(s) are down load is balanced between the other servers.

Load Balancing Code:

Go to the ip_loadbalancer.py and make the following changes:

```
 def _pick_server (self, key, inport):
   """
   Pick a server for a (hopefully) new connection
   """
   global selected_server
   #print selected_server, len(self.live_servers) a=self.live_servers.keys()
   a=self.live_servers.keys()
   if selected_server==len(self.live_servers):
     selected_server=0
   try:
     b=a[selected_server]
   except:
     selected_server=0
   b=a[selected_server]
   selected_server+=1
   return b
```

References:

HTTPS://PDFS.SEMANTICSCHOLAR.ORG/4003/55F7F9632E6C2F33024C45788ED4AE279519.PDF

http://csie.nqu.edu.tw/smallko/sdn/mySDN_Lab8.pdf