Download/Get Started with Mininet

The easiest way to get started is to **download a pre-packaged Mininet/Ubuntu VM**. This VM includes Mininet itself, all OpenFlow binaries and tools pre-installed, and tweaks to the kernel configuration to support larger Mininet networks.

Option 1: Mininet VM Installation (easy, recommended)

VM installation is the **easiest and most foolproof** way of installing Mininet, so it's what we recommend to start with.
Follow these steps for a VM install:

1. Download a **Mininet VM Image** from Mininet Releases.
2. Download and install a **virtualization system**. We recommend one of the following free options:
   - VirtualBox (GPL, macOS/Windows/Linux)
   - VMware Fusion (macOS)
   - VMware Workstation Player (Windows/Linux)

   You can also use any of:

   - Qemu (free, GPL) for any platform
   - Microsoft Hyper-V (Windows)
   - KVM (free, GPL, Linux)
3. *Optional, but recommended!* Sign up for the mininet-discuss mailing list. This is the source for Mininet **support** and discussion with the friendly Mininet community. ;-) (And don't forget the FAQ and documentation.)
4. Run through the VM Setup Notes to log in to the VM and customize it as desired.
5. Follow the Walkthrough to get familiar with Mininet commands and typical usage.

(In addition to the above resources, we've prepared a helpful Mininet FAQ as well as Documentation which you can refer to at any time! We recommend consulting them first if you have any questions.)
Once you've completed the Walkthrough, you should have a clear idea for what Mininet is and what you might use it for. The Introduction to Mininet explains the basics of Mininet's Python API. If you are interested in OpenFlow and Software-Defined Networking, you may wish to complete the OpenFlow tutorial as well. Good luck, and have fun!

Option 2: Native Installation from Source

This option works well for local VM, remote EC2, and native installation. It assumes the starting point of a fresh Ubuntu, Debian, or (experimentally) Fedora installation.

We strongly recommend more recent Ubuntu or Debian releases, because they include newer versions of Open vSwitch. (Fedora also includes recent OvS releases.)

To install natively from source, first you need to get the source code:

```
git clone git://github.com/mininet/mininet
```

Note that the above `git` command will check out the latest and greatest Mininet (which we recommend!) If you want to run the last tagged/released version of Mininet - or any other version - you may check that version out explicitly:

```
cd mininet
git tag  # list available versions
git checkout -b mininet-2.3.0 2.3.0  # or whatever version you wish to install
cd ..
```

Once you have the source tree, the command to install Mininet is:

```
mininet/util/install.sh [options]
```

Typical `install.sh` options include:

- `-a`: install everything that is included in the Mininet VM, including dependencies like Open vSwitch as well the additions like the OpenFlow wireshark dissector and POX. By default these tools will be built in directories created in your home directory.
- `-nfv`: install Mininet, the OpenFlow reference switch, and Open vSwitch
- `-s mydir`: use this option before other options to place source/build trees in a specified directory rather than in your home directory.

So, you will probably wish to use one (and only one) of the following commands:

```
To install everything (using your home directory): install.sh -a
To install everything (using another directory for build): install.sh -s mydir
-a
To install Mininet + user switch + OvS (using your home dir): install.sh -nfv
To install Mininet + user switch + OvS (using another dir:) install.sh -s
mydir -nfv
```

You can find out about other useful options (e.g. installing the OpenFlow wireshark dissector, if it's not already included in your version of wireshark) using

```
install.sh -h
```

After the installation has completed, test the basic Mininet functionality:

```
sudo mn --switch ovsbr --test pingall
```

Then continue with steps 3-5, above. If you run into errors, first consult the FAQ, Documentation, and mailing list archives to see if anything resembling your problem has been seen before and if there might be a possible solution. If those things don't help and you still have problems that you cannot solve on your own (or with some help from Google :)), you can request help on the friendly mininet-discuss mailing list.

Option 3: Installation from Packages

If you're running a recent Ubuntu release, or Debian 11+, you can install the Mininet packages. Note that this may give you an older version of Mininet, but it can be a very convenient way to get started.

To confirm which OS version you are running, run the command

```
lsb_release -a
```

Next, install the base Mininet package by entering **only one** of the following commands, corresponding to the distribution you are running:

```
Mininet 2.3.0 on Debian 11: sudo apt-get install mininet
Mininet 2.2.2 on Ubuntu 20.04 LTS: sudo apt-get install mininet
Mininet 2.2.2 on Ubuntu 18.04 LTS: sudo apt-get install mininet
```

If it's not obvious which Mininet version you have, you can try:

```
mn --version
```

Mininet supports multiple switches and OpenFlow controllers. For this test, we will use Open vSwitch in bridge/standalone mode.

To test this, try:

```
sudo mn --switch ovsbr --test pingall
```

If Mininet complains that Open vSwitch isn't working, make sure it is installed and running:

```
sudo apt-get install openvswitch-switch
sudo service openvswitch-switch start
```

If you wish to go through the Mininet walkthrough, you will want to install additional software. The following commands

```
git clone git://github.com/mininet/mininet
mininet/util/install.sh -fw
```

will install the OpenFlow reference switch, reference controller and Wireshark dissector.

Option 4. Upgrading an *existing* Mininet Installation

There are many ways to do this. If you haven't made any changes to Mininet, you can usually:

1. Check out the Mininet code, if you don't have it already:

```
2. git clone https://github.com/mininet/mininet
```

3. Remove old Mininet packages, if any:

```
4. sudo apt-get uninstall mininet        # if you have installed a Mininet apt
   package
5.
6. sudo pip uninstall mininet               # if you are upgrading an older
   Mininet VM
7.                                       #  where  Mininet  was  installed  with
   setuptools
```

8. Install the new Mininet version:

```
9. cd mininet
10.git fetch  # to fetch the latest and greatest branches and tags
11.git tag    # to see what versions are available
12.
13.git checkout -b mininet-2.3.0 2.3.0  # or whatever version/branch you want,
   or
14.                                      # master if you want the latest
15.
16.sudo make install   # only install new mnexec and mininet packages
```

Note that `sudo make install` only installs `mnexec` and the Mininet packages. If you wish to install Mininet and its dependencies, do this:

```
sudo apt-get update   # make sure apt works
util/install.sh -n    # install mininet and dependencies
```

If you wish to specify a specific Python version, you can do so:

```
sudo PYTHON=python3 make install
```

or

```
PYTHON=python3 util/install.sh -a
```

As an alternative to `sudo make install` you can also do `sudo make develop`, which will create symbolic links from `/usr/lib/python...` to your source tree.

Note that this will only upgrade Mininet itself - any other components such as Open vSwitch, etc. can be upgraded separately as desired.

Important Note: Python 2 and Python 3 Mininet

Mininet 2.3.0 and greater support Python 3 as well as Python 2.

The official Mininet VM images include Python 2 and Python 3 Mininet.

When you install from source, you can choose which version to install. For example:

```
sudo PYTHON=python2 mininet/util/install.sh -n   # install Python 2 Mininet
sudo PYTHON=python3 mininet/util/install.sh -n   # install Python 3 Mininet
```

You can install both versions side by side (as is done in the Mininet VM images) and pick which Python version to use for `mn`:

```
sudo python2 `which mn` ...
sudo python3 `which mn` ...
```

To find out what Python version that `mn` uses by default:

```
echo py sys.version | sudo mn -v output
```

Note that Python 3 is (unfortunately!) not generally backward compatible with legacy Python 2 code, so older Python 2 Mininet scripts may require some changes to work with Python 3. Since Python 2 is no longer officially supported by the CPython project and most OS releases, we recommend migrating to Python 3 (however painful that may be.)

If you run into Python 3 incompatibilities in Mininet, please file an issue on GitHub.