

POLITECNICO DI TORINO



**Politecnico  
di Torino**

# **Linux and mininet command reference**

“Computer network design and control” module of  
Communication and network systems

Academic year 2023/24

Andrea Bianco, Paolo Giaccone,  
Alessandro Cornacchia, Francesco Malandrino, Matteo Sacchetto

Version: November 22, 2023  
©2023-24

# Contents

Commands	3
Notation	3
Mininet commands	3
Overview	3
General	4
nodes	4
links	4
net	4
help	4
xterm	4
Network	4
ping/pingall	4
iperf [NOT RECOMMENDED - See iperf3 instead]	5
Linux commands	5
Overview	5
General	6
ls	6
cd	6
cat	6
gedit	6
man	7
stress-ng	7
Network	7
ifconfig	7
ping	8
iperf3	8
ovs-ofctl	9
add-flow	9
dump-flows	10
del-flows	10
Usage in mininet	10
tshark	10

# Commands

This document aims to be a simple guide on some of the commands you will need to use for the **Labs on SDN**

## Notation

Some general guidelines about the notation used in the next sections:

- [ ... ]: squared brackets are used to specify **optional** parameters
- < ... >: angular brackets are used to specify **required** parameters

## Mininet commands

The following commands are commands available directly from the **mininet**. All of these commands need to be run within the **mininet cli** and allow you to perform various tasks  
Ex.

```
mininet> net
```

All of the commands can be run also on a specific node by prefixing the command with the name of the node  
Ex.

```
mininet> h1 ping h2
```

## Overview

### General

- `nodes`: to show all the nodes in the current network topology (hosts, switches, ...)
- `links`: to show the connections between nodes
- `net`: to show a general overview of the network topology (it shows `nodes` and `links`)
- `help`: to show the help documentation of the given command
- `xterm`: open a terminal on the given node/nodes

### Network

- `ping`
- `iperf`

## General

### nodes

Displays network nodes (hosts, switches,...)

```
mininet> nodes
```

### links

Displays network links

```
mininet> links
```

### net

Displays the network topology (nodes and links)

```
mininet> net
```

### help

Displays documentation related to the mininet commands

```
mininet> help [command]
```

If command is omitted it will show a general help, displaying all available commands plus additional documentation

### xterm

Allows you to open a terminal on the given node/nodes

```
mininet> xterm [nodes...]
```

## Network

### ping/pingall

Allows to pings a series of hosts or all of them

```
mininet> ping <hosts...>
```

or

```
mininet> pingall
```

See documentation about [ping](#) in the Linux section

## iperf [NOT RECOMMENDED - See [iperf3](#) instead]

Allows you to test the bandwidth between a pair of hosts

```
mininet> iperf <host1> <host2>
```

While mininet provides `iperf` within the list of available commands, we do not recommend to use it, due to presence of some bugs. To test the bandwidth between hosts we suggest to open one `xterm` for each hosts and manually run [iperf3](#).

## Linux commands

The following commands are commands available within a **linux shell**. These commands can be run directly from a linux terminal or can be run from within the mininet CLI.

To run it from within the mininet CLI we need to tell mininet to **invoke the command from within a linux shell**. To do so we can simply add **sh** before each of the following commands  
Ex.

```
mininet> sh man iperf3
```

## Overview

### General

- `ls`: list the content of a directory
- `cd`: change directory
- `cat`: print the content of a file
- `gedit`: open the file editor
- `sudo`: run a command as admin
- `man`: open the manual for a given command (press `q` to exit from man)

### Network

- `ifconfig`: to show the configuration of the network interfaces (`ip -s addr`: equivalent of `ifconfig` using the `ip` command)
- `ping`: ping the device with the specified IP address
- `iperf`: bandwidth measurement tool
- `ovs-ofctl`: (Open Virtual Switch - OpenFlow Control) configure network switches
- `tshark`: dump and analyze network traffic

## General

### ls

list the content of a directory

```
$ ls [flags...] [files or dirs...]
```

### Useful flags

- `-l`: show additional information (permissions, owner, group, size, ...)
- `-a`: show hidden files
- `-h`: show human readable sizes (KB, MB, GB,...)
- `-t`: sort them based on their last modified timestamp (from newer to older)
- `-r`: reverse order

If you do not specify any file or dir it will list the content of the **current directory** (a.k.a. `.`)

### cd

Change directory to the specified one

```
$ cd [dir]
```

If the directory is not specified, it will go to the current user home directory (`/home/$USER`, a.k.a. `~`). To go to the parent directory you can use the `..` directory

Ex.

```
$ cd ..
```

In linux the directory separator character is the forward slash `/`.

Ex.

```
$ cd dir/subdir
```

### cat

Print the content of a file

```
$ cat <file>
```

### gedit

Open the text editor

```
$ gedit <file>
```

## man

Show the manual for a given command

```
$ man <cmd>
```

To navigate the manual you can use the following keys:

- ↑↓: to go up and down the manual
- /: to forward search for a specific word
- ?: to backward search for a specific word
- n: to go to the next search result
- q: to exit the manual
- h: to open the help section

## stress-ng

Stress various parts of a system (CPU, RAM, disk, ...)

```
$ stress-ng [options...]
```

### Useful flags

- `-c`, `--cpu <N>`: stress the CPU with N workers. The cpu stress method can also be specified with `--cpu-method <method>`)
- `--cpu-method <method>`: specify the cpu stress method. By default all stress methods are run sequentially, but is also possible to select a specific one. For a list of all available cpu methods refer to `stress-ng` manual
- `--io <N>`: I/O disk
- `-t`, `--timeout <T>`: the stress test will last T seconds. If not specified it will run until stopped with CTRL+C.
- `-m`, `--vm <N>`: stress the RAM with N workers

## Network

### ifconfig

Show the configuration of the network interfaces

```
$ ifconfig [interface]
```

The simple usage, without any parameter, will list the interfaces and show the configuration of all interfaces.

If you want to see the details of a given interface you can specify the name of the interface you are interested in.

### SIDE NOTE:

`ifconfig` is a program belonging to the `net-tools` package. While it is still useful today, its use has been deprecated in Linux in favor of the more modern `ip` command. In case you end up working with Linux be aware of that!

To replicate the above output using the `ip` command you can use the following command

```
$ ip -s addr
```

### **ping**

Simple utility to send ICMP requests to network host

```
$ ping [flags...] <host>
```

The most useful flags are

- `-i <seconds>`: to set an interval in seconds (values below 0.2 require you to become root (using `sudo`))
- `-c <count>`: to set a given number of packets to send
- `-D`: to print timestamps
- `-s <bytes>`: to set the packet size
- `-t <ttl>`: to set the TTL (time to live)

### **iperf3**

It is the new version of `iperf` with some improvements and bug fixes. It is a client-server program, so in order to use it you need one host as server and one as client.

#### **Server**

```
$ iperf3 -s
```

#### **Client**

```
$ iperf3 -c [flags...] <host>
```

The most useful flags are

- `-u`: use UDP rather than TCP
- `-b <kmgKMG | pps>`: bandwidth to send in bits/s or multiples
- `-l <size [KM]>`: size of the packet in bytes or multiples
- `-t <duration>`: duration in seconds



## ovs-ofctl

Open Virtual Switch - OpenFlow Control CLI. Through this CLI you can configure the OpenFlow switches which are created by mininet.

```
$ ovs-ofctl [sub-command] [options...]
```

The sub-commands which we will use in the labs are the following

- `add-flow`: this allows you to add a new rule to the OpenFlow switch
- `dump-flows`: show the current configured rules for the given OpenFlow switch
- `del-flows`: remove the selected rules from the OpenFlow switch

### add-flow

```
$ ovs-ofctl add-flow <switch> <matcher>,<action>
```

With this command we configure the switch to perform the specified action for all the traffic which matches a specific matcher.

#### Useful matchers:

- `in_port=<switch_port_number>`: **port based matcher** which configures the switch to perform the given action for all the traffic which it receives from the specified port
- `dl_type=<type>,nw_dest=<ip/netmask>`: **destination IP based matcher** which configures the switch to perform the given action for all traffic of the selected type destined to the specified IP
  - Standard values of `<type>` are:
    - \* 0x0800: IPv4
    - \* 0x86dd: IPv6
  - We will use only IPv4
- `dl_type=0x0800,nw_proto=<proto>,nw_dest=<ip/netmask>`: **destination IP + protocol based matcher** which configures the switch to perform the given action for all traffic of the selected type destined to the specified IP
  - Standard values of `<proto>` are:
    - \* 1: for ICMP
    - \* 6: for TCP
    - \* 17: for UDP
- `<shorthand>,nw_dest=<ip/netmask>`: same as the two above, but we are using shorthand notations
  - Some of the shorthands are:
    - \* `ip`: equivalent to `dl_type=0x0800`
    - \* `icmp`: equivalent to `dl_type=0x0800,nw_proto=1`

- \* tcp: equivalent to `dl_type=0x0800,nw_proto=6`
- \* udp: equivalent to `dl_type=0x0800,nw_proto=17`

### Action

The action usually involves routing the traffic through a given port of the selected switch. The syntax is `actions=output:<port>`, where `port` is the selected port number of the switch.

### dump-flows

```
$ ovs-ofctl dump-flows <switch>
```

This allows you to see the rules which have been configured for the selected switch. The switch name is the one configured by mininet (ex. `s1`).

### del-flows

```
$ ovs-ofctl del-flows <switch> [matcher]
```

This allows you to delete flows based on the specified matcher. The matchers you can use are the same already explained in section [add-flow](#). If no matcher is specified it will delete all rules configured on the specified switch

### Usage in mininet

It is recommended to use this CLI through mininet CLI. To do so, simply add `sh` before each `ovs-ofctl` command.

Ex.

```
mininet> sh ovs-ofctl dump-flows s1
```

### tshark

Is a network protocol analyzer

```
$ tshark [flags..]
```

The most useful flags are

- `-i <interface>`: select the capture interface (ex. `s1-eth1`)
  - The interface name is the one you obtain with [ifconfig](#)
- `-q`: don't display the continuous count of packets captured, just display, at the end of the capture, a count of packets captured
- `-z <options...>`: collect and show statistics. Some of the most useful options are
  - `ip_srcdst,tree`: calculate statistics on IPv4 addresses, with source and destination addresses all grouped together.
  - `conv,<type>`: Create a table that lists all conversations that could be seen in the capture. The most useful types are
    - \* `eth`: ethernet addresses
    - \* `ip`: ip addresses
    - \* `tcp`: TCP/IP socket pairs Both IPv4 and IPv6 are supported
    - \* `udp`: UDP/IP socket pairs Both IPv4 and IPv6 are supported