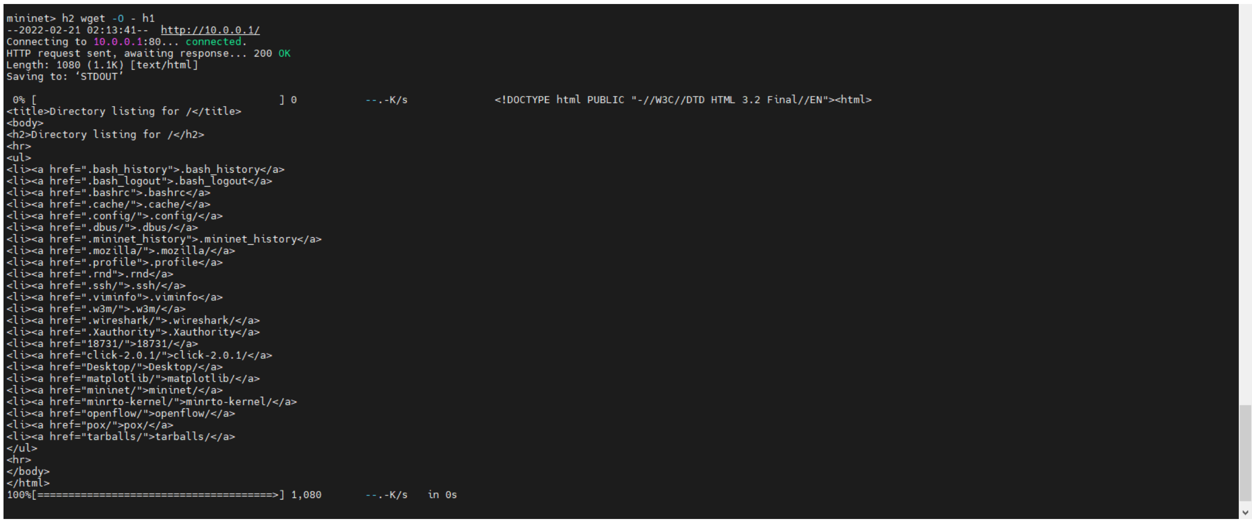
Handin

Hao Zhang
Haozhan2
Network Security

A.Screenshot Reactive

Without Dos

1. As you see, when h2 is requesting h1 (10.0.0.1), we can get the reply successfully.

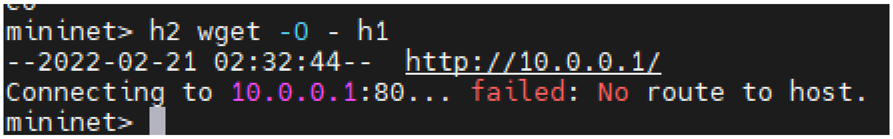


```
mininet> h2 wget -O - h1
--2022-02-21 02:13:41--  http://10.0.0.1/
Connecting to 10.0.0.1:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1080 (1.1K) [text/html]
Saving to: 'STDOUT'

 0% [                                       ] 0           --.-K/s              <!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=".bashrc">.bashrc</a>
<li><a href=".cache/">.cache/</a>
<li><a href=".config/">.config/</a>
<li><a href=".dbus/">.dbus/</a>
<li><a href=".mininet_history">.mininet_history</a>
<li><a href=".mozilla/">.mozilla/</a>
<li><a href=".profile">.profile</a>
<li><a href=".rnd">.rnd</a>
<li><a href=".ssh/">.ssh/</a>
<li><a href=".viminfo">.viminfo</a>
<li><a href=".w3m/">.w3m/</a>
<li><a href=".wireshark/">.wireshark/</a>
<li><a href=".Xauthority">.Xauthority</a>
<li><a href="18731/">18731/</a>
<li><a href="click-2.0.1/">click-2.0.1/</a>
<li><a href="Desktop/">Desktop/</a>
<li><a href="matplotlib/">matplotlib/</a>
<li><a href="mininet/">mininet/</a>
<li><a href="minrto-kernel/">minrto-kernel/</a>
<li><a href="openflow/">openflow/</a>
<li><a href="pox/">pox/</a>
<li><a href="tarballs/">tarballs/</a>
</ul>
<hr>
</body>
</html>
100%[======================================>] 1,080       --.-K/s   in 0s
```

With Dos

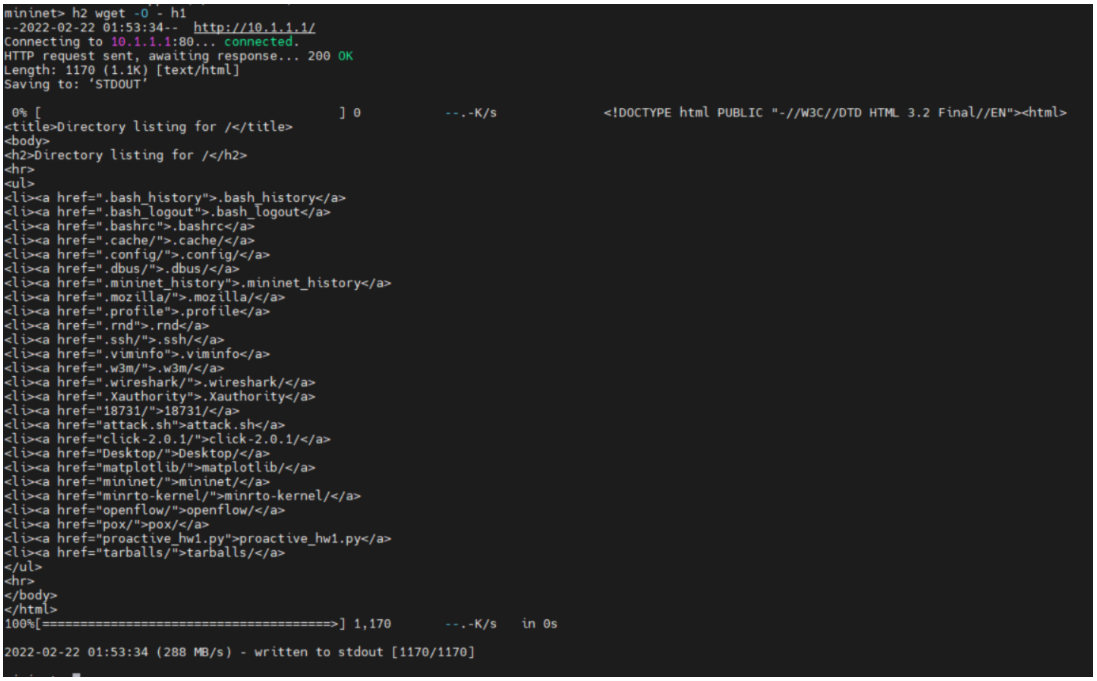
2. Now, in xterm3, we launch attack.sh. We cannot request h1 any more



```
mininet> h2 wget -O - h1
--2022-02-21 02:32:44--  http://10.0.0.1/
Connecting to 10.0.0.1:80... failed: No route to host.
mininet>
```

B. Screenshot for proactive controller

Without Dos: now, the h1 is 10.1.1.1 in proactive topology by running python proactive script. Just like before, before flood, h2 can get reply back from h1.

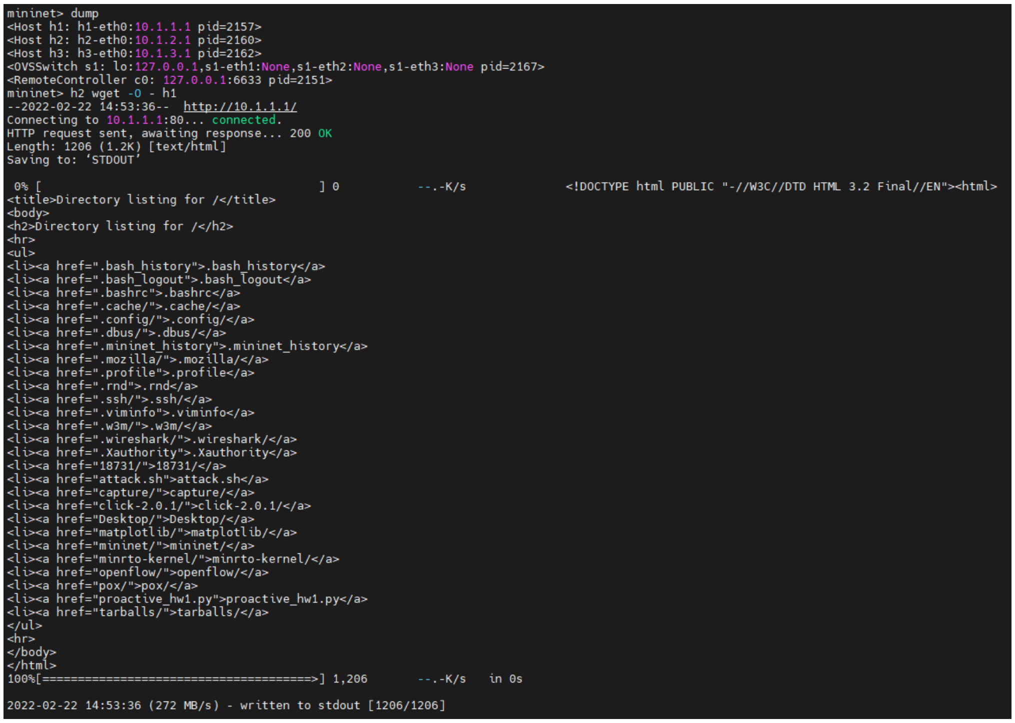


```
mininet> h2 wget -O - h1
--2022-02-22 01:53:34--  http://10.1.1.1/
Connecting to 10.1.1.1:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1170 (1.1K) [text/html]
Saving to: ‘STDOUT’

 0% [                                    ] 0           --.-K/s              <!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=".bashrc">.bashrc</a>
<li><a href=".cache/">.cache/</a>
<li><a href=".config/">.config/</a>
<li><a href=".dbus/">.dbus/</a>
<li><a href=".mininet_history">.mininet_history</a>
<li><a href=".mozilla/">.mozilla/</a>
<li><a href=".profile">.profile</a>
<li><a href=".rnd">.rnd</a>
<li><a href=".ssh/">.ssh/</a>
<li><a href=".viminfo">.viminfo</a>
<li><a href=".w3m/">.w3m/</a>
<li><a href=".wireshark/">.wireshark/</a>
<li><a href=".Xauthority">.Xauthority</a>
<li><a href="18731/">18731/</a>
<li><a href="attack.sh">attack.sh</a>
<li><a href="click-2.0.1/">click-2.0.1/</a>
<li><a href="Desktop/">Desktop/</a>
<li><a href="matplotlib/">matplotlib/</a>
<li><a href="mininet/">mininet/</a>
<li><a href="minrto-kernel/">minrto-kernel/</a>
<li><a href="openflow/">openflow/</a>
<li><a href="pox/">pox/</a>
<li><a href="proactive_hw1.py">proactive_hw1.py</a>
<li><a href="tarballs/">tarballs/</a>
</ul>
<hr>
</body>
</html>
100%[=====================================>] 1,170       --.-K/s   in 0s

2022-02-22 01:53:34 (288 MB/s) - written to stdout [1170/1170]
```

With Dos: Due to proactive controller, h2 can still get reply back from h1.



```
mininet> dump
<Host h1: h1-eth0:10.1.1.1 pid=2157>
<Host h2: h2-eth0:10.1.2.1 pid=2160>
<Host h3: h3-eth0:10.1.3.1 pid=2162>
<OVSSwitch s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None,s1-eth3:None pid=2167>
<RemoteController c0: 127.0.0.1:6633 pid=2151>
mininet> h2 wget -O - h1
--2022-02-22 14:53:36--  http://10.1.1.1/
Connecting to 10.1.1.1:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1206 (1.2K) [text/html]
Saving to: ‘STDOUT’

 0% [                                    ] 0           --.-K/s              <!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=".bashrc">.bashrc</a>
<li><a href=".cache/">.cache/</a>
<li><a href=".config/">.config/</a>
<li><a href=".dbus/">.dbus/</a>
<li><a href=".mininet_history">.mininet_history</a>
<li><a href=".mozilla/">.mozilla/</a>
<li><a href=".profile">.profile</a>
<li><a href=".rnd">.rnd</a>
<li><a href=".ssh/">.ssh/</a>
<li><a href=".viminfo">.viminfo</a>
<li><a href=".w3m/">.w3m/</a>
<li><a href=".wireshark/">.wireshark/</a>
<li><a href=".Xauthority">.Xauthority</a>
<li><a href="18731/">18731/</a>
<li><a href="attack.sh">attack.sh</a>
<li><a href="capture/">capture/</a>
<li><a href="click-2.0.1/">click-2.0.1/</a>
<li><a href="Desktop/">Desktop/</a>
<li><a href="matplotlib/">matplotlib/</a>
<li><a href="mininet/">mininet/</a>
<li><a href="minrto-kernel/">minrto-kernel/</a>
<li><a href="openflow/">openflow/</a>
<li><a href="pox/">pox/</a>
<li><a href="proactive_hw1.py">proactive_hw1.py</a>
<li><a href="tarballs/">tarballs/</a>
</ul>
<hr>
</body>
</html>
100%[=====================================>] 1,206       --.-K/s   in 0s

2022-02-22 14:53:36 (272 MB/s) - written to stdout [1206/1206]
```

C. Explanation of attack code.

```
# Execute in h3 terminal through xterm h3
hping3 -S -p 80 -i 1 --flood --rand-source 10.0.0.1
```

The idea of attack is that because the system is using reactive controller, if host A wants to initiate a connection with host B, host A would needs to ask controller for the path to reach B(a server). Now, Mallory can exploit this system by spoofing random IP address and flood request to B. Then controller would needs to resolve the path of all these requests and set up a path from this spoofed IP address to B, which the system can die.

Now, as shown above, I launch the attack on h3 targeting Http server h1(10.0.0.1) with port 80. I set the frequency of packets sent to 1 packet /s by setting -i to 1. I then enable flood. Also, since we want to thrash the controller, I enable random address so each new requests need to be resolved by the controller.

D. Difference between reactive and proactive

Yes. There are differences on flow table, traffic capture, controller requests, and how attack works.

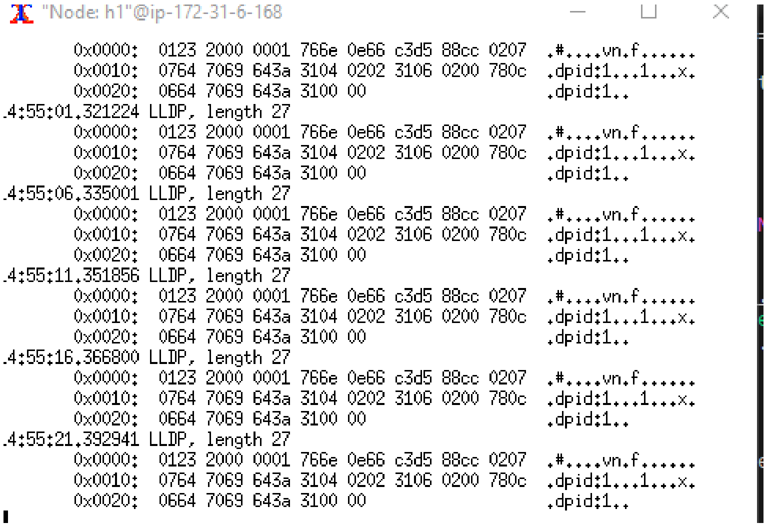
Flow Table

Reactive

Before the flood, tcpdump h1 gives no entries.

Proactive

In proactive mode, the controller constantly discovers the network through Overflow Discovery protocol (OFDP), which sends out a modified version of LLDP packets as shown below.

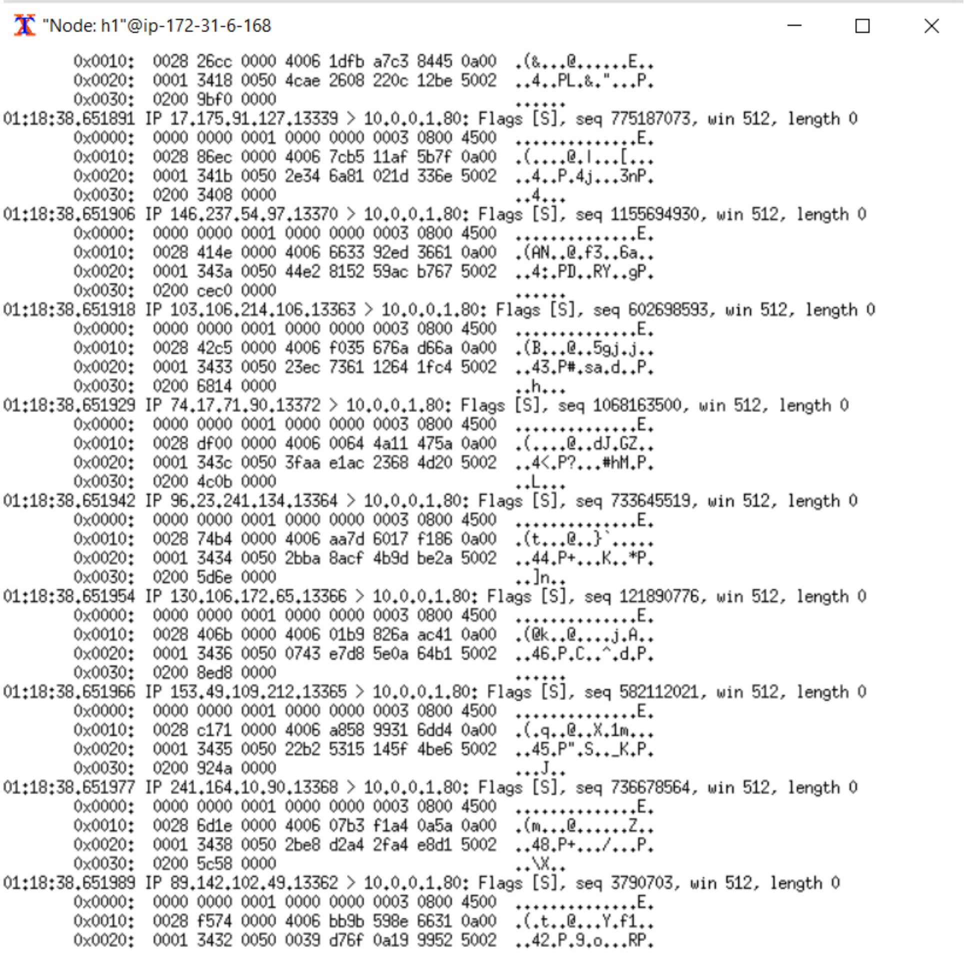


```
        0x0000:  0123 2000 0001 766e 0e66 c3d5 88cc 0207  .#....vn.f......
        0x0010:  0764 7069 643a 3104 0202 3106 0200 780c  .dpid:1...1...x.
        0x0020:  0664 7069 643a 3100 00                   .dpid:1..
.4:55:01.321224 LLDP, length 27
        0x0000:  0123 2000 0001 766e 0e66 c3d5 88cc 0207  .#....vn.f......
        0x0010:  0764 7069 643a 3104 0202 3106 0200 780c  .dpid:1...1...x.
        0x0020:  0664 7069 643a 3100 00                   .dpid:1..
.4:55:06.335001 LLDP, length 27
        0x0000:  0123 2000 0001 766e 0e66 c3d5 88cc 0207  .#....vn.f......
        0x0010:  0764 7069 643a 3104 0202 3106 0200 780c  .dpid:1...1...x.
        0x0020:  0664 7069 643a 3100 00                   .dpid:1..
.4:55:11.351856 LLDP, length 27
        0x0000:  0123 2000 0001 766e 0e66 c3d5 88cc 0207  .#....vn.f......
        0x0010:  0764 7069 643a 3104 0202 3106 0200 780c  .dpid:1...1...x.
        0x0020:  0664 7069 643a 3100 00                   .dpid:1..
.4:55:16.366800 LLDP, length 27
        0x0000:  0123 2000 0001 766e 0e66 c3d5 88cc 0207  .#....vn.f......
        0x0010:  0764 7069 643a 3104 0202 3106 0200 780c  .dpid:1...1...x.
        0x0020:  0664 7069 643a 3100 00                   .dpid:1..
.4:55:21.392941 LLDP, length 27
        0x0000:  0123 2000 0001 766e 0e66 c3d5 88cc 0207  .#....vn.f......
        0x0010:  0764 7069 643a 3104 0202 3106 0200 780c  .dpid:1...1...x.
        0x0020:  0664 7069 643a 3100 00                   .dpid:1..
```

Traffic Capture

Reactive

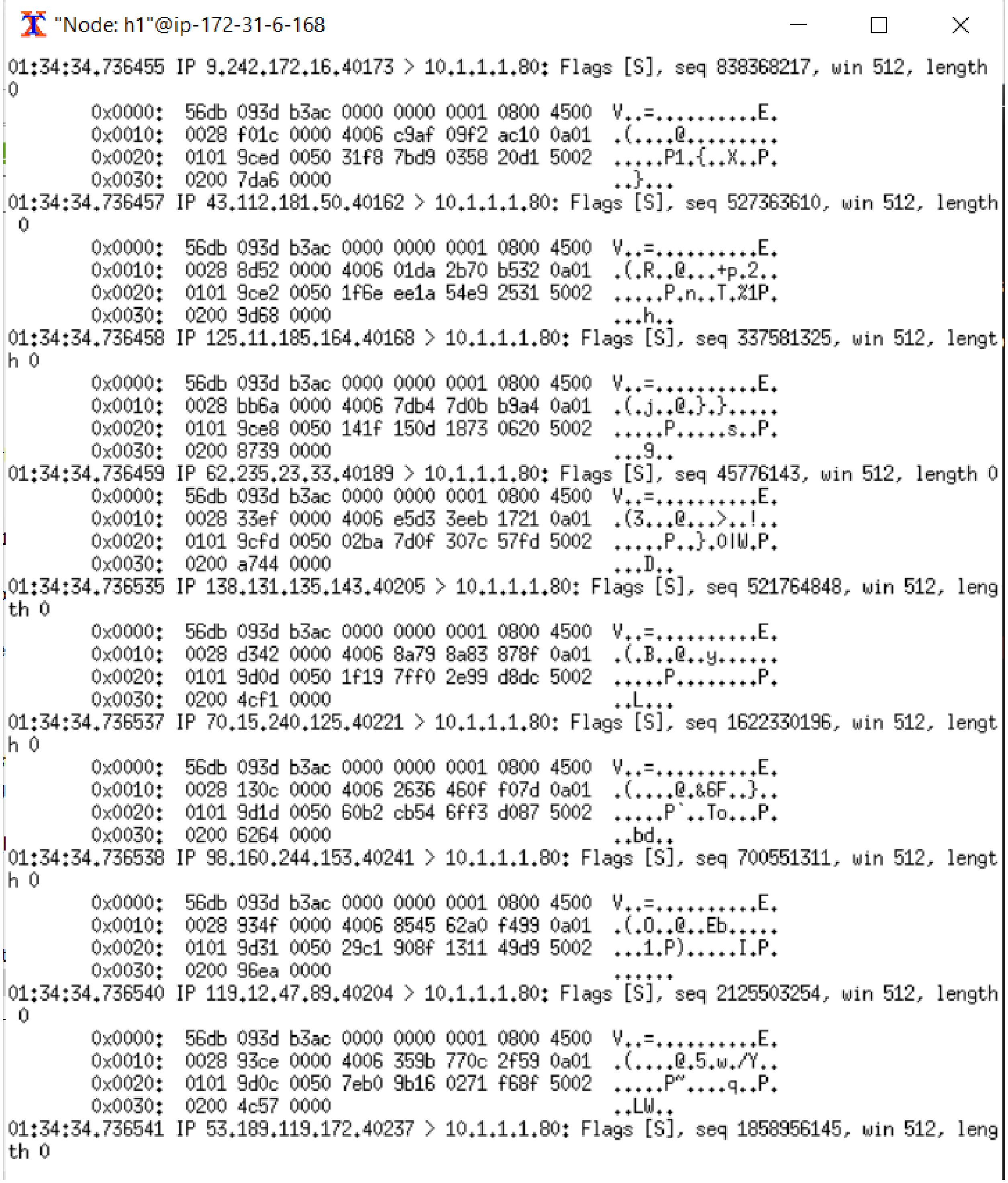
When I launch an attack in reactive mode, all the packets with random spoofed addresses are sending requests to my HTTP server h1. HTTP server would be overwhelmed and down (the screenshot below is before I set the packet frequency to 1 s, so there is so inconsistency) .My controller needs to install the flow for every single new request from a different IP address as shown below.



```
        0x0010:  0028 26cc 0000 4006 1dfb a7c3 8445 0a00  .(&...@......E..
        0x0020:  0001 3418 0050 4cae 2608 220c 12be 5002  ..4..PL.&."...P.
        0x0030:  0200 9bf0 0000                           ......
01:18:38.651891 IP 17.175.91.127.13339 > 10.0.0.1.80: Flags [S], seq 775187073, win 512, length 0
        0x0000:  0000 0000 0001 0000 0000 0003 0800 4500  ..............E.
        0x0010:  0028 86ec 0000 4006 7cb5 11af 5b7f 0a00  .(....@.|...[...
        0x0020:  0001 341b 0050 2e34 6a81 021d 336e 5002  ..4..P.4j...3nP.
        0x0030:  0200 3408 0000                           ..4...
01:18:38.651906 IP 146.237.54.97.13370 > 10.0.0.1.80: Flags [S], seq 1155694930, win 512, length 0
        0x0000:  0000 0000 0001 0000 0000 0003 0800 4500  ..............E.
        0x0010:  0028 414e 0000 4006 6633 92ed 3661 0a00  .(AN..@.f3..6a..
        0x0020:  0001 343a 0050 44e2 8152 59ac b767 5002  ..4:.PD..RY..gP.
        0x0030:  0200 cec0 0000                           ......
01:18:38.651918 IP 103.106.214.106.13363 > 10.0.0.1.80: Flags [S], seq 602698593, win 512, length 0
        0x0000:  0000 0000 0001 0000 0000 0003 0800 4500  ..............E.
        0x0010:  0028 42c5 0000 4006 f035 676a d66a 0a00  .(B...@..5gj.j..
        0x0020:  0001 3433 0050 23ec 7361 1264 1fc4 5002  ..43.P#.sa.d..P.
        0x0030:  0200 6814 0000                           ..h...
01:18:38.651929 IP 74.17.71.90.13372 > 10.0.0.1.80: Flags [S], seq 1068163500, win 512, length 0
        0x0000:  0000 0000 0001 0000 0000 0003 0800 4500  ..............E.
        0x0010:  0028 df00 0000 4006 0064 4a11 475a 0a00  .(....@..dJ.GZ..
        0x0020:  0001 343c 0050 3faa e1ac 2368 4d20 5002  ..4<.P?...#hM.P.
        0x0030:  0200 4c0b 0000                           ..L...
01:18:38.651942 IP 96.23.241.134.13364 > 10.0.0.1.80: Flags [S], seq 733645519, win 512, length 0
        0x0000:  0000 0000 0001 0000 0000 0003 0800 4500  ..............E.
        0x0010:  0028 74b4 0000 4006 aa7d 6017 f186 0a00  .(t...@..}`.....
        0x0020:  0001 3434 0050 2bba 8acf 4b9d be2a 5002  ..44.P+...K..*P.
        0x0030:  0200 5d6e 0000                           ..]n..
01:18:38.651954 IP 130.106.172.65.13366 > 10.0.0.1.80: Flags [S], seq 121890776, win 512, length 0
        0x0000:  0000 0000 0001 0000 0000 0003 0800 4500  ..............E.
        0x0010:  0028 406b 0000 4006 01b9 826a ac41 0a00  .(@k..@....j.A..
        0x0020:  0001 3436 0050 0743 e7d8 5e0a 64b1 5002  ..46.P.C..^.d.P.
        0x0030:  0200 8ed8 0000                           ......
01:18:38.651966 IP 153.49.109.212.13365 > 10.0.0.1.80: Flags [S], seq 582112021, win 512, length 0
        0x0000:  0000 0000 0001 0000 0000 0003 0800 4500  ..............E.
        0x0010:  0028 c171 0000 4006 a858 9931 6dd4 0a00  .(.q..@..X.1m...
        0x0020:  0001 3435 0050 22b2 5315 145f 4be6 5002  ..45.P".S.._K.P.
        0x0030:  0200 924a 0000                           ...J..
01:18:38.651977 IP 241.164.10.90.13368 > 10.0.0.1.80: Flags [S], seq 736678564, win 512, length 0
        0x0000:  0000 0000 0001 0000 0000 0003 0800 4500  ..............E.
        0x0010:  0028 6d1e 0000 4006 07b3 f1a4 0a5a 0a00  .(m...@......Z..
        0x0020:  0001 3438 0050 2be8 d2a4 2fa4 e8d1 5002  ..48.P+.../...P.
        0x0030:  0200 5c58 0000                           ..\X..
01:18:38.651989 IP 89.142.102.49.13362 > 10.0.0.1.80: Flags [S], seq 3790703, win 512, length 0
        0x0000:  0000 0000 0001 0000 0000 0003 0800 4500  ..............E.
        0x0010:  0028 f574 0000 4006 bb9b 598e 6631 0a00  .(.t..@...Y.f1..
        0x0020:  0001 3432 0050 0039 d76f 0a19 9952 5002  ..42.P.9.o...RP.
```

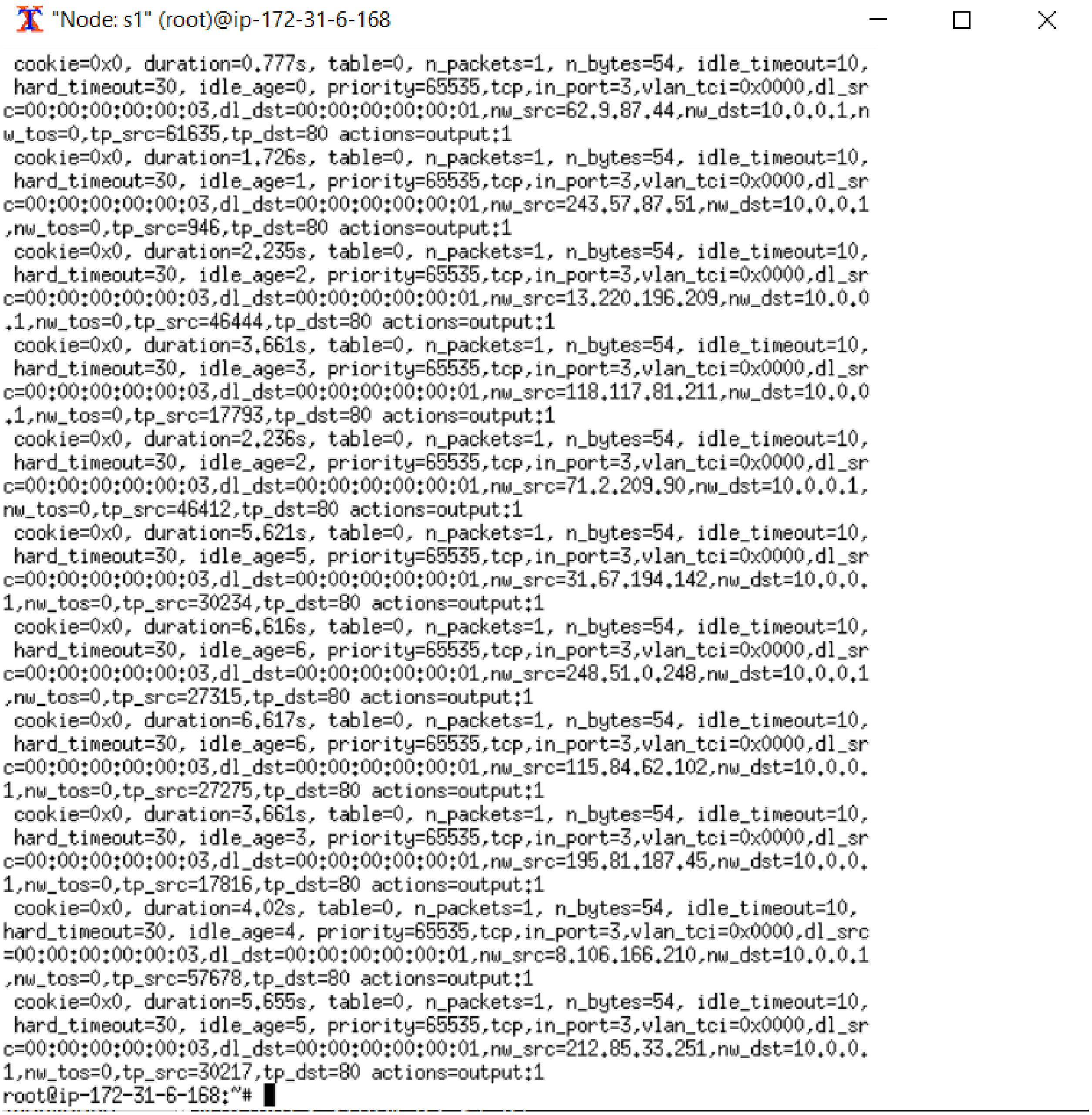
Proactive

In terms of the IP level (layer 3), the traffic flow seems similar to reactive, which makes sense as flood are those spoofed address.



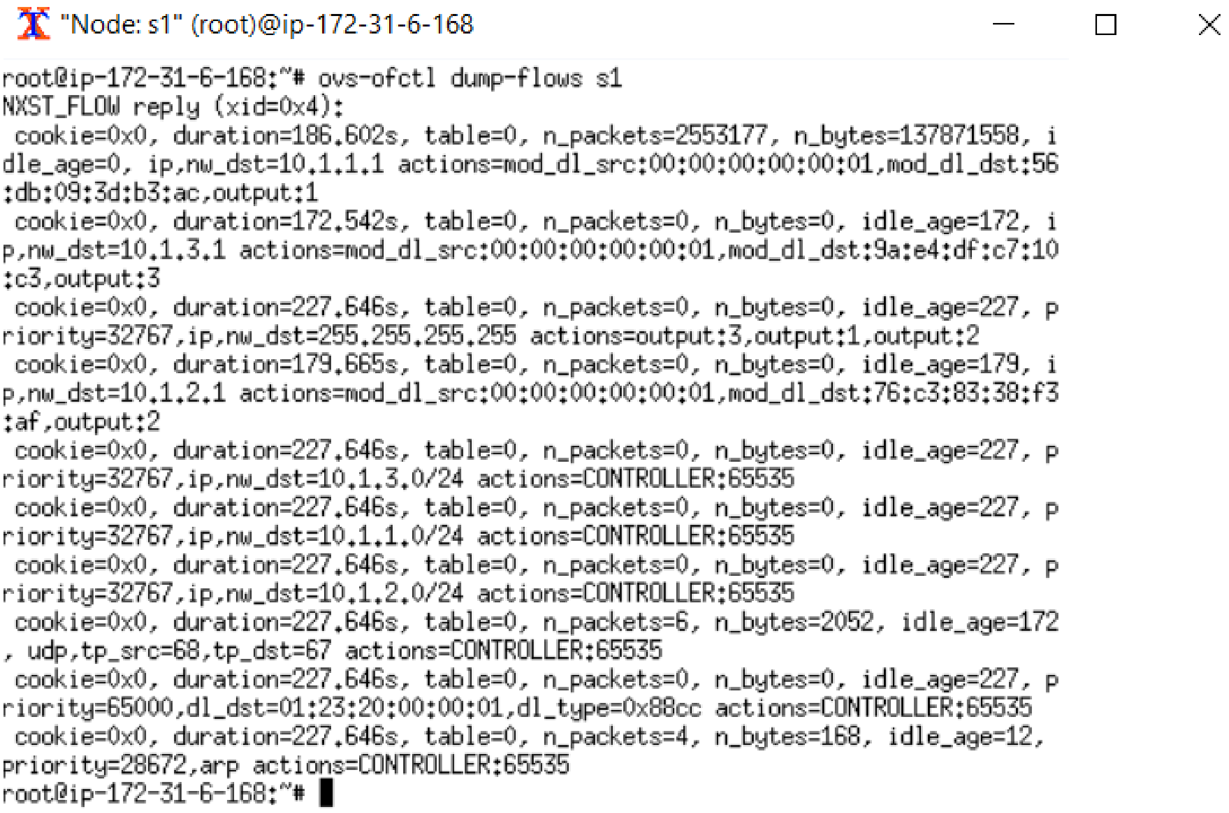
```
01:34:34.736455 IP 9.242.172.16.40173 > 10.1.1.1.80: Flags [S], seq 838368217, win 512, length
0
        0x0000:  56db 093d b3ac 0000 0000 0001 0800 4500  V..=..........E.
        0x0010:  0028 f01c 0000 4006 c9af 09f2 ac10 0a01  .(....@.........
        0x0020:  0101 9ced 0050 31f8 7bd9 0358 20d1 5002  .....P1.{..X..P.
        0x0030:  0200 7da6 0000                           ..}...
01:34:34.736457 IP 43.112.181.50.40162 > 10.1.1.1.80: Flags [S], seq 527363610, win 512, length
 0
        0x0000:  56db 093d b3ac 0000 0000 0001 0800 4500  V..=..........E.
        0x0010:  0028 8d52 0000 4006 01da 2b70 b532 0a01  .(.R..@...+p.2..
        0x0020:  0101 9ce2 0050 1f6e ee1a 54e9 2531 5002  .....P.n..T.%1P.
        0x0030:  0200 9d68 0000                           ...h..
01:34:34.736458 IP 125.11.185.164.40168 > 10.1.1.1.80: Flags [S], seq 337581325, win 512, lengt
h 0
        0x0000:  56db 093d b3ac 0000 0000 0001 0800 4500  V..=..........E.
        0x0010:  0028 bb6a 0000 4006 7db4 7d0b b9a4 0a01  .(.j..@.}.}.....
        0x0020:  0101 9ce8 0050 141f 150d 1873 0620 5002  .....P.....s..P.
        0x0030:  0200 8739 0000                           ...9..
01:34:34.736459 IP 62.235.23.33.40189 > 10.1.1.1.80: Flags [S], seq 45776143, win 512, length 0
        0x0000:  56db 093d b3ac 0000 0000 0001 0800 4500  V..=..........E.
        0x0010:  0028 33ef 0000 4006 e5d3 3eeb 1721 0a01  .(3...@...>..!..
        0x0020:  0101 9cfd 0050 02ba 7d0f 307c 57fd 5002  .....P..}.0|W.P.
        0x0030:  0200 a744 0000                           ...D..
01:34:34.736535 IP 138.131.135.143.40205 > 10.1.1.1.80: Flags [S], seq 521764848, win 512, leng
th 0
        0x0000:  56db 093d b3ac 0000 0000 0001 0800 4500  V..=..........E.
        0x0010:  0028 d342 0000 4006 8a79 8a83 878f 0a01  .(.B..@..y......
        0x0020:  0101 9d0d 0050 1f19 7ff0 2e99 d8dc 5002  .....P........P.
        0x0030:  0200 4cf1 0000                           ..L...
01:34:34.736537 IP 70.15.240.125.40221 > 10.1.1.1.80: Flags [S], seq 1622330196, win 512, lengt
h 0
        0x0000:  56db 093d b3ac 0000 0000 0001 0800 4500  V..=..........E.
        0x0010:  0028 130c 0000 4006 2636 460f f07d 0a01  .(....@.&6F..}..
        0x0020:  0101 9d1d 0050 60b2 cb54 6ff3 d087 5002  .....P`..To...P.
        0x0030:  0200 6264 0000                           ..bd..
01:34:34.736538 IP 98.160.244.153.40241 > 10.1.1.1.80: Flags [S], seq 700551311, win 512, lengt
h 0
        0x0000:  56db 093d b3ac 0000 0000 0001 0800 4500  V..=..........E.
        0x0010:  0028 934f 0000 4006 8545 62a0 f499 0a01  .(.O..@..Eb.....
        0x0020:  0101 9d31 0050 29c1 908f 1311 49d9 5002  ...1.P).....I.P.
        0x0030:  0200 96ea 0000                           ......
01:34:34.736540 IP 119.12.47.89.40204 > 10.1.1.1.80: Flags [S], seq 2125503254, win 512, length
 0
        0x0000:  56db 093d b3ac 0000 0000 0001 0800 4500  V..=..........E.
        0x0010:  0028 93ce 0000 4006 359b 770c 2f59 0a01  .(....@.5.w./Y..
        0x0020:  0101 9d0c 0050 7eb0 9b16 0271 f68f 5002  .....P~....q..P.
        0x0030:  0200 4c57 0000                           ..LW..
01:34:34.736541 IP 53.189.119.172.40237 > 10.1.1.1.80: Flags [S], seq 1858956145, win 512, leng
th 0
```

Reactive
Dump the switch with ovs-ofctl dump-flows s1 after flood.
I find so many flow entries (won’t fit a page of the terminal) with each flow duration around 1 second.



```
 cookie=0x0, duration=0.777s, table=0, n_packets=1, n_bytes=54, idle_timeout=10,
 hard_timeout=30, idle_age=0, priority=65535,tcp,in_port=3,vlan_tci=0x0000,dl_sr
c=00:00:00:00:00:03,dl_dst=00:00:00:00:00:01,nw_src=62.9.87.44,nw_dst=10.0.0.1,n
w_tos=0,tp_src=61635,tp_dst=80 actions=output:1
 cookie=0x0, duration=1.726s, table=0, n_packets=1, n_bytes=54, idle_timeout=10,
 hard_timeout=30, idle_age=1, priority=65535,tcp,in_port=3,vlan_tci=0x0000,dl_sr
c=00:00:00:00:00:03,dl_dst=00:00:00:00:00:01,nw_src=243.57.87.51,nw_dst=10.0.0.1
,nw_tos=0,tp_src=946,tp_dst=80 actions=output:1
 cookie=0x0, duration=2.235s, table=0, n_packets=1, n_bytes=54, idle_timeout=10,
 hard_timeout=30, idle_age=2, priority=65535,tcp,in_port=3,vlan_tci=0x0000,dl_sr
c=00:00:00:00:00:03,dl_dst=00:00:00:00:00:01,nw_src=13.220.196.209,nw_dst=10.0.0
.1,nw_tos=0,tp_src=46444,tp_dst=80 actions=output:1
 cookie=0x0, duration=3.661s, table=0, n_packets=1, n_bytes=54, idle_timeout=10,
 hard_timeout=30, idle_age=3, priority=65535,tcp,in_port=3,vlan_tci=0x0000,dl_sr
c=00:00:00:00:00:03,dl_dst=00:00:00:00:00:01,nw_src=118.117.81.211,nw_dst=10.0.0
.1,nw_tos=0,tp_src=17793,tp_dst=80 actions=output:1
 cookie=0x0, duration=2.236s, table=0, n_packets=1, n_bytes=54, idle_timeout=10,
 hard_timeout=30, idle_age=2, priority=65535,tcp,in_port=3,vlan_tci=0x0000,dl_sr
c=00:00:00:00:00:03,dl_dst=00:00:00:00:00:01,nw_src=71.2.209.90,nw_dst=10.0.0.1,
nw_tos=0,tp_src=46412,tp_dst=80 actions=output:1
 cookie=0x0, duration=5.621s, table=0, n_packets=1, n_bytes=54, idle_timeout=10,
 hard_timeout=30, idle_age=5, priority=65535,tcp,in_port=3,vlan_tci=0x0000,dl_sr
c=00:00:00:00:00:03,dl_dst=00:00:00:00:00:01,nw_src=31.67.194.142,nw_dst=10.0.0.
1,nw_tos=0,tp_src=30234,tp_dst=80 actions=output:1
 cookie=0x0, duration=6.616s, table=0, n_packets=1, n_bytes=54, idle_timeout=10,
 hard_timeout=30, idle_age=6, priority=65535,tcp,in_port=3,vlan_tci=0x0000,dl_sr
c=00:00:00:00:00:03,dl_dst=00:00:00:00:00:01,nw_src=248.51.0.248,nw_dst=10.0.0.1
,nw_tos=0,tp_src=27315,tp_dst=80 actions=output:1
 cookie=0x0, duration=6.617s, table=0, n_packets=1, n_bytes=54, idle_timeout=10,
 hard_timeout=30, idle_age=6, priority=65535,tcp,in_port=3,vlan_tci=0x0000,dl_sr
c=00:00:00:00:00:03,dl_dst=00:00:00:00:00:01,nw_src=115.84.62.102,nw_dst=10.0.0.
1,nw_tos=0,tp_src=27275,tp_dst=80 actions=output:1
 cookie=0x0, duration=3.661s, table=0, n_packets=1, n_bytes=54, idle_timeout=10,
 hard_timeout=30, idle_age=3, priority=65535,tcp,in_port=3,vlan_tci=0x0000,dl_sr
c=00:00:00:00:00:03,dl_dst=00:00:00:00:00:01,nw_src=195.81.187.45,nw_dst=10.0.0.
1,nw_tos=0,tp_src=17816,tp_dst=80 actions=output:1
 cookie=0x0, duration=4.02s, table=0, n_packets=1, n_bytes=54, idle_timeout=10,
hard_timeout=30, idle_age=4, priority=65535,tcp,in_port=3,vlan_tci=0x0000,dl_src
=00:00:00:00:00:03,dl_dst=00:00:00:00:00:01,nw_src=8.106.166.210,nw_dst=10.0.0.1
,nw_tos=0,tp_src=57678,tp_dst=80 actions=output:1
 cookie=0x0, duration=5.655s, table=0, n_packets=1, n_bytes=54, idle_timeout=10,
 hard_timeout=30, idle_age=5, priority=65535,tcp,in_port=3,vlan_tci=0x0000,dl_sr
c=00:00:00:00:00:03,dl_dst=00:00:00:00:00:01,nw_src=212.85.33.251,nw_dst=10.0.0.
1,nw_tos=0,tp_src=30217,tp_dst=80 actions=output:1
root@ip-172-31-6-168:~# █
```

Proactive
Dump the switch after flood only gives out limited number of flow entries. Each flow has a duration of more than 100 seconds.



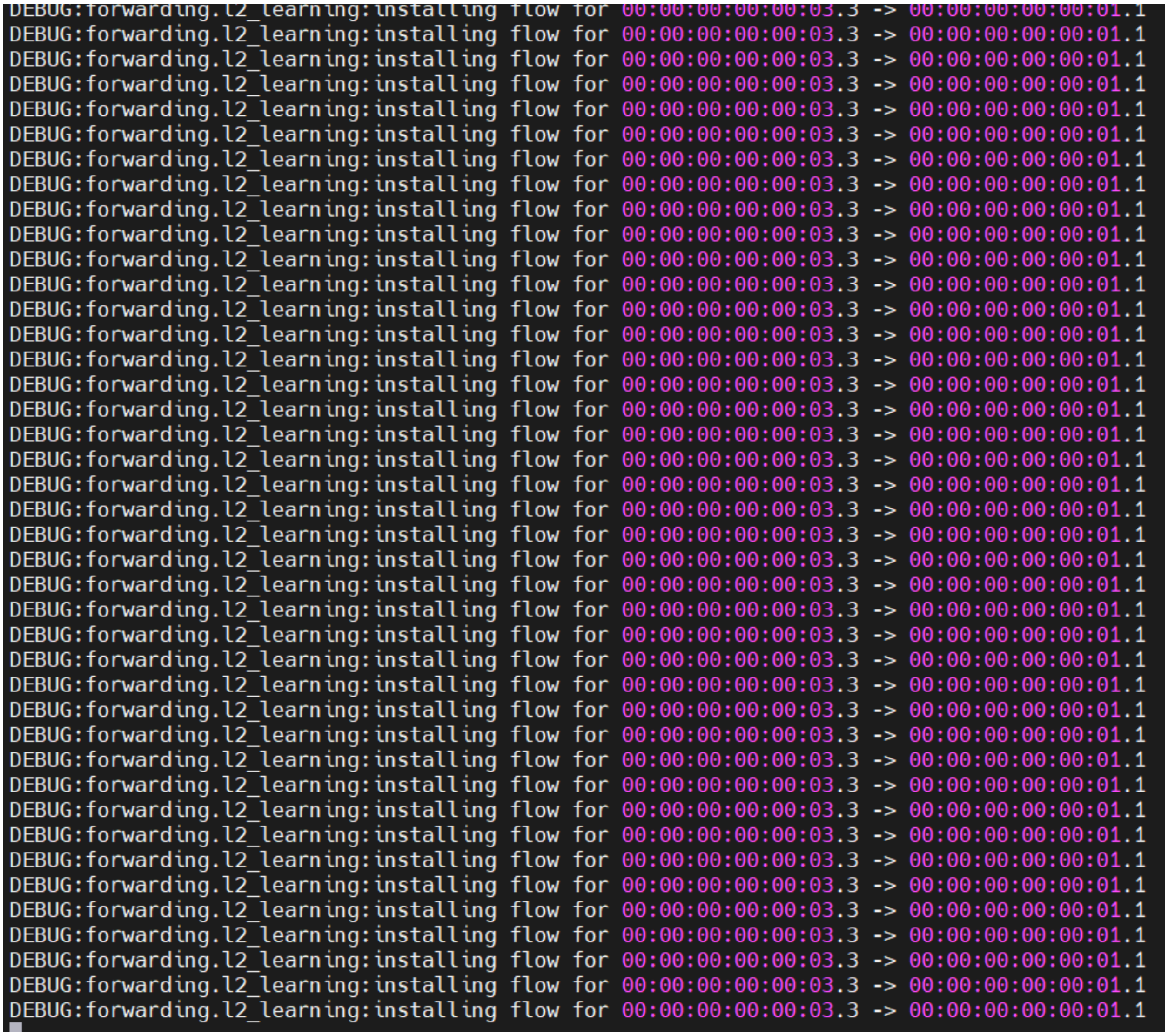
```
root@ip-172-31-6-168:~# ovs-ofctl dump-flows s1
NXST_FLOW reply (xid=0x4):
 cookie=0x0, duration=186.602s, table=0, n_packets=2553177, n_bytes=137871558, i
dle_age=0, ip,nw_dst=10.1.1.1 actions=mod_dl_src:00:00:00:00:00:01,mod_dl_dst:56
:db:09:3d:b3:ac,output:1
 cookie=0x0, duration=172.542s, table=0, n_packets=0, n_bytes=0, idle_age=172, i
p,nw_dst=10.1.3.1 actions=mod_dl_src:00:00:00:00:00:01,mod_dl_dst:9a:e4:df:c7:10
:c3,output:3
 cookie=0x0, duration=227.646s, table=0, n_packets=0, n_bytes=0, idle_age=227, p
riority=32767,ip,nw_dst=255.255.255.255 actions=output:3,output:1,output:2
 cookie=0x0, duration=179.665s, table=0, n_packets=0, n_bytes=0, idle_age=179, i
p,nw_dst=10.1.2.1 actions=mod_dl_src:00:00:00:00:00:01,mod_dl_dst:76:c3:83:38:f3
:af,output:2
 cookie=0x0, duration=227.646s, table=0, n_packets=0, n_bytes=0, idle_age=227, p
riority=32767,ip,nw_dst=10.1.3.0/24 actions=CONTROLLER:65535
 cookie=0x0, duration=227.646s, table=0, n_packets=0, n_bytes=0, idle_age=227, p
riority=32767,ip,nw_dst=10.1.1.0/24 actions=CONTROLLER:65535
 cookie=0x0, duration=227.646s, table=0, n_packets=0, n_bytes=0, idle_age=227, p
riority=32767,ip,nw_dst=10.1.2.0/24 actions=CONTROLLER:65535
 cookie=0x0, duration=227.646s, table=0, n_packets=6, n_bytes=2052, idle_age=172
, udp,tp_src=68,tp_dst=67 actions=CONTROLLER:65535
 cookie=0x0, duration=227.646s, table=0, n_packets=0, n_bytes=0, idle_age=227, p
riority=65000,dl_dst=01:23:20:00:00:01,dl_type=0x88cc actions=CONTROLLER:65535
 cookie=0x0, duration=227.646s, table=0, n_packets=4, n_bytes=168, idle_age=12,
priority=28672,arp actions=CONTROLLER:65535
root@ip-172-31-6-168:~# █
```

By comparison, for reactive controller, every time a host makes a new connection, it will have contact controller, and controller will define a route. That’s why they are many flow entries in the switch dump. Also, every flow entry can timeout in a short period of time.

For proactive, because proactive controller use OFDP to discovery the topology. Hence, it associates entities with mac address. The flow is predefined by controller. Therefore, there aren’t many flow entries, which won’t flood h1.

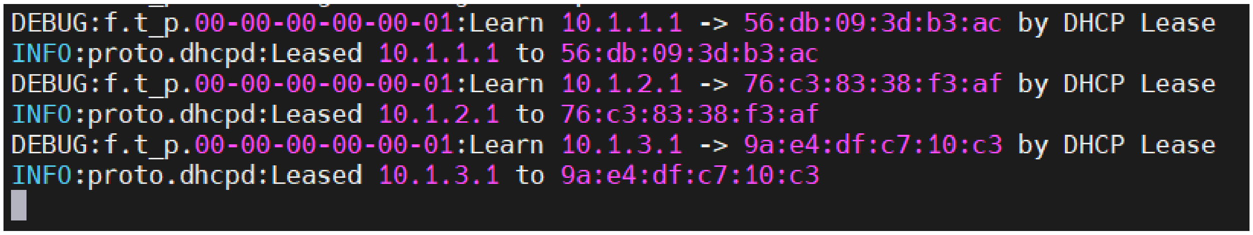
Controller Request

Reactive
During flood, controller will install new flow for every spoofed ip address.



```
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
DEBUG:forwarding.l2_learning:installing flow for 00:00:00:00:00:03.3 -> 00:00:00:00:00:01.1
```

In proactive, the flow is predefined by controller. Therefore, during the flood, I won't see so many install flows.



```
DEBUG:f.t_p.00-00-00-00-00-01:Learn 10.1.1.1 -> 56:db:09:3d:b3:ac by DHCP Lease
INFO:proto.dhcpd:Leased 10.1.1.1 to 56:db:09:3d:b3:ac
DEBUG:f.t_p.00-00-00-00-00-01:Learn 10.1.2.1 -> 76:c3:83:38:f3:af by DHCP Lease
INFO:proto.dhcpd:Leased 10.1.2.1 to 76:c3:83:38:f3:af
DEBUG:f.t_p.00-00-00-00-00-01:Learn 10.1.3.1 -> 9a:e4:df:c7:10:c3 by DHCP Lease
INFO:proto.dhcpd:Leased 10.1.3.1 to 9a:e4:df:c7:10:c3
```