Διαχείριση Δικτύων Βασισμένων στο Λογισμικό
9ο εργαστήριο: "RYU Controller & MiniNAM"

ΟΝΟΜΑΤΕΠΩΝΥΜΟ:	Νικόλας Μαυρόπουλος
Α.Μ.:	21865

MiniNAM: A Network Animator for Visualizing Real-Time Packet Flows in Mininet

Visit https://www.ucc.ie/en/misl/research/software/mininam/ and read the manual. Summary:

- Install and start MiniNAM:

sudo git clone https://github.com/uccmisl/MiniNAM

```
eirini@eirini-VirtualBox:~$ sudo git clone https://github.com/uccmisl/MiniNAM
[sudo] password for eirini:
Cloning into 'MiniNAM'...
remote: Enumerating objects: 41, done.
remote: Counting objects: 100% (9/9), done.
remote: Compressing objects: 100% (9/9), done.
remote: Total 41 (delta 3), reused 0 (delta 0), pack-reused 32
Unpacking objects: 100% (41/41), done.
```

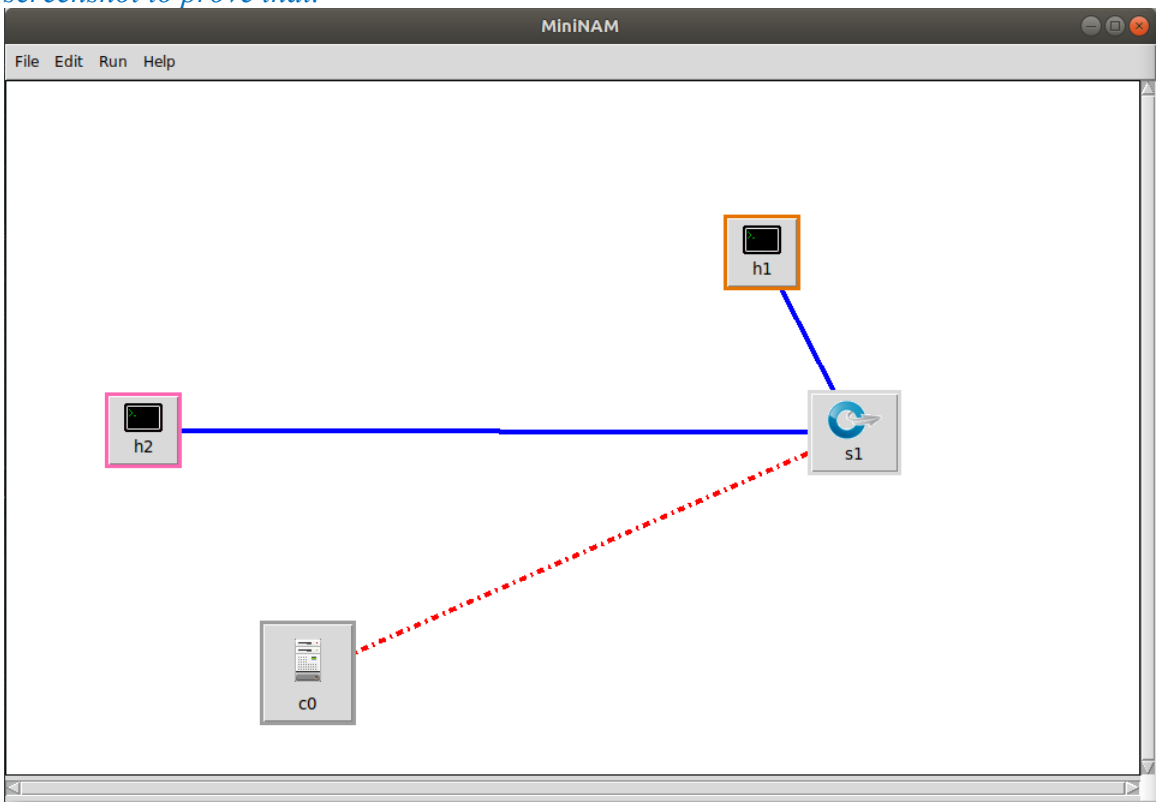
sudo apt-get install python-imaging* OR *sudo apt-get install python-imaging-tk

cd MiniNAM

sudo python MiniNAM.py

```
eirini@eirini-VirtualBox:~/MiniNAM$ sudo python MiniNAM.py
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ...
*** Starting CLI:
mininet>
```

You should now have opened the graphical environment of MiniNAM. Provide a screenshot to prove that.



- Download and install RYU Controller: https://ryu-sdn.org/:

git clone https://github.com/faucetsdn/ryu.git

cd ryu

pip install .

- If needed: ***sudo apt install python-ryu***

Part 1: RYU hub and switch applications

- Visit **https://learning.knetsolutions.in/docs/ryu/#7-ryu-controller---basics** and find the Section "Simple Proactive Hub Application":

Run the hub application in RYU (copy hub.py in ryu/ryu/app) and provide a screenshot of the flows in the switch after the pingall command, showing the FLOOD action.

```
eirini@nikolas:~$ sudo ovs-ofctl -O OpenFlow13 dump-flows s1
 cookie=0x0, duration=39.346s, table=0, n_packets=53, n_bytes=3710, priority=0
actions=FLOOD
```

- Now find the Section “Simple Switch Application (in built)”:

Run the switch application in RYU (cop y l3_switch.py in ryu/ryu/app) and provide a screenshot of the flows in the switch after the ping command. Which are the Match criteria in this case?

```
eirini@nikolas:~$ sudo ovs-ofctl -O OpenFlow13 dump-flows s1
 cookie=0x0, duration=12.291s, table=0, n_packets=11, n_bytes=1078, priority=1
,ip,nw_src=10.0.0.1,nw_dst=10.0.0.2 actions=output:"s1-eth2"
 cookie=0x0, duration=12.290s, table=0, n_packets=11, n_bytes=1078, priority=1
,ip,nw_src=10.0.0.2,nw_dst=10.0.0.1 actions=output:"s1-eth1"
 cookie=0x0, duration=40.728s, table=0, n_packets=14, n_bytes=924, priority=0
actions=CONTROLLER:65535
```

The Match criteria in this case are ip, nw_src=10.0.0.1, nw_dst=10.0.0.2

Do the same for l4_switch.py → Which are the Match criteria in this case?

```
eirini@nikolas:~$ sudo ovs-ofctl -O OpenFlow13 dump-flows s1
 cookie=0x0, duration=10.978s, table=0, n_packets=10, n_bytes=980, priority=1,
icmp,nw_src=10.0.0.1,nw_dst=10.0.0.2 actions=output:"s1-eth2"
 cookie=0x0, duration=10.977s, table=0, n_packets=10, n_bytes=980, priority=1,
icmp,nw_src=10.0.0.2,nw_dst=10.0.0.1 actions=output:"s1-eth1"
 cookie=0x0, duration=32.527s, table=0, n_packets=14, n_bytes=924, priority=0
actions=CONTROLLER:65535
```

The Match criteria in this case are icmp, nw_src=10.0.0.1, nw_dst=10.0.0.2

Finally, run Section “Simple Switch with flow expiry” and show the flows with the timeout expiry.

```
eirini@nikolas:~$ sudo ovs-ofctl -O OpenFlow13 dump-flows s1
[sudo] password for eirini:
 cookie=0x0, duration=11.178s, table=0, n_packets=3, n_bytes=238, idle_timeout=10, hard_timeout=30, priority=1,in_port=
"s1-eth2",dl_src=00:00:00:00:00:02,dl_dst=00:00:00:00:00:01 actions=output:"s1-eth1"
 cookie=0x0, duration=11.177s, table=0, n_packets=3, n_bytes=238, idle_timeout=10, hard_timeout=30, priority=1,in_port=
"s1-eth1",dl_src=00:00:00:00:00:01,dl_dst=00:00:00:00:00:02 actions=output:"s1-eth2"
 cookie=0x0, duration=11.174s, table=0, n_packets=3, n_bytes=238, idle_timeout=10, hard_timeout=30, priority=1,in_port=
"s1-eth3",dl_src=00:00:00:00:00:03,dl_dst=00:00:00:00:00:01 actions=output:"s1-eth1"
 cookie=0x0, duration=11.173s, table=0, n_packets=2, n_bytes=140, idle_timeout=10, hard_timeout=30, priority=1,in_port=
"s1-eth1",dl_src=00:00:00:00:00:01,dl_dst=00:00:00:00:00:03 actions=output:"s1-eth3"
 cookie=0x0, duration=11.170s, table=0, n_packets=3, n_bytes=238, idle_timeout=10, hard_timeout=30, priority=1,in_port=
"s1-eth4",dl_src=00:00:00:00:00:04,dl_dst=00:00:00:00:00:01 actions=output:"s1-eth1"
 cookie=0x0, duration=11.169s, table=0, n_packets=2, n_bytes=140, idle_timeout=10, hard_timeout=30, priority=1,in_port=
"s1-eth1",dl_src=00:00:00:00:00:01,dl_dst=00:00:00:00:00:04 actions=output:"s1-eth4"
 cookie=0x0, duration=11.164s, table=0, n_packets=3, n_bytes=238, idle_timeout=10, hard_timeout=30, priority=1,in_port=
"s1-eth3",dl_src=00:00:00:00:00:03,dl_dst=00:00:00:00:00:02 actions=output:"s1-eth2"
 cookie=0x0, duration=11.163s, table=0, n_packets=2, n_bytes=140, idle_timeout=10, hard_timeout=30, priority=1,in_port=
"s1-eth2",dl_src=00:00:00:00:00:02,dl_dst=00:00:00:00:00:03 actions=output:"s1-eth3"
 cookie=0x0, duration=11.160s, table=0, n_packets=3, n_bytes=238, idle_timeout=10, hard_timeout=30, priority=1,in_port=
"s1-eth4",dl_src=00:00:00:00:00:04,dl_dst=00:00:00:00:00:02 actions=output:"s1-eth2"
 cookie=0x0, duration=11.159s, table=0, n_packets=2, n_bytes=140, idle_timeout=10, hard_timeout=30, priority=1,in_port=
"s1-eth2",dl_src=00:00:00:00:00:02,dl_dst=00:00:00:00:00:04 actions=output:"s1-eth4"
 cookie=0x0, duration=11.153s, table=0, n_packets=3, n_bytes=238, idle_timeout=10, hard_timeout=30, priority=1,in_port=
"s1-eth4",dl_src=00:00:00:00:00:04,dl_dst=00:00:00:00:00:03 actions=output:"s1-eth3"
 cookie=0x0, duration=11.152s, table=0, n_packets=2, n_bytes=140, idle_timeout=10, hard_timeout=30, priority=1,in_port=
"s1-eth3",dl_src=00:00:00:00:00:03,dl_dst=00:00:00:00:00:04 actions=output:"s1-eth4"
 cookie=0x0, duration=17.322s, table=0, n_packets=21, n_bytes=1274, priority=0 actions=CONTROLLER:65535
eirini@nikolas:~$
eirini@nikolas:~$ sudo ovs-ofctl -O OpenFlow13 dump-flows s1
 cookie=0x0, duration=41.326s, table=0, n_packets=25, n_bytes=1554, priority=0 actions=CONTROLLER:65535
```

Part 2: NAT

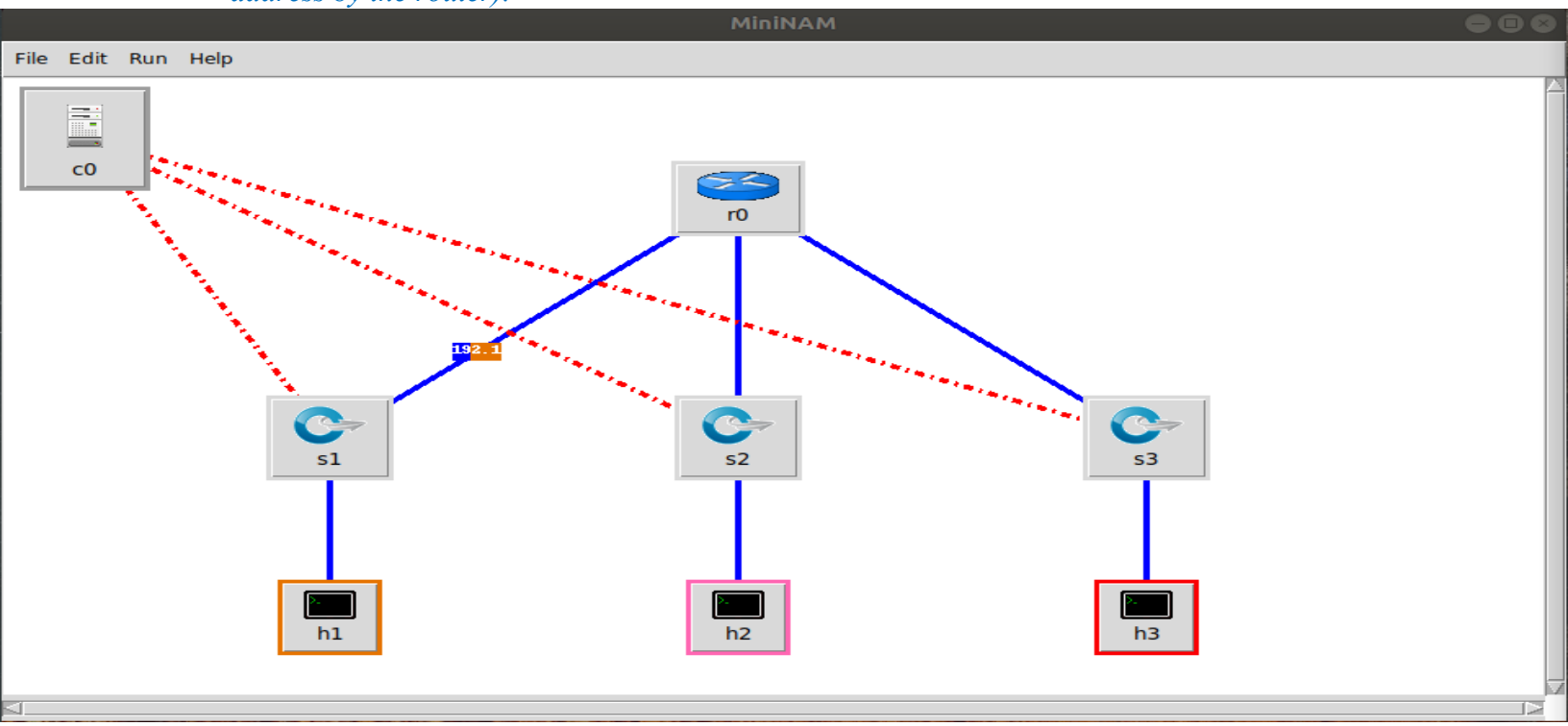
Run:

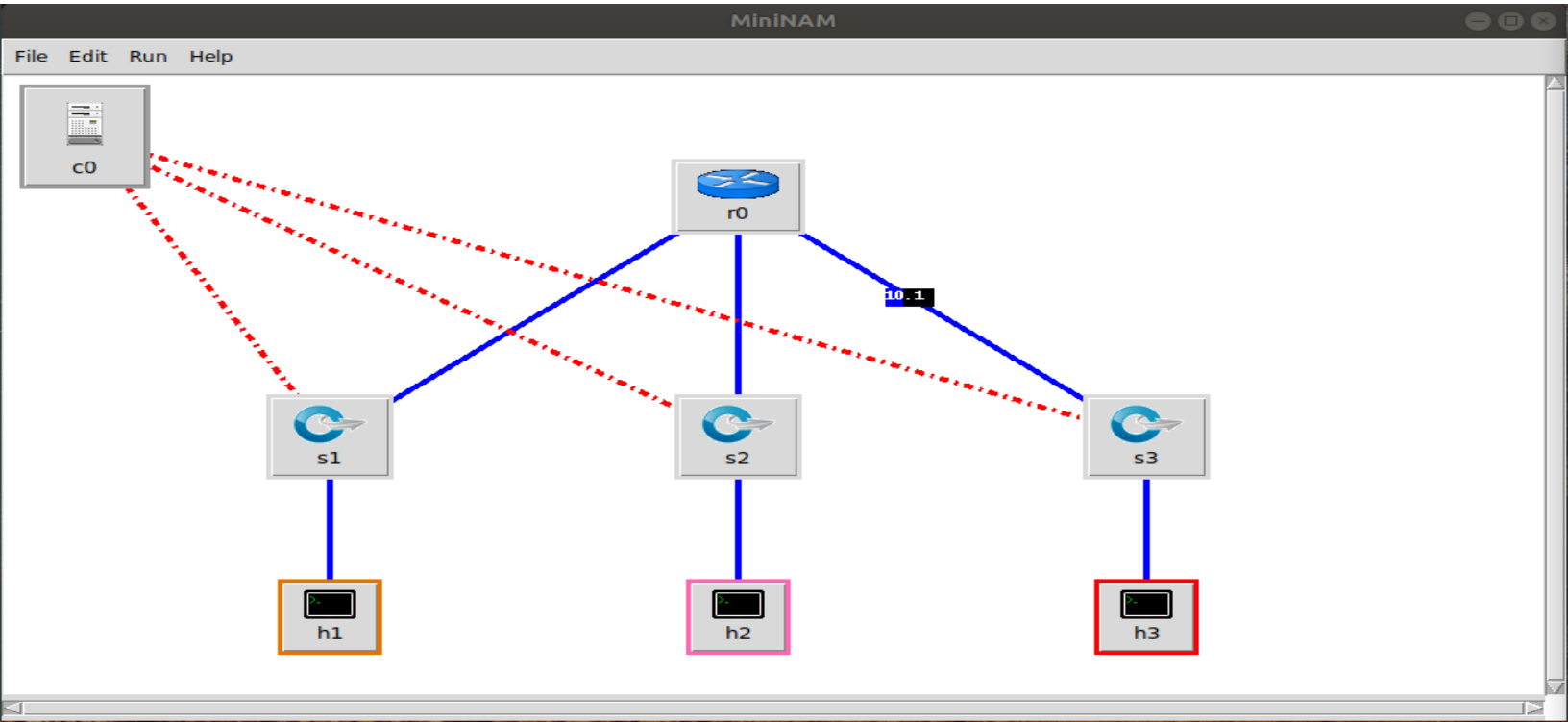
cd MiniNAM/Examples/NAT

sudo python MiniNAM.py --config conf.config --custom goodNAT.py --topo mytopo

h1# ping 10.0.0.100 -c3

Provide two screenshots from MiniNAM showing the NAT functionality (change of IP address by the router).





Part 3: ROUTING
Run:

cd ryu/ryu/app

ryu-manager simple_switch_stp_13.py

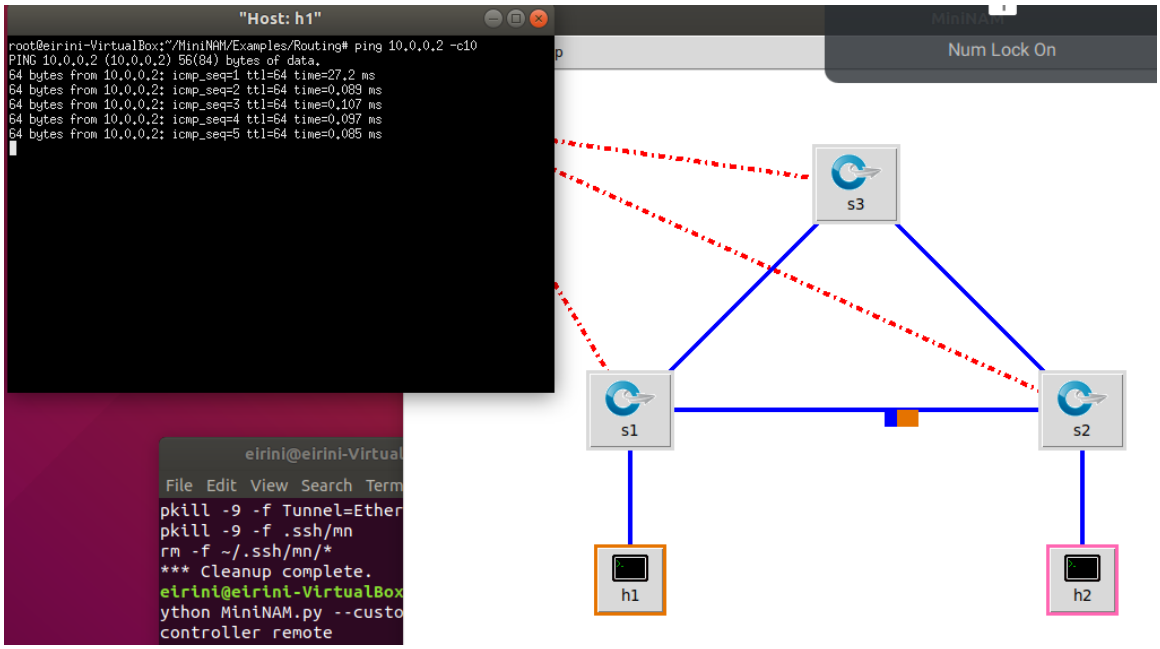
```
eirini@eirini-VirtualBox:~/ryu/ryu/app$ ryu-manager simple_switch_stp_13.py
loading app simple_switch_stp_13.py
loading app ryu.controller.ofp_handler
instantiating app None of Stp
creating context stplib
instantiating app simple_switch_stp_13.py of SimpleSwitch13
instantiating app ryu.controller.ofp_handler of OFPHandler
```

cd MiniNAM/Examples/Routing

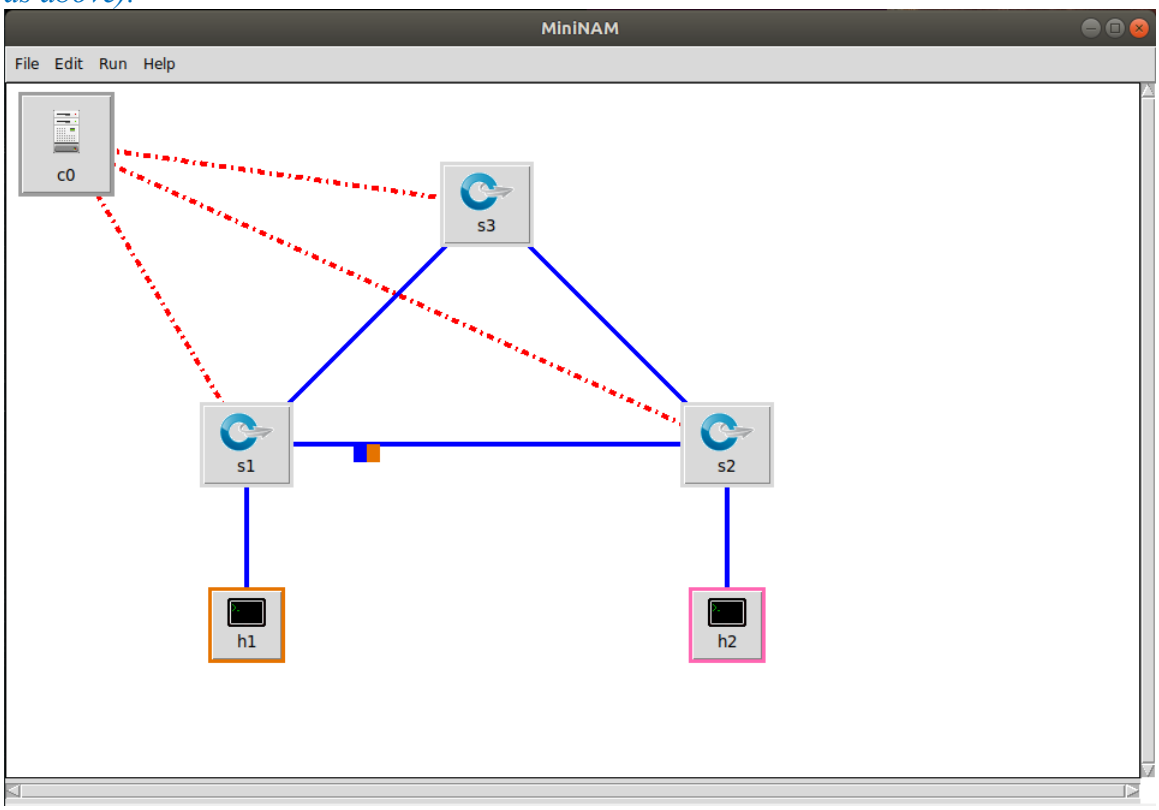
```
eirini@eirini-VirtualBox:~/MiniNAM/Examples/Routing$ sudo python MiniNAM.py --c
ustom spanning_tree.py --topo mytopo --controller remote
[sudo] password for eirini:
*** Creating network
*** Adding controller
Connecting to remote controller at 127.0.0.1:6653
*** Adding hosts:
h1 h2
*** Adding switches:
s1 s2 s3
*** Adding links:
(s1, h1) (s1, s2) (s2, h2) (s2, s3) (s3, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 3 switches
s1 s2 s3 ...
```

WAIT for Controller.

At h1 terminal:
ping 10.0.0.2 -c10

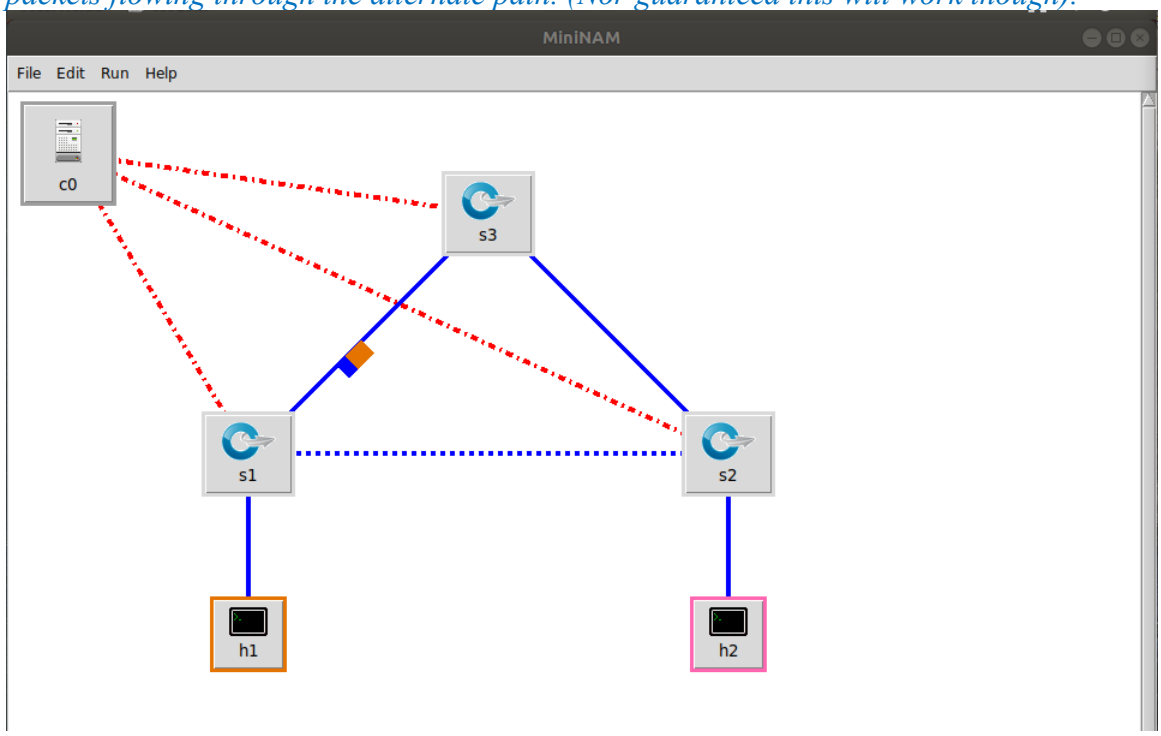


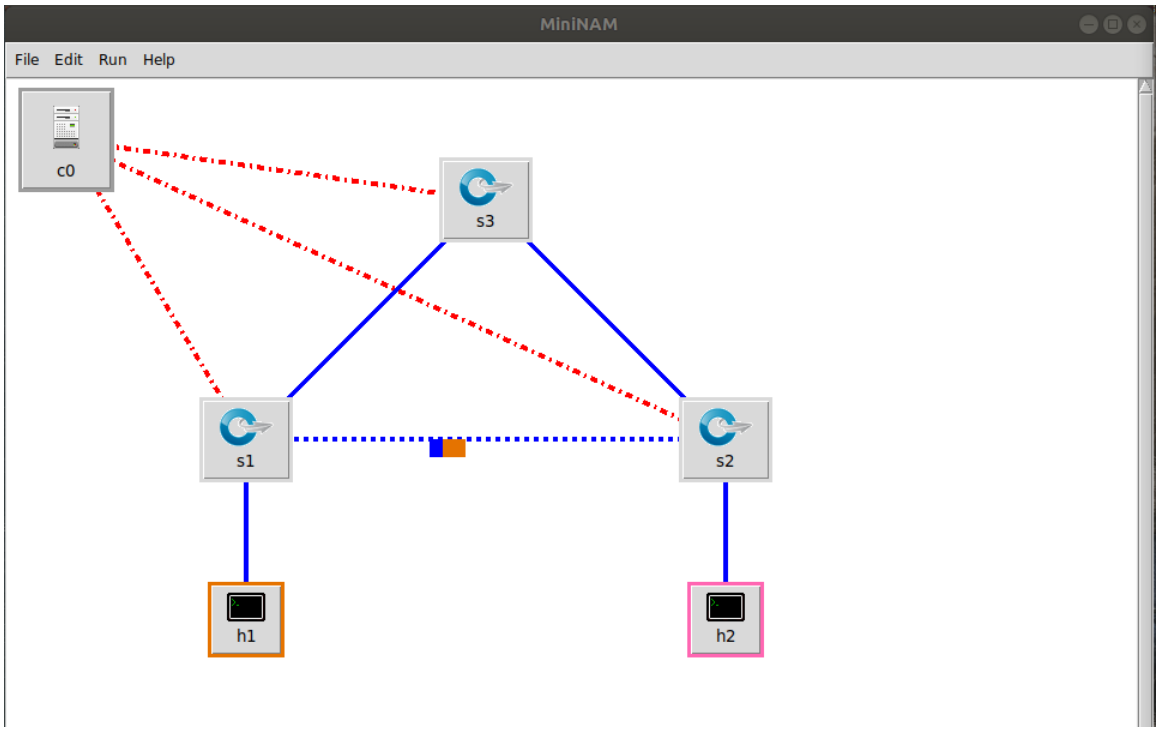
Provide a screenshot from MiniNAM showing the Routing functionality (packets flowing as above).



Optional: (test if this works OK)
Now right-click on the link between s1 and s2 and click on Link Down to simulate a broken link (or type "link s1 s2 down"). The controller will re-configure switches to

update the path. WAIT for a while. Send pings to h2 from h1 and you can observe ICMP packets flowing through the alternate path. (Nor guaranteed this will work though).





In my testing only 2 packets were transmitted through the alternate path and the other 8 packets were transmitted normally, as shown in the pictures above. Thus this doesn't work correctly