

Alex Davila
1465932
CSE150/L

Pre-Lab Questions

1. What command will show you which groups you are part of?
 - ◆ `$ groups` //shows which groups your Unix/Linux user belongs to
 - ◆ `id -Gn` //to print both numeric and named based group id
2. What does the environmental variable `$?` Hold?
 - ◆ `0`
3. What key combination will suspend a currently running process and place it as a background process?
 - ◆ `Ctrl-C` //kills a process in the foreground and puts it in the background
4. With what command (and arguments) can you find out your kernel version and the “nodename”? [The output should not include any other information]
 - ◆ `uname -r` //you should receive a result similar to the following: `2.6.32-431.11.2.el6.x86_64`

The kernel version output from above can be interpreted with the following key:

2.6.32-431.11.2.el6.x86_64

2	=	The Main Kernel Version
6	=	The Major Revision
32	=	The Minor Revision
431.11.2.el6	=	The Minor Fix/Revision

5. What is the difference between the paths “.”, “..”, and “~”? What does the path “/” refer to when not preceded by anything?
 - PATH is an environmental variable in Linux/Unix OS that tells the shell in which directories to search for executable files
 - ◆ “.” //current directory
 - ◆ “..” //go up a directory
 - ◆ “~” //home directory
 - ◆ “/” //if you type / or ./ this is the root directory

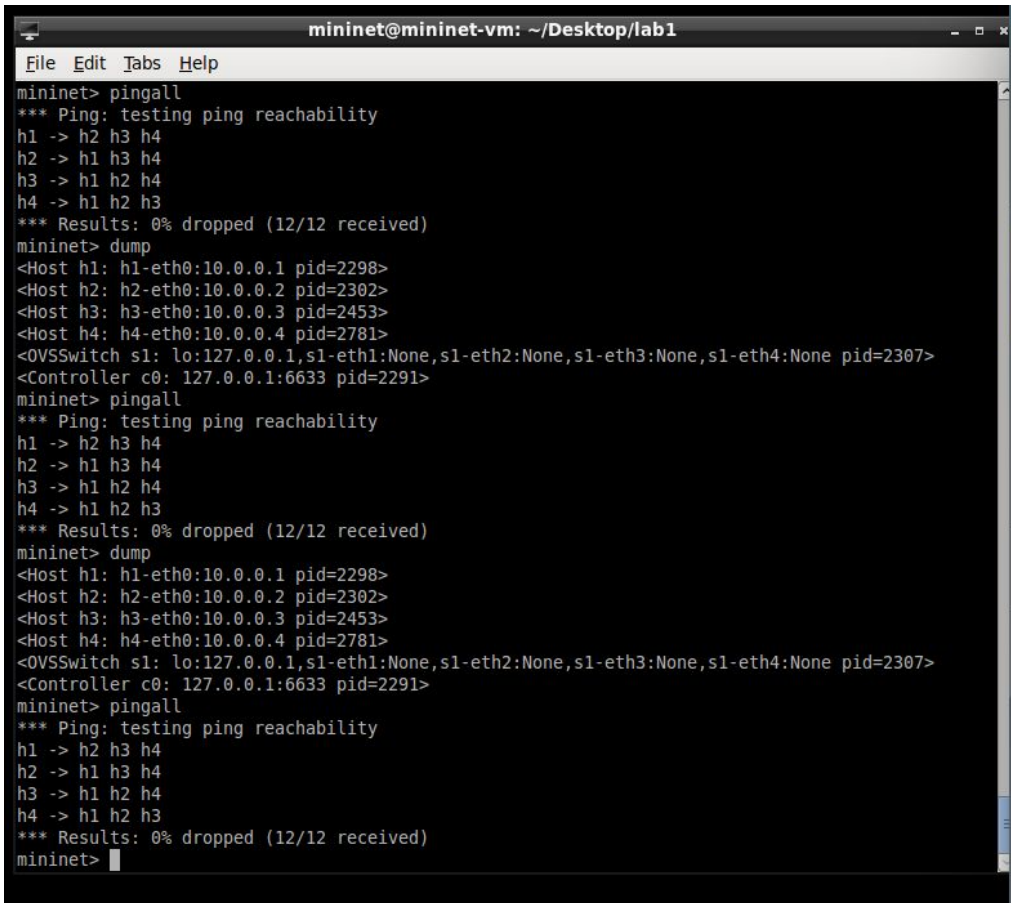
6. What is a pid? Which command would you use to find the “pid” for a running process?
 - ◆ PID - process identifier is a unique number that identifies each of the running processes in an operating system such as Linux, Unix, macOS, and Microsoft Windows.
 - ◆ This can be done by running:
 1. pidof: //find the process ID of a running program
 2. pgrep: //look up or signal processes based on name and other attributes
 3. ps: //report a snapshot of the current processes
 4. pstree: //display a tree processes
 5. ss: //is used to dump socket statistics
 6. netstat: //displays a list of open sockets
 7. lsof: //list open files
 8. fuser: //list process IDs of all processes that have one or more files open
 9. systemctl: //control the systemd system and service manager

Image Name	PID	Session Name	Session #	Mem Usage
System Idle Process	0	Services	0	24 K
System	4	Services	0	154,040 K
smss.exe	308	Services	0	1,424 K
avgchsva.exe	400	Services	0	127,476 K

7. Write a single command that will return every user's default shell. [You may chain commands using piping and redirects] (Hint: See 'Chapter 19: filters' of linux-training.be as well as the man page for the /etc/passwd file: <https://linux.die.net/man/5/passwd>)
 - ◆ `grep -oE '^[^:]+ ' /etc/passwd //gives you the user`
 - ◆ `awk -F": " '$7 == "/usr/bin/false" {print "User: "$1 "shell: "$7}' /etc/passwd //it looks for all users with the shell bin/false`
 - ◆ `awk -F": " '{print "User: "$1 "shell: "$7}' /etc/passwd //it prints all users and their shell`
8. What is the difference between “sudo” and “su root”?
 - ◆ `sudo` //when you want to run a command as root privileges
 - ◆ `su root` //when you want to elevate/switch into the root user
9. How would you tell your computer to run a program or script on a schedule or set interval on Linux?
E.g. Run this program once every 30 minutes.
 - ◆ `* /30 * * * * /path/to/script.sh`
 - ◆ `//use crontab.guru for cron check`
10. Write a shell script that only prints the even numbered lines of each file in the current directory. The output should be filename: line for each even numbered line. You do not need to print line numbers.
 - ◆ `alexdavila-script.sh`

The Lab 1[100 pts]:

1. In Mininet change the default configuration to have 4 hosts connected to a switch
2. Save a screenshot of dump and pingall output. Explain what is being shown in the screenshot.



```
mininet@mininet-vm: ~/Desktop/lab1
File Edit Tabs Help
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4
h2 -> h1 h3 h4
h3 -> h1 h2 h4
h4 -> h1 h2 h3
*** Results: 0% dropped (12/12 received)
mininet> dump
<Host h1: h1-eth0:10.0.0.1 pid=2298>
<Host h2: h2-eth0:10.0.0.2 pid=2302>
<Host h3: h3-eth0:10.0.0.3 pid=2453>
<Host h4: h4-eth0:10.0.0.4 pid=2781>
<OVSSwitch s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None,s1-eth3:None,s1-eth4:None pid=2307>
<Controller c0: 127.0.0.1:6633 pid=2291>
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4
h2 -> h1 h3 h4
h3 -> h1 h2 h4
h4 -> h1 h2 h3
*** Results: 0% dropped (12/12 received)
mininet> dump
<Host h1: h1-eth0:10.0.0.1 pid=2298>
<Host h2: h2-eth0:10.0.0.2 pid=2302>
<Host h3: h3-eth0:10.0.0.3 pid=2453>
<Host h4: h4-eth0:10.0.0.4 pid=2781>
<OVSSwitch s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None,s1-eth3:None,s1-eth4:None pid=2307>
<Controller c0: 127.0.0.1:6633 pid=2291>
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4
h2 -> h1 h3 h4
h3 -> h1 h2 h4
h4 -> h1 h2 h3
*** Results: 0% dropped (12/12 received)
mininet>
```

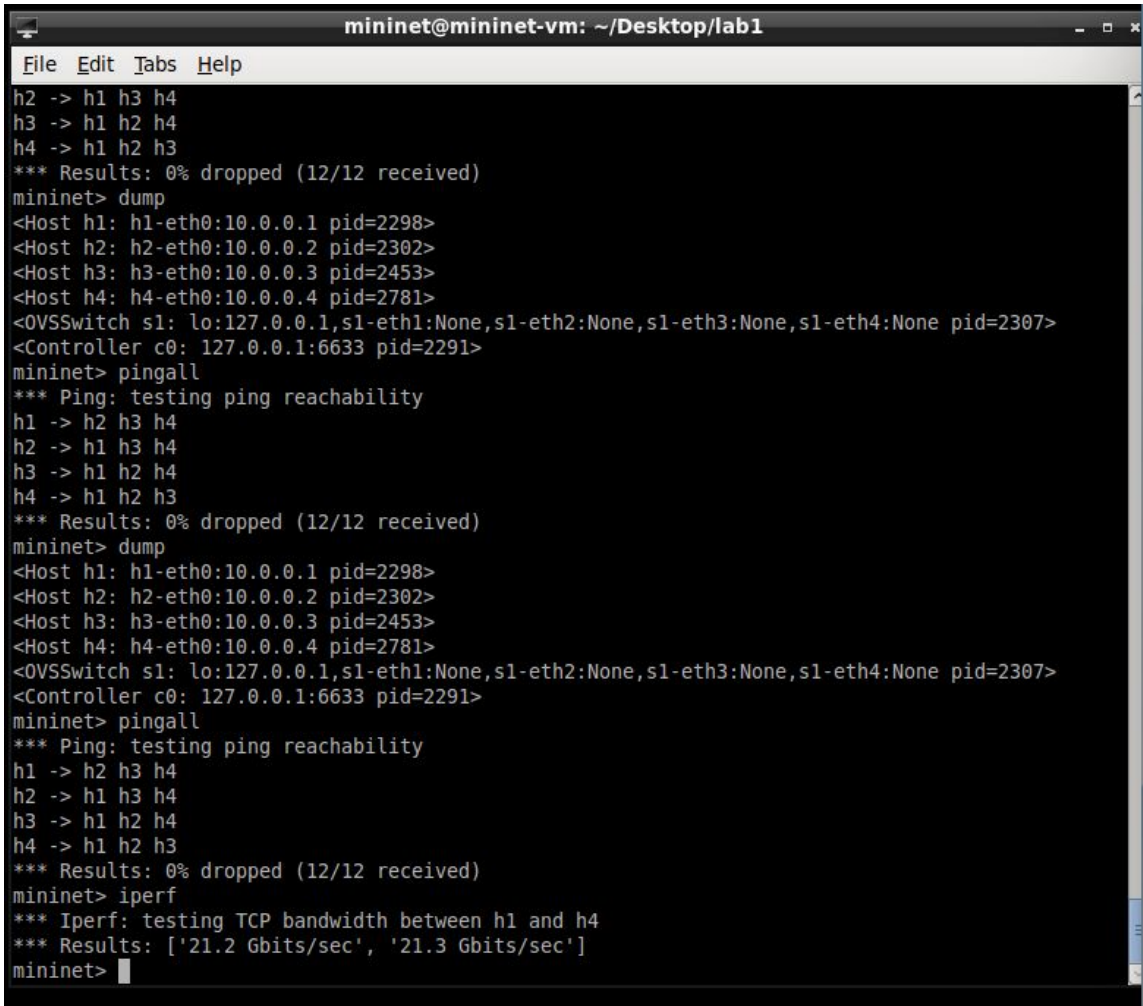
dump:

- dumps node information

pingall:

- it displays the connectivity between all hosts and tells us which hosts are connected to each other

3. Run iperf command. Screenshot the output. How fast is the connect?

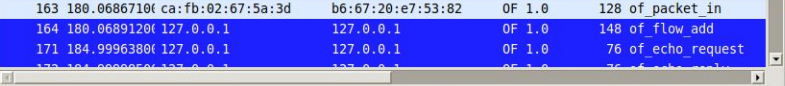
A screenshot of a terminal window titled "mininet@mininet-vm: ~/Desktop/lab1". The terminal shows a series of commands and their outputs. The commands include: "h2 -> h1 h3 h4", "h3 -> h1 h2 h4", "h4 -> h1 h2 h3", "mininet> dump", "mininet> pingall", and "mininet> iperf". The outputs show network topology details, ping results (0% dropped), and iperf results (21.2 Gbits/sec and 21.3 Gbits/sec).

```
mininet@mininet-vm: ~/Desktop/lab1
File Edit Tabs Help
h2 -> h1 h3 h4
h3 -> h1 h2 h4
h4 -> h1 h2 h3
*** Results: 0% dropped (12/12 received)
mininet> dump
<Host h1: h1-eth0:10.0.0.1 pid=2298>
<Host h2: h2-eth0:10.0.0.2 pid=2302>
<Host h3: h3-eth0:10.0.0.3 pid=2453>
<Host h4: h4-eth0:10.0.0.4 pid=2781>
<OVSSwitch s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None,s1-eth3:None,s1-eth4:None pid=2307>
<Controller c0: 127.0.0.1:6633 pid=2291>
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4
h2 -> h1 h3 h4
h3 -> h1 h2 h4
h4 -> h1 h2 h3
*** Results: 0% dropped (12/12 received)
mininet> dump
<Host h1: h1-eth0:10.0.0.1 pid=2298>
<Host h2: h2-eth0:10.0.0.2 pid=2302>
<Host h3: h3-eth0:10.0.0.3 pid=2453>
<Host h4: h4-eth0:10.0.0.4 pid=2781>
<OVSSwitch s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None,s1-eth3:None,s1-eth4:None pid=2307>
<Controller c0: 127.0.0.1:6633 pid=2291>
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4
h2 -> h1 h3 h4
h3 -> h1 h2 h4
h4 -> h1 h2 h3
*** Results: 0% dropped (12/12 received)
mininet> iperf
*** Iperf: testing TCP bandwidth between h1 and h4
*** Results: ['21.2 Gbits/sec', '21.3 Gbits/sec']
mininet> 
```

iperf:

- It is generally used for TCP connection, used to test bandwidth between hosts
- It connects within ['21.2 Gbits/sec', '21.3 Gbits/sec']

- ii. 5 of_packet_in messages show up

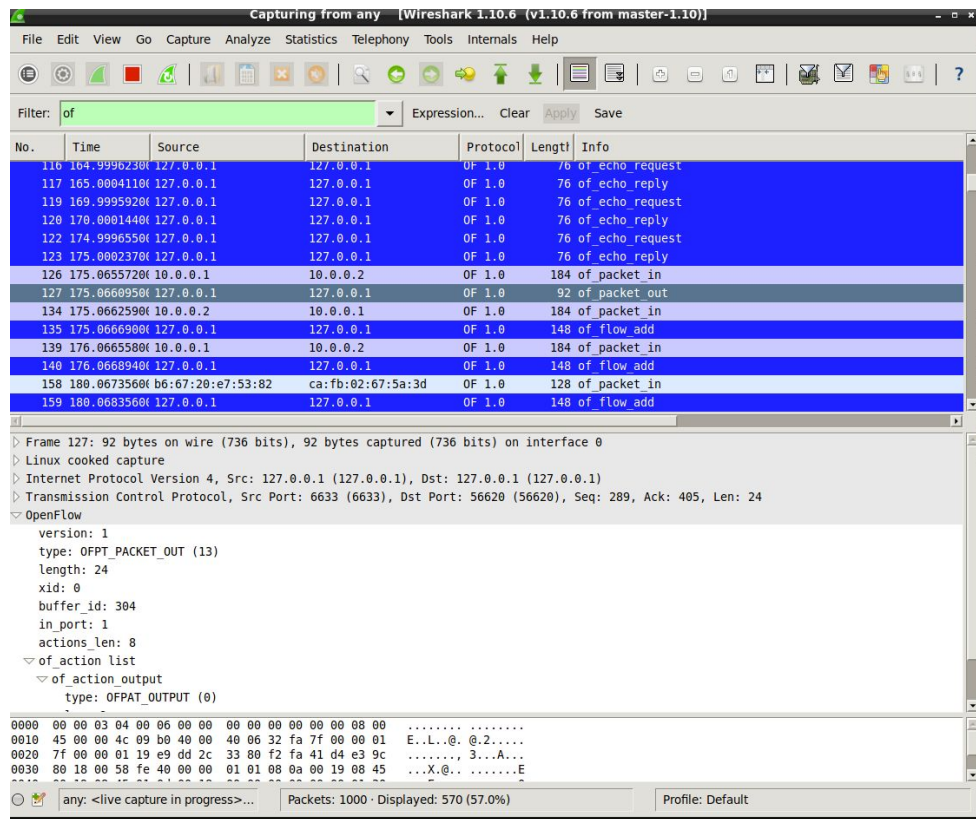


- i. Source and destination of `of_packet_in`

Source	Destination
10.0.0.1	10.0.0.2
10.0.0.2	10.0.0.1
10.0.0.1	10.0.0.2
b6:67:20:e7:53:82	ca:fb:02:67:5a:3d
ca:fb:02:67:5a:3d	b6:67:20:e7:53:82

- ii. Find another packet that matches the “of” filter with the OpenFlow typefield set to OFPT_PACKET_OUT. What is the source and destination IP addresses for this entry? Take screenshots of your result.

Source	Destination
127.0.0.1	127.0.0.1



- c. Replace the display filter for “of” to “icmp && not of”. Run pingall again, how many entries are generated in wireshark? What type of icmp entries show up? Take a screenshot of your results.
 - i. There are 48 entries
 - ii. The types of entries are request and reply

Wireshark 2.10.0 (121200 from master 2100)

File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help

Filter: icmp && not of Expression... Clear Apply Save

No.	Time	Source	Destination	Protocol	Length	Info
19	0.012075000	10.0.0.3	10.0.0.2	ICMP	100	Echo (ping) reply id=0x14a6, seq=1/256, ttl=64 (request in 18)
20	0.012196000	10.0.0.3	10.0.0.2	ICMP	100	Echo (ping) reply id=0x14a6, seq=1/256, ttl=64
21	0.014511000	10.0.0.2	10.0.0.4	ICMP	100	Echo (ping) request id=0x14a7, seq=1/256, ttl=64
22	0.014651000	10.0.0.2	10.0.0.4	ICMP	100	Echo (ping) request id=0x14a7, seq=1/256, ttl=64 (reply in 23)
23	0.014666000	10.0.0.4	10.0.0.2	ICMP	100	Echo (ping) reply id=0x14a7, seq=1/256, ttl=64 (request in 22)
24	0.014801000	10.0.0.4	10.0.0.2	ICMP	100	Echo (ping) reply id=0x14a7, seq=1/256, ttl=64
25	0.017276000	10.0.0.3	10.0.0.1	ICMP	100	Echo (ping) request id=0x14a8, seq=1/256, ttl=64
26	0.017415000	10.0.0.3	10.0.0.1	ICMP	100	Echo (ping) request id=0x14a8, seq=1/256, ttl=64 (reply in 27)
27	0.017431000	10.0.0.1	10.0.0.3	ICMP	100	Echo (ping) reply id=0x14a8, seq=1/256, ttl=64 (request in 26)
28	0.017566000	10.0.0.1	10.0.0.3	ICMP	100	Echo (ping) reply id=0x14a8, seq=1/256, ttl=64
29	0.019055000	10.0.0.3	10.0.0.2	ICMP	100	Echo (ping) request id=0x14a9, seq=1/256, ttl=64
30	0.019337000	10.0.0.3	10.0.0.2	ICMP	100	Echo (ping) request id=0x14a9, seq=1/256, ttl=64 (reply in 31)
31	0.019946000	10.0.0.2	10.0.0.3	ICMP	100	Echo (ping) reply id=0x14a9, seq=1/256, ttl=64 (request in 30)
32	0.020118000	10.0.0.2	10.0.0.3	ICMP	100	Echo (ping) reply id=0x14a9, seq=1/256, ttl=64
33	0.027359000	10.0.0.3	10.0.0.4	ICMP	100	Echo (ping) request id=0x14aa, seq=1/256, ttl=64
34	0.027650000	10.0.0.3	10.0.0.4	ICMP	100	Echo (ping) request id=0x14aa, seq=1/256, ttl=64 (reply in 35)
35	0.027724000	10.0.0.4	10.0.0.3	ICMP	100	Echo (ping) reply id=0x14aa, seq=1/256, ttl=64 (request in 34)
36	0.028094000	10.0.0.4	10.0.0.3	ICMP	100	Echo (ping) reply id=0x14aa, seq=1/256, ttl=64
37	0.030808000	10.0.0.4	10.0.0.1	ICMP	100	Echo (ping) request id=0x14ab, seq=1/256, ttl=64
38	0.030899000	10.0.0.4	10.0.0.1	ICMP	100	Echo (ping) request id=0x14ab, seq=1/256, ttl=64 (reply in 39)
39	0.030898000	10.0.0.1	10.0.0.4	ICMP	100	Echo (ping) reply id=0x14ab, seq=1/256, ttl=64 (request in 38)
40	0.031031000	10.0.0.1	10.0.0.4	ICMP	100	Echo (ping) reply id=0x14ab, seq=1/256, ttl=64
41	0.032557000	10.0.0.4	10.0.0.2	ICMP	100	Echo (ping) request id=0x14ac, seq=1/256, ttl=64
42	0.032634000	10.0.0.4	10.0.0.2	ICMP	100	Echo (ping) request id=0x14ac, seq=1/256, ttl=64 (reply in 43)
43	0.032644000	10.0.0.2	10.0.0.4	ICMP	100	Echo (ping) reply id=0x14ac, seq=1/256, ttl=64 (request in 42)
44	0.032680000	10.0.0.2	10.0.0.4	ICMP	100	Echo (ping) reply id=0x14ac, seq=1/256, ttl=64
45	0.033962000	10.0.0.4	10.0.0.3	ICMP	100	Echo (ping) request id=0x14ad, seq=1/256, ttl=64
46	0.034024000	10.0.0.4	10.0.0.3	ICMP	100	Echo (ping) request id=0x14ad, seq=1/256, ttl=64 (reply in 47)
47	0.034032000	10.0.0.3	10.0.0.4	ICMP	100	Echo (ping) reply id=0x14ad, seq=1/256, ttl=64 (request in 46)
48	0.034122000	10.0.0.3	10.0.0.4	ICMP	100	Echo (ping) reply id=0x14ad, seq=1/256, ttl=64

Frame 1: 100 bytes on wire (800 bits), 100 bytes captured (800 bits) on interface 0

- Linux cooked capture
- Internet Protocol Version 4, Src: 10.0.0.1 (10.0.0.1), Dst: 10.0.0.2 (10.0.0.2)
- Internet Control Message Protocol

0000 00 03 00 01 00 06 ca fb 02 67 5a 3d 00 00 08 00gZ.....
0010 45 00 00 54 1d 40 40 00 40 01 09 67 0a 00 00 01 E..T..@. @.g....
0020 0a 00 00 02 08 00 92 a7 14 a2 00 01 8e db a7 5d
0030 2d 79 02 00 09 0a 0b 0c 0d 0e 0f 10 11 12 13 y.....

Frame (frame), 100 bytes Packets: 51 · Displayed: 48 (94.1%) · Dropped: 0 (0.0%) Profile: Default