CSCD58 Computer Networks
MiniEdit: Mininet's GUI

Marcelo Ponce

Fall 2025

Department of Computer and Mathematical Sciences - UTSC

Today's Tutorial

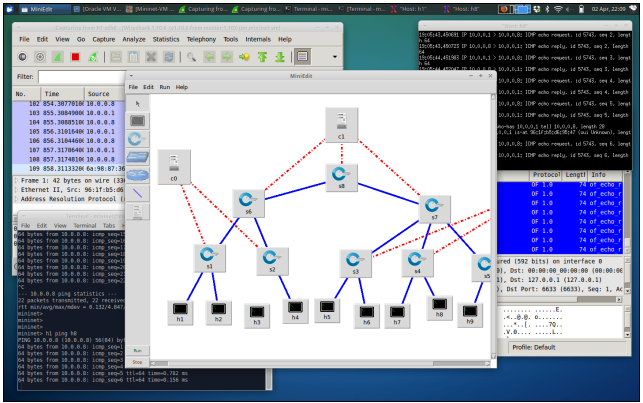
MiniEdit: Mininet's graphical user interface

Lab. Report

MiniEdit: Mininet's graphical user interface

MiniEdit

- the Mininet repository, `https://github.com/mininet/mininet` includes a tool "MiniEdit" to act as a simple GUI editor for Mininet.
- MiniEdit is an experimental tool, i.e. a python script offering a graphical user interface



MiniEdit – setup

- We will use the same VM as you used for the SimpleRouter assignment
- but we need to have a way to display the graphical interface
- We also need to clone the actual mininet's github repository

Requirements

- Mininet // VM + repo
- Python, tkinter
- a way to display a graphical output
  e.g. X-forwarding, VNC or a combination of these depending on how are you working (i.e. in lab machines)

MiniEdit – setup i

```
# let's start by cloning the mininet repo
git clone https://github.com/mininet/mininet
```

```
mininet@mininet-vm:~$ tree -d mininet/
mininet/
├── bin
├── custom
├── debian
│   └── source
├── doc
├── examples
│   └── test
├── mininet
│   ├── examples -> ../examples
│   └── test
└── util
```

MiniEdit – setup ii

```
mininet@mininet-vm:~$ ls mininet/examples/
baresshd.py       consoles.py      __init__.py         multilink.py
    popenpoll.py      test
bind.py           controllers2.py  intfoptions.py      multiping.py      popen
    .py           tree1024.py
clustercli.py     controllers.py   limit.py            multipoll.py
    README.md         treeping64.py
clusterdemo.py    controlnet.py    linearbandwidth.py  multitest.py
    scratchnet.py     vlanhost.py
clusterperf.py    cpu.py           linuxrouter.py      natnet.py
    scratchnetuser.py
cluster.py        emptynet.py      miniedit.py         nat.py
    simpleperf.py
clusterSanity.py  hwintf.py        mobility.py         numberedports.py  sshd.
    py
```

MiniEdit – setup iii

SRC:

`https://github.com/mininet/mininet/blob/master/examples/miniedit.py`

Launching miniedit i

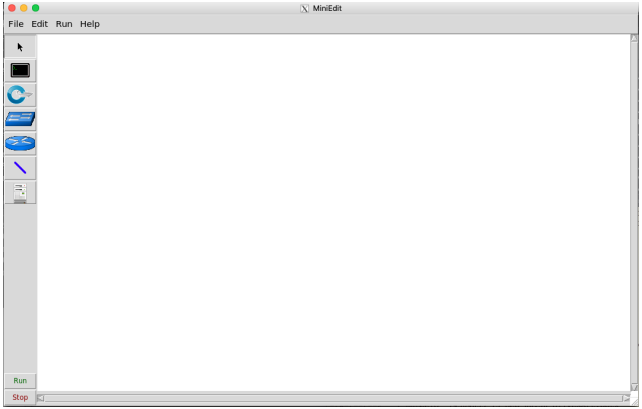
On the mininet VM:

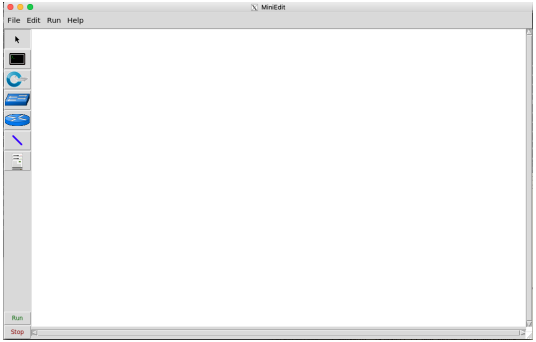
```
mininet@mininet-vm:~$ python mininet/examples/miniedit.py

# this should also work
$ sudo ~/mininet/examples/miniedit.py
```

Using the SimpleRouter modified VM:

```
python /usr/local/lib/python3.8/dist-packages/mininet/examples/miniedit.py
```

Launching miniedit ii





Select tool typically used for the movement of nodes

Host tool creates end-host nodes

Switch tool creates an OpenFlow-enabled switch

Legacy Switch tool creates a traditional learning Ethernet switch

Legacy Router tool creates a basic router that will operate indep., without a controller

NetLink tool creates links between nodes

Controller tool creates a controller

Run/Stop

menues... Open/Save/Export Level 2 Script

 The Select tool is used to move nodes around on the canvas.
Click and drag any existing node. Interestingly the Select tool is not needed to select a node or link on the canvas. To select an existing node or link, just hover the mouse pointer over it — this works regardless of the tool that is currently active — and then either right-click to reveal a configuration menu for the selected element or press the Delete key to remove the selected element.

 The Host tool creates nodes on the canvas that will perform the function of host computers.
Click on the tool, then click anywhere on the canvas you wish to place a node. As long as the tool remains selected, you can keep adding hosts by clicking anywhere on the canvas. The user may configure each host by right-clicking on it and choosing Properties from the menu.

MiniEdit Tools ii

 The Switch tool creates OpenFlow-enabled switches on the canvas.
These switches are expected to be connected to a controller. The tool operates the same way as the Hosts tool above. The user may configure each switch by right-clicking on it an choosing Properties from the menu.

 The Legacy Switch tool creates a learning Ethernet switch with default settings. The switch will operate independently, without a controller. The legacy switch cannot be configured and is set up with Spanning Tree disabled, so do not connect legacy switches in a loop.

 The Legacy Router tool creates a basic router that will operate independently, without a controller.
It is basically just a host with IP Forwarding enabled. The legacy router cannot be configured from the MiniEdit GUI.

MiniEdit Tools iii

 The NetLink tool creates links between nodes on the canvas.
Create links by selecting the NetLink tool, then clicking on one node and dragging the link to the target node. The user may configure the properties of each link by right-clicking on it an choosing Properties from the menu.

 The Controller tool creates a controller.
Multiple controllers can be added. By default, the MiniEdit creates a mininet openFlow referencecontroller, which implements the behavior of a learning switch. Other controller types can be configured. The user may configure the properties of each controller by right-clicking on it an choosing Properties from the menu.

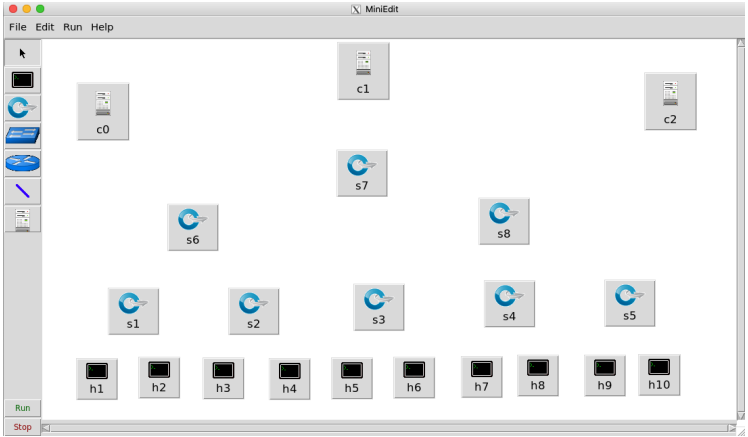
MiniEdit Tools iv

The Run starts Mininet simulation scenario currently displayed in the MiniEdit canvas.

The Stop button stops it. When MininEdit simulation is in the “Run” state, right-clicking on network elements reveals operational functiosn such as opening a terminal window, viewing switch configuration, or setting the status of a link to “up” or “down”.

Create a custom network topology... i

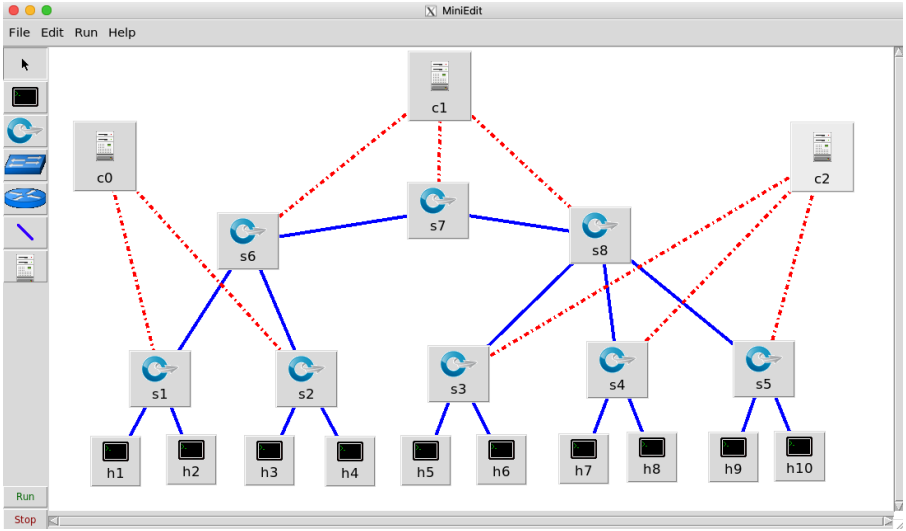
Let's try to reproduce the following network



- 3 controllers
- 8 switches
- 10 hosts

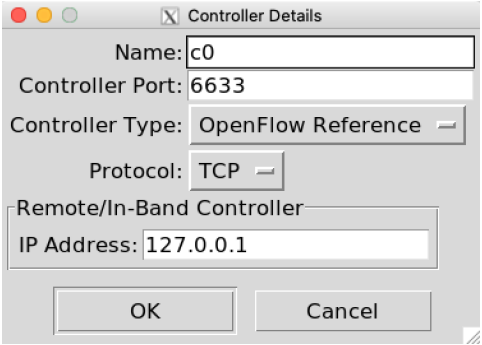
Create a custom network topology... ii

Next, let's connect the network using the "Netlink tool",



Configuring the Network... i

- There are three controllers in the example we just built. For this simple case, we will use the default OpenFlow Reference controller that comes built into Mininet. However, we need to configure each controller so it uses a different port.
- Right-click on each controller and select Properties from the menu that appears. The default port number for each controller is 6633. Change this so the port numbers used by controllers c0, c1, and c2 are 6633, 6634, and 6635, respectively.



Configuring the Network... ii

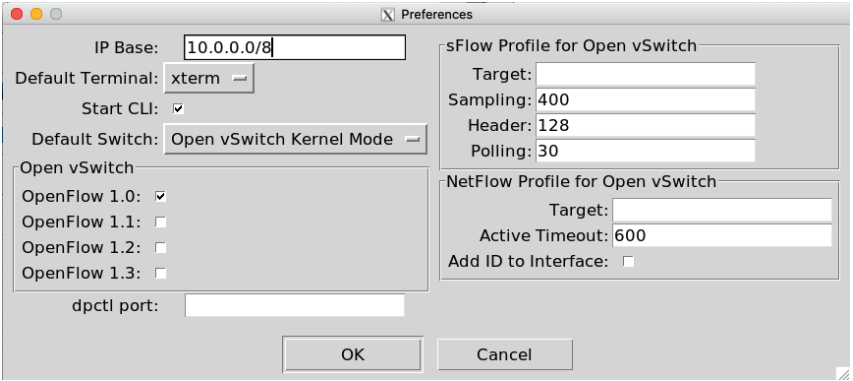
Similarly this can be done for the other elements in the network...

Take a few minutes to explore these...

MiniEdit Preferences

By default, the MiniEdit console window does not give the user access to the Mininet command line interface.

If you want to be able to use the Mininet CLI when a simulation is running, check the Start CLI box.



Saving the configuration

- The MiniEdit Preferences are saved in the MiniEdit topology file for each scenario so you may have different preferences for each saved scenario.

Save topology
To save the Mininet Topology (*.mn) file, click on File in the top menu bar and select Save from the drop-down menu. Type in the file name and save the file.

Save custom Mininet script
To save the Mininet Custom Topology (*.py) file, click on File in the top menu bar and select Save Level 2 Script from the drop-down menu. Type in the file name and save the file.

Running the network configuration i

First thing to recall is that mininet has to be run with `sudo`.

But if you try to do so using X11-forwarding mist likely you will run into this problem:

```
X11 connection rejected because of wrong authentication.
```

The problem is that X-forwarding was established as a normal user but not fot a sudoer one!

The solution is described in the troubleshoorting section from A3:

```
# request magic cookie
# xxxxxyyyyyxxxxxxzzzzz should be something else...
$ xauth list $DISPLAY
mininet-vm.unix:10 MIT-MAGIC-COOKIE-1 xxxxxyyyyyxxxxxxzzzzz

# become sudo usr
```

Running the network configuration ii

```
$ sudo -s

# add the magic cookie to the 'sudo' usr authority file
$ xauth add mininet-vm/unix:10 MIT-MAGIC-COOKIE-1 xxxxxyyyyyxxxxxxzzzzz
```

```
New Prefs = {'ipBase': '10.0.0.0/8', 'sflow': {'sflowPolling': '30', '
    sflowSampling': '400', 'sflowHeader': '128', 'sflowTarget': ''}, '
    terminalType': 'xterm', 'startCLI': '1', 'switchType': 'ovs', 'netflow': {
    'nflowAddId': '0', 'nflowTarget': '', 'nflowTimeout': '600'}, 'dpctl': '',
     'openFlowVersions': {'ovsOf11': '0', 'ovsOf10': '1', 'ovsOf13': '0', '
    ovsOf12': '0'}}
Getting Hosts and Switches.
Getting controller selection:ref
Getting controller selection:ref
Getting controller selection:ref
```

Running the network configuration iii

```
Getting Links.
*** Configuring hosts
h8 h7 h6 h3 h1 h5 h4 h2 h10 h9
**** Starting 3 controllers
c1 c2 c0
**** Starting 8 switches
s5 s7 s6 s2 s3 s8 s4 s1
No NetFlow targets specified.
No sFlow targets specified.


 NOTE: PLEASE REMEMBER TO EXIT THE CLI BEFORE YOU PRESS THE STOP BUTTON. Not
    exiting will prevent MiniEdit from quitting and will prevent you from
    starting the network again during this session.
```

Running the network configuration iv

```
*** Starting CLI:
mininet>
```

Running "experiments"... i

After starting the simulation scenario, we will view the status of different elements in the network, open terminal windows, run network traffic, run programs on simulated hosts, and simulate network failures.

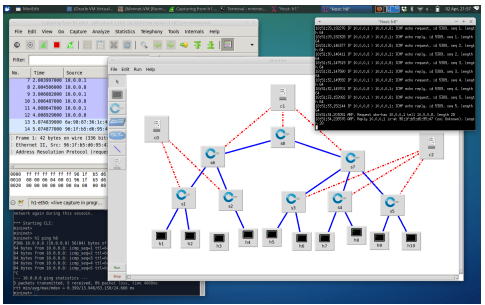
1. View Open vSwitch Configurations
   MiniEdit menu: Run → Show OVS Summary – list of switch configurations
2. Check switch flow tables
   To view the flow tables of some of the switches using the `ovs-ofctl dump-flows` command. We need to run this command on the host computer (or the virtual machine) that is running Mininet. So, we need to use a terminal window connected to the computer (not to one of the nodes in the network simulation). We can use MiniEdit to open an xterm connected to the host computer by using the MiniEdit menu command, Run → Root Terminal

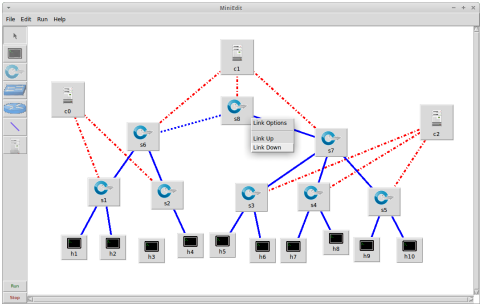
Running "experiments"... ii

3. Generate and monitor Traffic

   Open a xterm in a couple of hosts, right click on the host and select Terminal

   In one, we can launch one monitoring tool, such as, `wireshark` or `tcpdump`, and in a second one run any of the commands that will trigger some actvity, e.g. `ping`, `iperf`, ...





4. Simulate a broken link

   To simulate a broken link in the network, move the mouse pointer over one of the blue links in the network and right-click.

   Choose Link Down from the pulldown menu – the link will turm into a dashed blue line, indicating it is down.

Stopping a run...
First, one must exit the mininet CLI by typing `exit` at the `mininet>` prompt.

Then, press the Stop button on the MiniEdit GUI.

Working with saved mininet configurations

An alternative to running a simulation directly in MiniEdit is to run a Mininet custom topology script created by MiniEdit.
This is the file with the .py extension previously created when we used the menu command: File → Save Level 2 Script.

The script sets up the network topology and the `mininet>` command line prompt appears.

Lab. Report

Lab. Report #04

- Generate a network configuration as the one shown in the tutorial using miniEdit
- Experiment with the different actions discussed in the tutorial: generate and monitor traffic & simulate a broken link
- Capture a screenshot from the miniEdit GUI canvas showing the network you built, miniedit terminal generating some traffic and wireshark capturing in one of the interfaces.
- Submit both a python script and mininet configuration file, for the cases you studied, as well as the captured screeenshot.