PicoHub: the PicoNet in a Hosted JupyterLab Cloud

Jianping Pan | University of Victoria, BC, Canada | Pan@UVic.CA

* readandspell.com

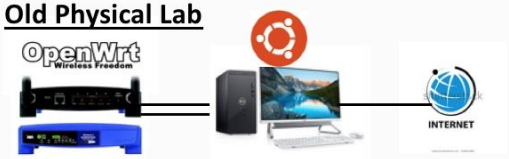
* dreamstime.com

Why? How?

1. Lightweight on students laptop
2. Lightweight on lab provisioning

3. Fun labs!

Old Physical Lab

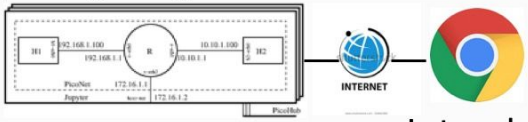


Intermediate Virtual Lab

Works well in a Physical Lab, *not* on students laptop

Physical Lab *not* accessible
VM too heavy on laptop
MiniNet too powerful

New Lightweight Virtual Hosted Lab



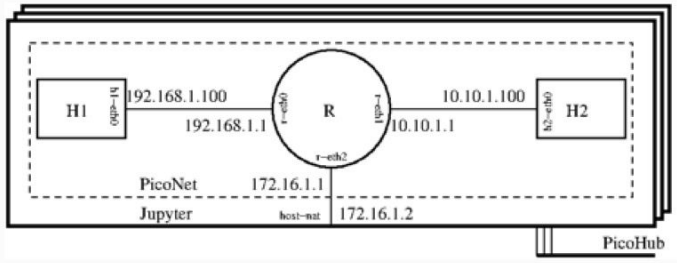
Hosted in the Cloud

Just a web browser on students laptop

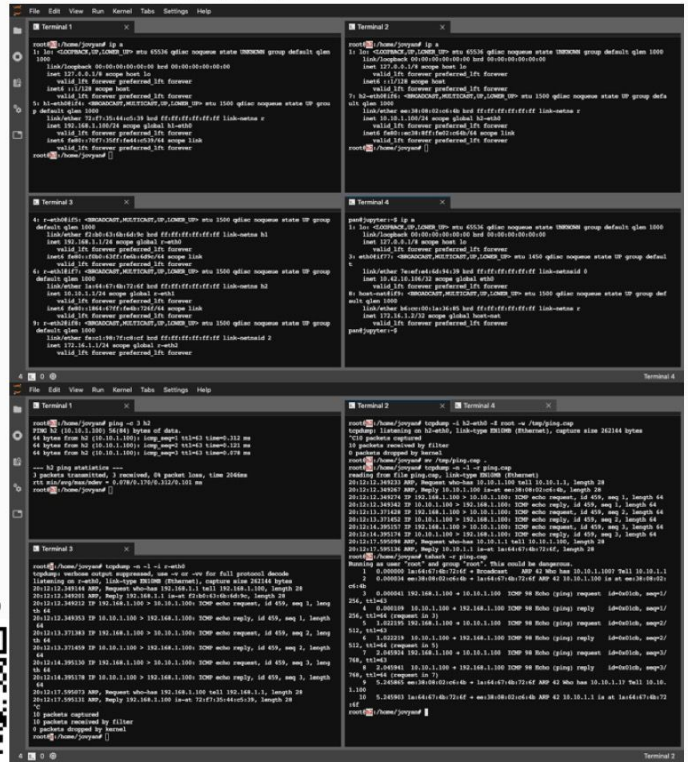
University of Victoria | Learning and Teaching Support and Innovation

PANDALAB

PicoNet: the Minimized MiniNet



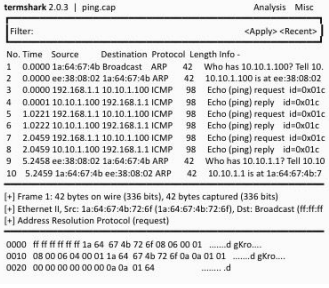
https://picohub.csc.uvic.ca



```
root@h1:/home/jovyan# ping -c 3 h2
PING h2 (10.10.1.100) 56(84) bytes of data.
64 bytes from h2 (10.10.1.100): icmp_seq=1 ttl=63 time=0.312 ms
64 bytes from h2 (10.10.1.100): icmp_seq=2 ttl=63 time=0.121 ms
64 bytes from h2 (10.10.1.100): icmp_seq=3 ttl=63 time=0.078 ms
--- h2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2046ms
rtt min/avg/max/mdev = 0.078/0.170/0.312/0.101 ms

root@r:/home/jovyan# tcpdump -n -l -i r-eth0
tcpdump: verbose output suppressed, use -v or -vv for full protocol
listening on r-eth0, link-type EN10MB (Ethernet), capture size 262144
20:12:12.349144 ARP, Request who-has 192.168.1.1 tell 192.168.1.100,
20:12:12.349201 ARP, Reply 192.168.1.1 is-at f2:b0:63:6b:6d:9c,
20:12:12.349212 IP 192.168.1.100 > 10.10.1.100: ICMP echo request, id
20:12:12.349353 IP 10.10.1.100 > 192.168.1.100: ICMP echo reply, id
20:12:13.371383 IP 192.168.1.100 > 10.10.1.100: ICMP echo request, id
20:12:13.371459 IP 10.10.1.100 > 192.168.1.100: ICMP echo reply, id
20:12:14.395130 IP 192.168.1.100 > 10.10.1.100: ICMP echo request, id
20:12:14.395178 IP 10.10.1.100 > 192.168.1.100: ICMP echo reply, id
20:12:17.595073 ARP, Request who-has 192.168.1.100 tell 192.168.1.1,
20:12:17.595131 ARP, Reply 192.168.1.100 is-at 72:f7:35:44:c5:39,
^C
10 packets captured
10 packets received by filter
0 packets dropped by kernel

root@h2:/home/jovyan# tcpdump -i h2-eth0 -Z root -w /tmp/ping.cap
tcpdump: listening on h2-eth0, link-type EN10MB (Ethernet), capture
size 262144 bytes
^C10 packets captured
10 packets received by filter
0 packets dropped by kernel
root@h2:/home/jovyan# mv /tmp/ping.cap .
root@h2:/home/jovyan# tshark -r ping.cap
Running as user "root" and group "root". This could be dangerous.
 1   0.000000 1a:64:67:4b:72:6f → Broadcast    ARP 42 Who has
 2   0.000034 ee:38:08:02:c6:4b → 1a:64:67:4b:72:6f ARP 42 10.10.1.100
 3   0.000041 192.168.1.100 → 10.10.1.100  ICMP 98 Echo (ping)
 4   0.000109  10.10.1.100 → 192.168.1.100 ICMP 98 Echo (ping)
 5   1.022195 192.168.1.100 → 10.10.1.100  ICMP 98 Echo (ping)
 6   1.022219  10.10.1.100 → 192.168.1.100 ICMP 98 Echo (ping)
 7   2.045924 192.168.1.100 → 10.10.1.100  ICMP 98 Echo (ping)
 8   2.045941  10.10.1.100 → 192.168.1.100 ICMP 98 Echo (ping)
 9   5.245865 ee:38:08:02:c6:4b → 1a:64:67:4b:72:6f ARP 42 Who has
10   5.245903 1a:64:67:4b:72:6f → ee:38:08:02:c6:4b ARP 42 10.10.1.1
```


termshark 2.0.3 | ping.cap
Analysis Misc
Filter:
<Apply> <Recent>
No. Time Source Destination Protocol Length Info
1 0.0000 1a:64:67:4b Broadcast ARP 42 Who has 10.10.1.100? Tell 10.
2 0.0000 ee:38:08:02 1a:64:67:4b ARP 42 10.10.1.100 is at ee:38:08:02
3 0.0000 192.168.1.1 10.10.1.100 ICMP 98 Echo (ping) request id=0x01c
4 0.0001 10.10.1.100 192.168.1.1 ICMP 98 Echo (ping) reply id=0x01c
5 1.0221 192.168.1.1 10.10.1.100 ICMP 98 Echo (ping) request id=0x01c
6 1.0222 10.10.1.100 192.168.1.1 ICMP 98 Echo (ping) reply id=0x01c
7 2.0459 192.168.1.1 10.10.1.100 ICMP 98 Echo (ping) request id=0x01c
8 2.0459 10.10.1.100 192.168.1.1 ICMP 98 Echo (ping) reply id=0x01c
9 5.2458 ee:38:08:02 1a:64:67:4b ARP 42 Who has 10.10.1.1? Tell 10.10
10 5.2459 1a:64:67:4b ee:38:08:02 ARP 42 10.10.1.1 is at 1a:64:67:4b:7
[+] Frame 1: 42 bytes on wire (336 bits), 42 bytes captured (336 bits)
[+] Ethernet II, Src: 1a:64:67:4b:72:6f (1a:64:67:4b:72:6f), Dst: Broadcast (ff:ff:ff
[+] Address Resolution Protocol (request)
0000 ff ff ff ff ff ff 1a 64 67 4b 72 6f 08 06 00 01 ....d gKro....
0010 08 00 06 04 00 01 1a 64 67 4b 72 6f 0a 0a 01 01 ....d gKro....
0020 00 00 00 00 00 00 0a 0a 01 64 .....d
```

More Lab modules on HTTP, DNS, TCP, UDP IP, NAT, ICMP, routing Ethernet, ARP and more at

http://tinyurl.com/picohub

Thanks: Mantis, Miguel, Tomas Victoria, Zhiming, Rui, Wenjun

ACM SIGITE Information Technology Education
```

PicoHub: the PicoNet in a Hosted JupyterLab Cloud

UVic CSc 361
Lab Redesign and Development Team
Summer and Fall 2020

Old CSc361 Lab platform (2007--2017)

- Minimized OpenWRT kernel (firmware image)
  - 10 lab modules on **HTTP, DNS, TCP, UDP, IP, ICMP, routing, Ethernet, forwarding** and **ARP**
  - **netem** (network emulator) to introduce **packet delay, loss, duplication** and **reordering**
- Running on Linksys WRT54GL (customized firmware)
  - Students have **root** (privileged) access on router: **tcpdump** and more
  - Can change all configurations in RAM (memory): "*really **own** a network*"
  - Default configuration in the read-only ROM (reflashable when needed)
  - A power cycle between lab sections resets everything to default: **0-maintenance**
- Dual cabled to the host computer running stock Ubuntu Linux
  - As both the *client* computer (at home) and the *server* computer (on the Internet)
  - **Packet trace analysis, C programming development,** etc
  - With the "send-to-self" kernel patch: *repatching needed if kernel upgrades*
- Installed in ECS360 (computer networks teaching lab)
  - Students like it, **some flashed the image at home**, and *a few even took away two routers* ;-)
- Supported by UVic LTC (learning/teaching center) course development grant
  - Published at ACM SIGCSE 2010: **Teaching computer networks in a real network**
  - https://dl.acm.org/doi/10.1145/1734263.1734311

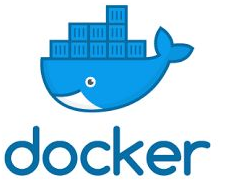
Intermediate CSc361 Lab platform (2018--2020)

- New challenges on the old lab platform
  - Limited seats in ECS360
    - Increased from 18 to 30 after the room reconfiguration
    - Still cannot catch up with the enrolment increase
  - Not enough Linksys routers
    - Even with the donated ones from the research lab
    - No new budget to purchase the small routers
  - Kernel patching is cumbersome
    - Although only configuration needed with newer kernels*
  - **C programming---a big problem for many students**
- An intermediate solution---***also moved from C to Python***
  - MiniNet (mainly a research platform) in a full/old Ubuntu VM in Virtualbox
  - VM (virtual machine) works reasonably well on desktop computers in ECS360
- New challenges due to CoViD-19 in Spring 2020
  - Students cannot access ECS360 in person
  - Remote access is disabled due to security concerns---difficult with GUI too
  - Student's laptop is not sufficient to run the MiniNet in a full GUI VM, Virtualbox issues, etc

* http://blog.asiantuntijakaveri.fi/2012/08/send-to-self-on-linux.html

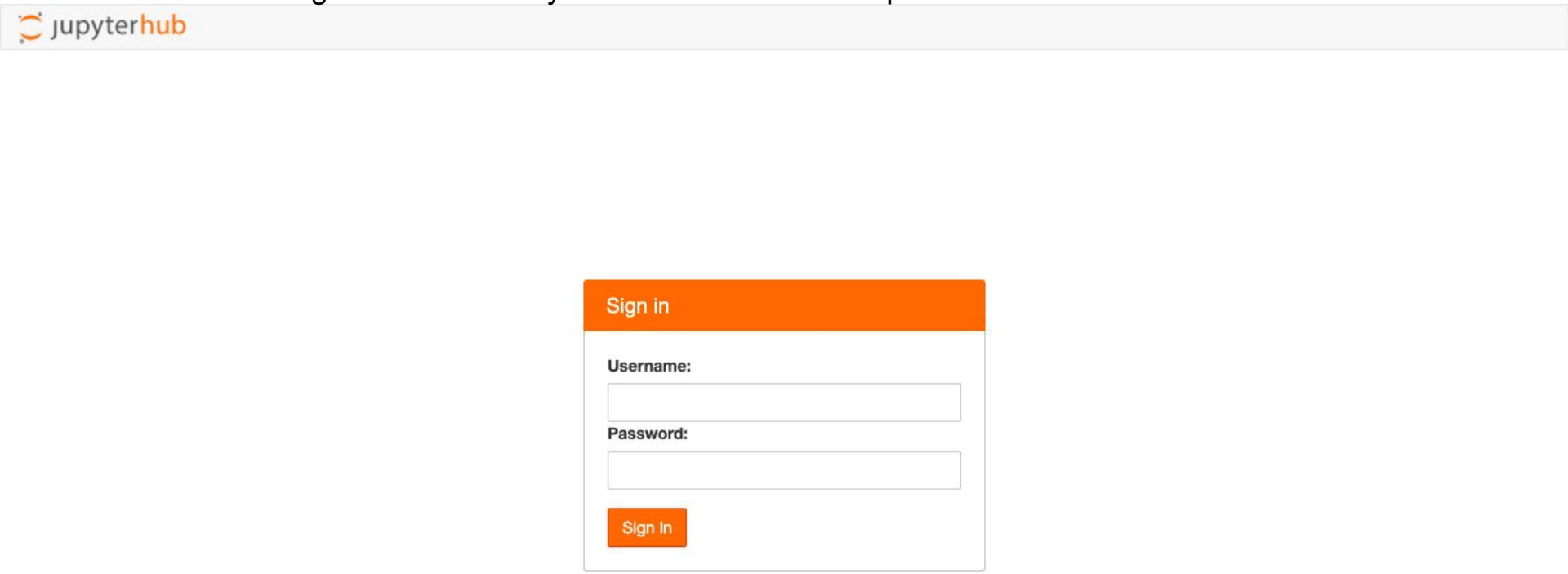
PicoHub: New CSc361 Lab platform (2020--)

- New requirements
  - Lightweight on student's laptop
    - Only a web browser (e.g., Chrome) needed
  - Lightweight on lab provisioning
    - Virtual lab hosted in the cloud by the dept
    - Tested with minimized Ubuntu VM first
    - Settled down with the lightest Docker container
    - A powerful server can host many Docker instances
  - PicoNet: the minimized MiniNet
    - Use the same Linux network namespaces technology
    - Fixed "client-router-server" topology, less chance to go wrong
    - Targeted ***only*** for the introductory computer networks course (CSc361)
  - JupyterLab UI (user interface) familiar to many students already
- Available at **https://picohub.csc.uvic.ca**
  - Internal test site https://picotest.csc.uvic.ca for the teaching team only
  - Accessibility features incorporated and enhanced
    - Screen-reader compatibility for blind/low-vision students

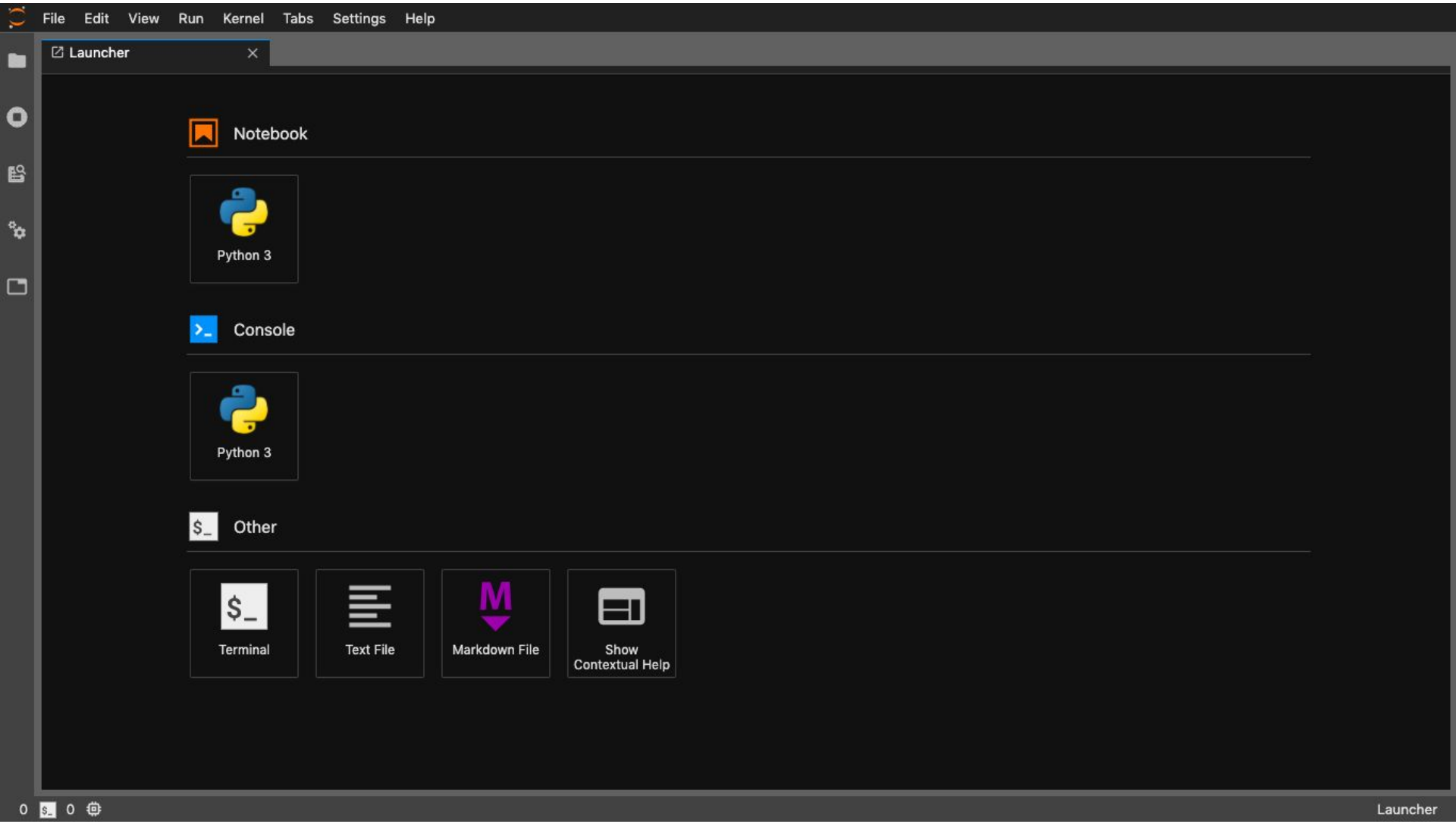


https://picohub.csc.uvic.ca

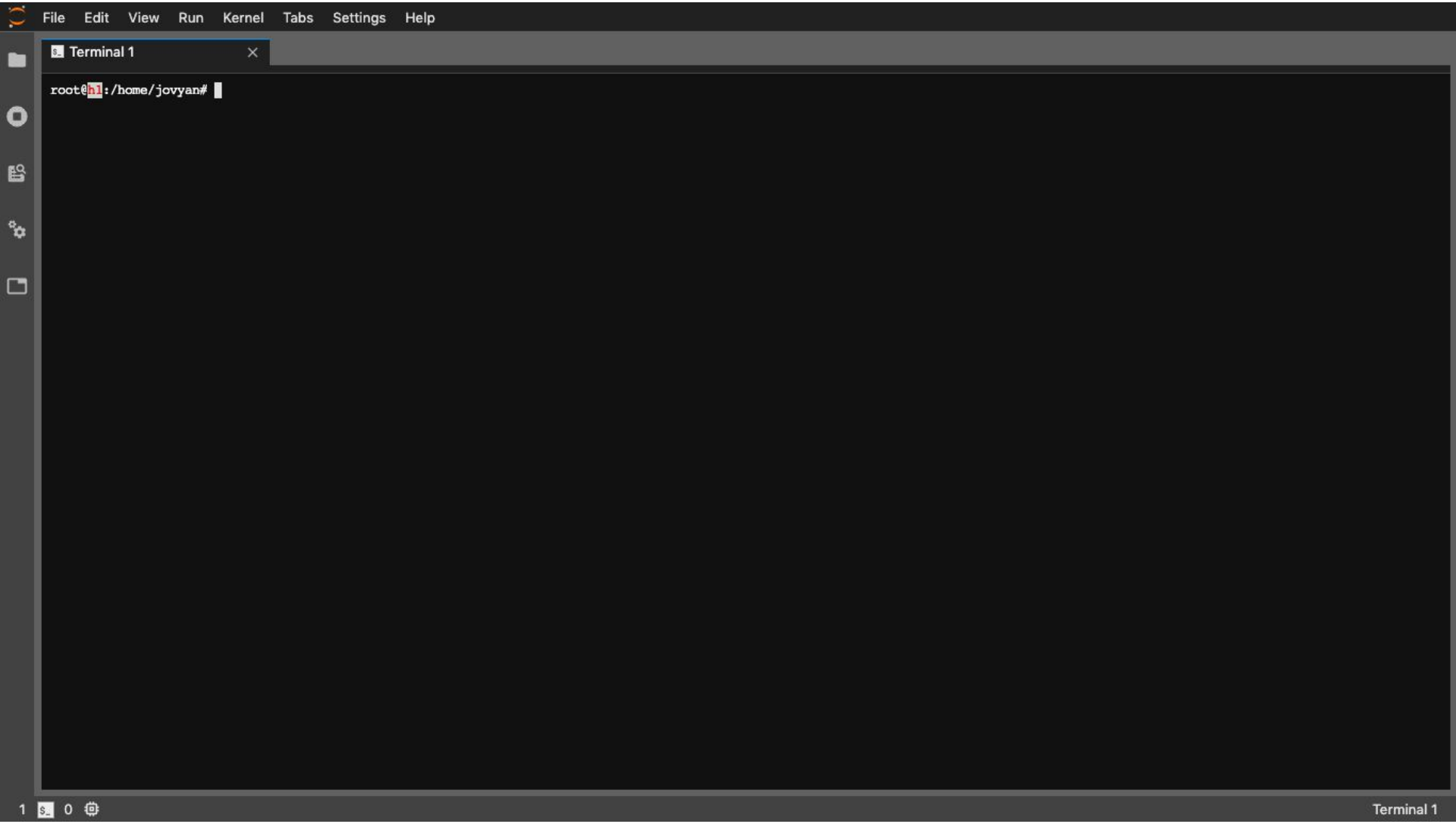
- **Please connect to UVic through UVic VPN first**
  - And login PicoHub with your UVic Netlink ID and password



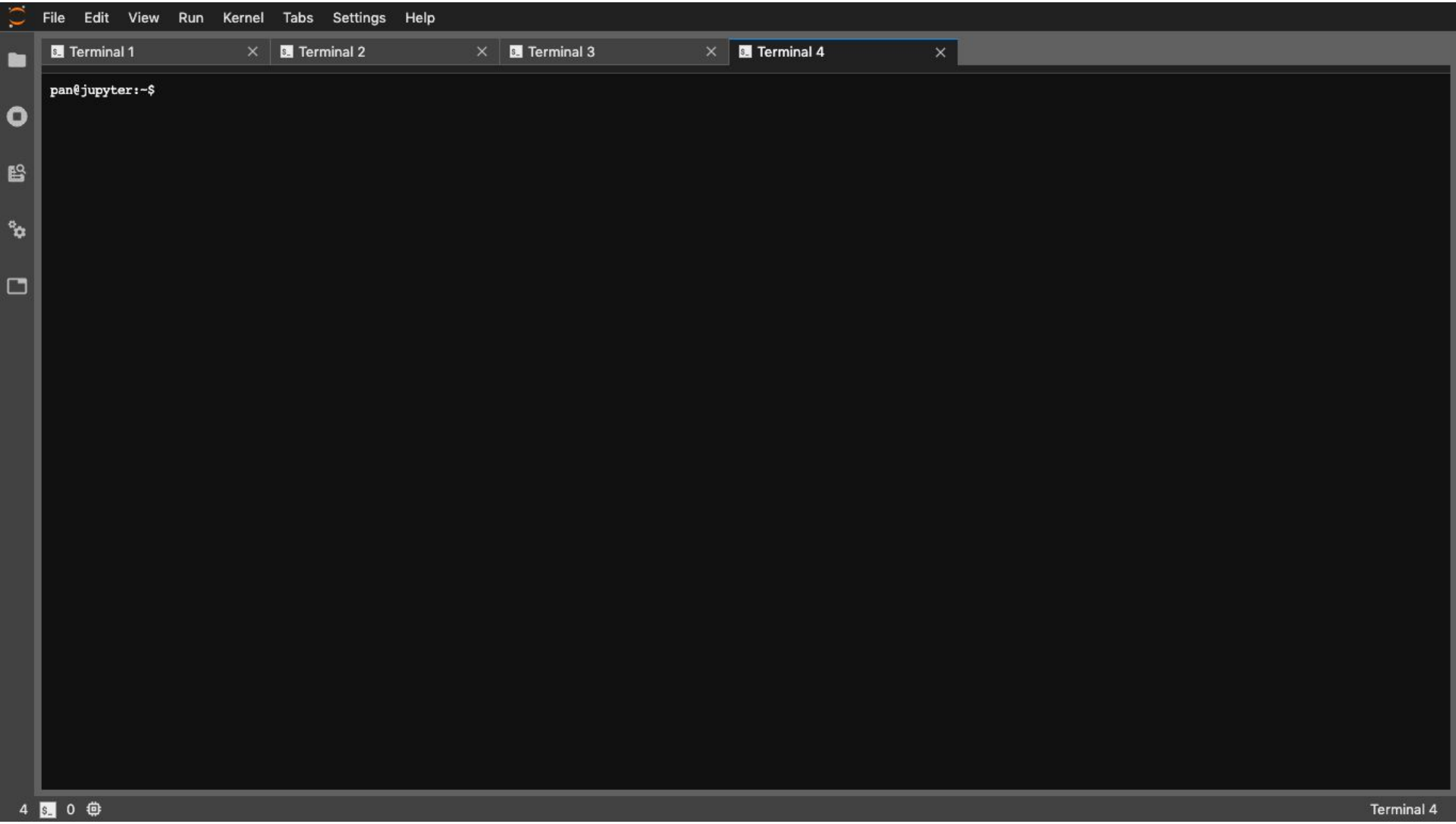
You will see a landing page like this



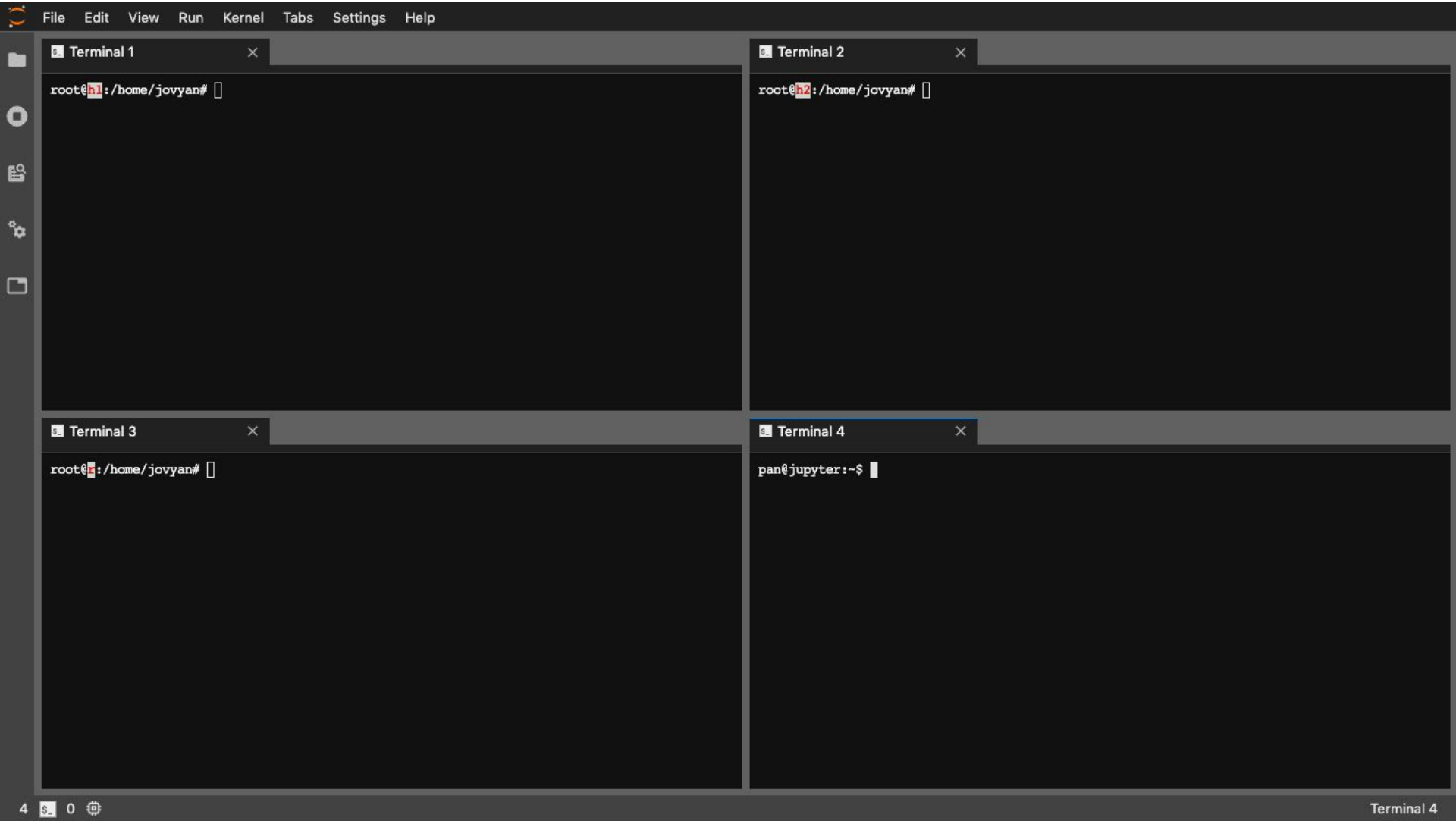
“Launcher -> Terminal” to get the client host (H1)



“File -> New -> Terminal” 3 times (for H2, R, Jupyter)

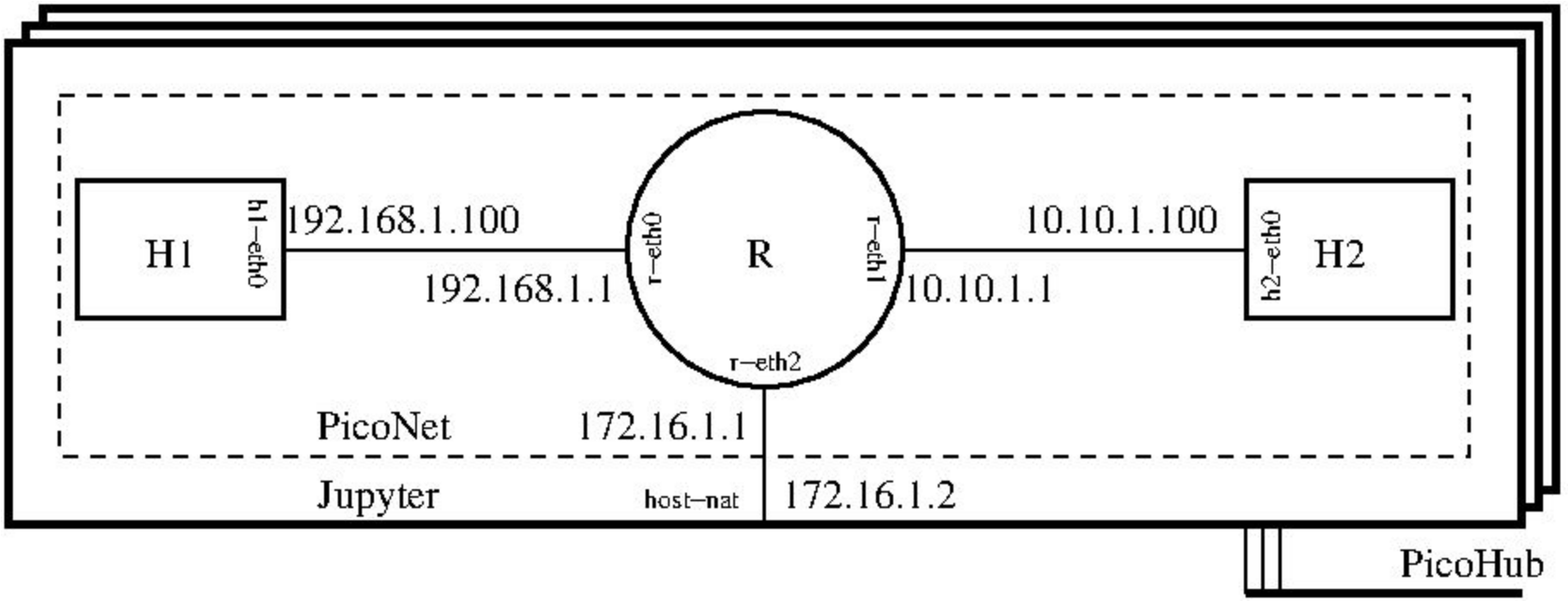


Grab Terminal tab and move to edge/concern to tile

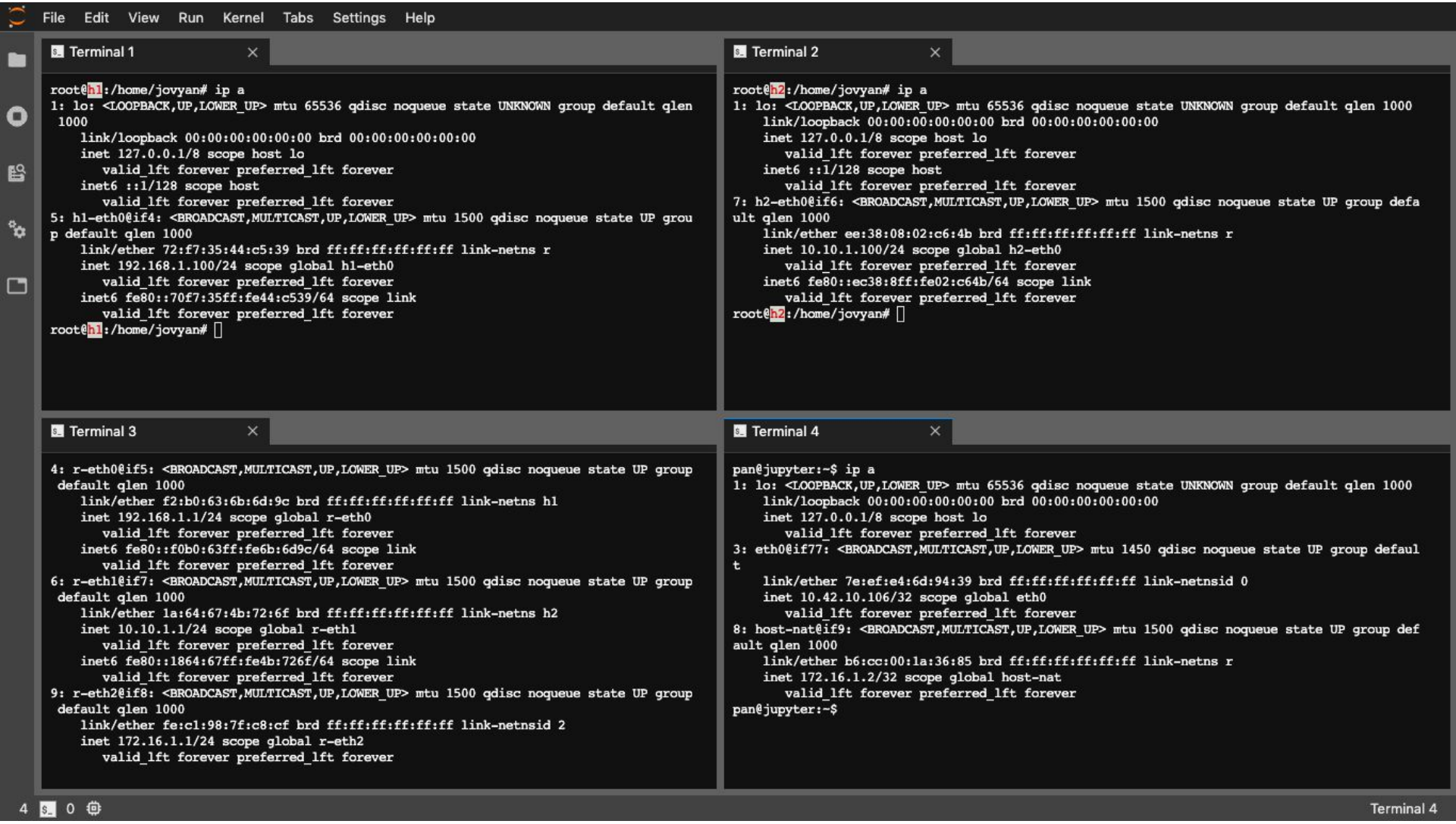


PicoNet vs Jupyter vs PicoHub in a nutshell

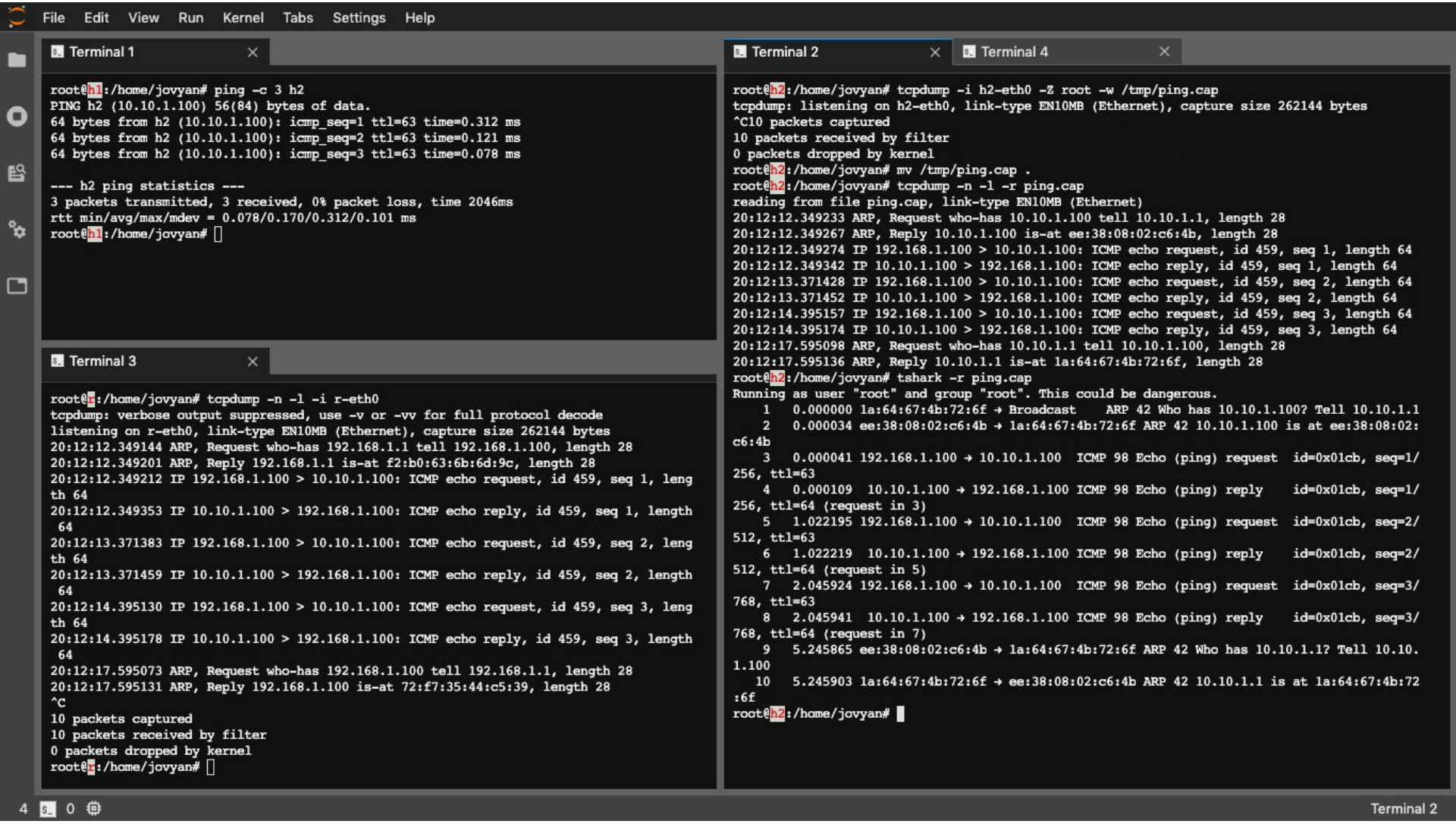
- H1: the client computer (connected to R through h1-eth0)
- R: the router (connected to H1 and H2 thru r-eth0 and r-eth1, respectively)
  - Also connected through r-eth2 to the PicoNet host (host-nat) and then to the Internet
- H2: the server computer (connected to R through h2-eth0)



PicoNet: H1 (client) -- R (router) -- H2 (server)



Let’s play with PicoNet a bit: H1 ping H2



On H1 as the client

```
root@h1:/home/jovyan# ping -c 3 h2
PING h2 (10.10.1.100) 56(84) bytes of data.
64 bytes from h2 (10.10.1.100): icmp_seq=1 ttl=63 time=0.312 ms
64 bytes from h2 (10.10.1.100): icmp_seq=2 ttl=63 time=0.121 ms
64 bytes from h2 (10.10.1.100): icmp_seq=3 ttl=63 time=0.078 ms

--- h2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2046ms
rtt min/avg/max/mdev = 0.078/0.170/0.312/0.101 ms
root@h1:/home/jovyan#
```

On R as the router connecting the client and server

```
root@r:/home/jovyan# tcpdump -n -l -i r-eth0
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on r-eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
20:12:12.349144 ARP, Request who-has 192.168.1.1 tell 192.168.1.100, length 28
20:12:12.349201 ARP, Reply 192.168.1.1 is-at f2:b0:63:6b:6d:9c, length 28
20:12:12.349212 IP 192.168.1.100 > 10.10.1.100: ICMP echo request, id 459, seq
1, length 64
20:12:12.349353 IP 10.10.1.100 > 192.168.1.100: ICMP echo reply, id 459, seq
1, length 64
20:12:13.371383 IP 192.168.1.100 > 10.10.1.100: ICMP echo request, id 459, seq
2, length 64
20:12:13.371459 IP 10.10.1.100 > 192.168.1.100: ICMP echo reply, id 459, seq
2, length 64
20:12:14.395130 IP 192.168.1.100 > 10.10.1.100: ICMP echo request, id 459, seq
3, length 64
20:12:14.395178 IP 10.10.1.100 > 192.168.1.100: ICMP echo reply, id 459, seq
3, length 64
20:12:17.595073 ARP, Request who-has 192.168.1.100 tell 192.168.1.1, length 28
20:12:17.595131 ARP, Reply 192.168.1.100 is-at 72:f7:35:44:c5:39, length 28
^C
10 packets captured
10 packets received by filter
0 packets dropped by kernel
```

On H2 as the server to write to file for analysis

```
root@h2:/home/jovyan# tcpdump -i h2-eth0 -Z root -w /tmp/ping.cap
tcpdump: listening on h2-eth0, link-type EN10MB (Ethernet), capture size
262144 bytes
^C10 packets captured
10 packets received by filter
0 packets dropped by kernel
root@h2:/home/jovyan# mv /tmp/ping.cap .
root@h2:/home/jovyan# tcpdump -n -l -r ping.cap
reading from file ping.cap, link-type EN10MB (Ethernet)
20:12:12.349233 ARP, Request who-has 10.10.1.100 tell 10.10.1.1, length 28
20:12:12.349267 ARP, Reply 10.10.1.100 is-at ee:38:08:02:c6:4b, length 28
20:12:12.349274 IP 192.168.1.100 > 10.10.1.100: ICMP echo request, id 459, seq
1, length 64
20:12:12.349342 IP 10.10.1.100 > 192.168.1.100: ICMP echo reply, id 459, seq
1, length 64
20:12:13.371428 IP 192.168.1.100 > 10.10.1.100: ICMP echo request, id 459, seq
2, length 64
20:12:13.371452 IP 10.10.1.100 > 192.168.1.100: ICMP echo reply, id 459, seq
2, length 64
20:12:14.395157 IP 192.168.1.100 > 10.10.1.100: ICMP echo request, id 459, seq
3, length 64
20:12:14.395174 IP 10.10.1.100 > 192.168.1.100: ICMP echo reply, id 459, seq
3, length 64
20:12:17.595098 ARP, Request who-has 10.10.1.1 tell 10.10.1.100, length 28
20:12:17.595136 ARP, Reply 10.10.1.1 is-at 1a:64:67:4b:72:6f, length 28
```

Or use tshark

```
root@h2:/home/jovyan# tshark -r ping.cap
Running as user "root" and group "root". This could be dangerous.
    1   0.000000 1a:64:67:4b:72:6f → Broadcast    ARP 42 Who has 10.10.1.100?
Tell 10.10.1.1
    2   0.000034 ee:38:08:02:c6:4b → 1a:64:67:4b:72:6f ARP 42 10.10.1.100 is
at ee:38:08:02:c6:4b
    3   0.000041 192.168.1.100 → 10.10.1.100  ICMP 98 Echo (ping) request
id=0x01cb, seq=1/256, ttl=63
    4   0.000109  10.10.1.100 → 192.168.1.100 ICMP 98 Echo (ping) reply
id=0x01cb, seq=1/256, ttl=64 (request in 3)
    5   1.022195 192.168.1.100 → 10.10.1.100  ICMP 98 Echo (ping) request
id=0x01cb, seq=2/512, ttl=63
    6   1.022219  10.10.1.100 → 192.168.1.100 ICMP 98 Echo (ping) reply
id=0x01cb, seq=2/512, ttl=64 (request in 5)
    7   2.045924 192.168.1.100 → 10.10.1.100  ICMP 98 Echo (ping) request
id=0x01cb, seq=3/768, ttl=63
    8   2.045941  10.10.1.100 → 192.168.1.100 ICMP 98 Echo (ping) reply
id=0x01cb, seq=3/768, ttl=64 (request in 7)
    9   5.245865 ee:38:08:02:c6:4b → 1a:64:67:4b:72:6f ARP 42 Who has
10.10.1.1? Tell 10.10.1.100
   10   5.245903 1a:64:67:4b:72:6f → ee:38:08:02:c6:4b ARP 42 10.10.1.1 is at
1a:64:67:4b:72:6f
```

Or use “termshark -r ping.cap”

File Edit View Run Kernel Tabs Settings Help

Terminal 1

```
root@h1:/home/jovyan# ping -c 3 h2
PING h2 (10.10.1.100) 56(84) bytes of data.
64 bytes from h2 (10.10.1.100): icmp_seq=1 ttl=63 time=0.312 ms
64 bytes from h2 (10.10.1.100): icmp_seq=2 ttl=63 time=0.121 ms
64 bytes from h2 (10.10.1.100): icmp_seq=3 ttl=63 time=0.078 ms

--- h2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2046ms
rtt min/avg/max/mdev = 0.078/0.170/0.312/0.101 ms
root@h1:/home/jovyan#
```

Terminal 3

```
root@r:/home/jovyan# tcpdump -n -l -i r-eth0
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on r-eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
20:12:12.349144 ARP, Request who-has 192.168.1.1 tell 192.168.1.100, length 28
20:12:12.349201 ARP, Reply 192.168.1.1 is-at f2:b0:63:6b:6d:9c, length 28
20:12:12.349212 IP 192.168.1.100 > 10.10.1.100: ICMP echo request, id 459, seq 1, leng
th 64
20:12:12.349353 IP 10.10.1.100 > 192.168.1.100: ICMP echo reply, id 459, seq 1, length
 64
20:12:13.371383 IP 192.168.1.100 > 10.10.1.100: ICMP echo request, id 459, seq 2, leng
th 64
20:12:13.371459 IP 10.10.1.100 > 192.168.1.100: ICMP echo reply, id 459, seq 2, length
 64
20:12:14.395130 IP 192.168.1.100 > 10.10.1.100: ICMP echo request, id 459, seq 3, leng
th 64
20:12:14.395178 IP 10.10.1.100 > 192.168.1.100: ICMP echo reply, id 459, seq 3, length
 64
20:12:17.595073 ARP, Request who-has 192.168.1.100 tell 192.168.1.1, length 28
20:12:17.595131 ARP, Reply 192.168.1.100 is-at 72:f7:35:44:c5:39, length 28
^C
10 packets captured
10 packets received by filter
0 packets dropped by kernel
root@r:/home/jovyan#
```

Terminal 2 | **Terminal 4**

termshark 2.0.3 | ping.cap — Analysis Misc

Filter: <Apply> <Recent>

No.	Time	Source	Destination	Protocol	Length	Info
1	0.00000	1a:64:67:4b:	Broadcast	ARP	42	Who has 10.10.1.100? Tell 10.10
2	0.00003	ee:38:08:02:	1a:64:67:4b:	ARP	42	10.10.1.100 is at ee:38:08:02:c
3	0.00004	192.168.1.10	10.10.1.100	ICMP	98	Echo (ping) request id=0x01cb,
4	0.00010	10.10.1.100	192.168.1.10	ICMP	98	Echo (ping) reply id=0x01cb,
5	1.02219	192.168.1.10	10.10.1.100	ICMP	98	Echo (ping) request id=0x01cb,
6	1.02221	10.10.1.100	192.168.1.10	ICMP	98	Echo (ping) reply id=0x01cb,
7	2.04592	192.168.1.10	10.10.1.100	ICMP	98	Echo (ping) request id=0x01cb,
8	2.04594	10.10.1.100	192.168.1.10	ICMP	98	Echo (ping) reply id=0x01cb,
9	5.24586	ee:38:08:02:	1a:64:67:4b:	ARP	42	Who has 10.10.1.1? Tell 10.10.1
10	5.24590	1a:64:67:4b:	ee:38:08:02:	ARP	42	10.10.1.1 is at 1a:64:67:4b:72:

```
[+] Frame 1: 42 bytes on wire (336 bits), 42 bytes captured (336 bits)
[+] Ethernet II, Src: 1a:64:67:4b:72:6f (1a:64:67:4b:72:6f), Dst: Broadcast (ff:ff:ff:ff:ff:
[-] Address Resolution Protocol (request)
      Hardware type: Ethernet (1)
      Protocol type: IPv4 (0x0800)
      Hardware size: 6
      Protocol size: 4
      Opcode: request (1)
      Sender MAC address: 1a:64:67:4b:72:6f (1a:64:67:4b:72:6f)
      Sender IP address: 10.10.1.1
      Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
      Target IP address: 10.10.1.100
```

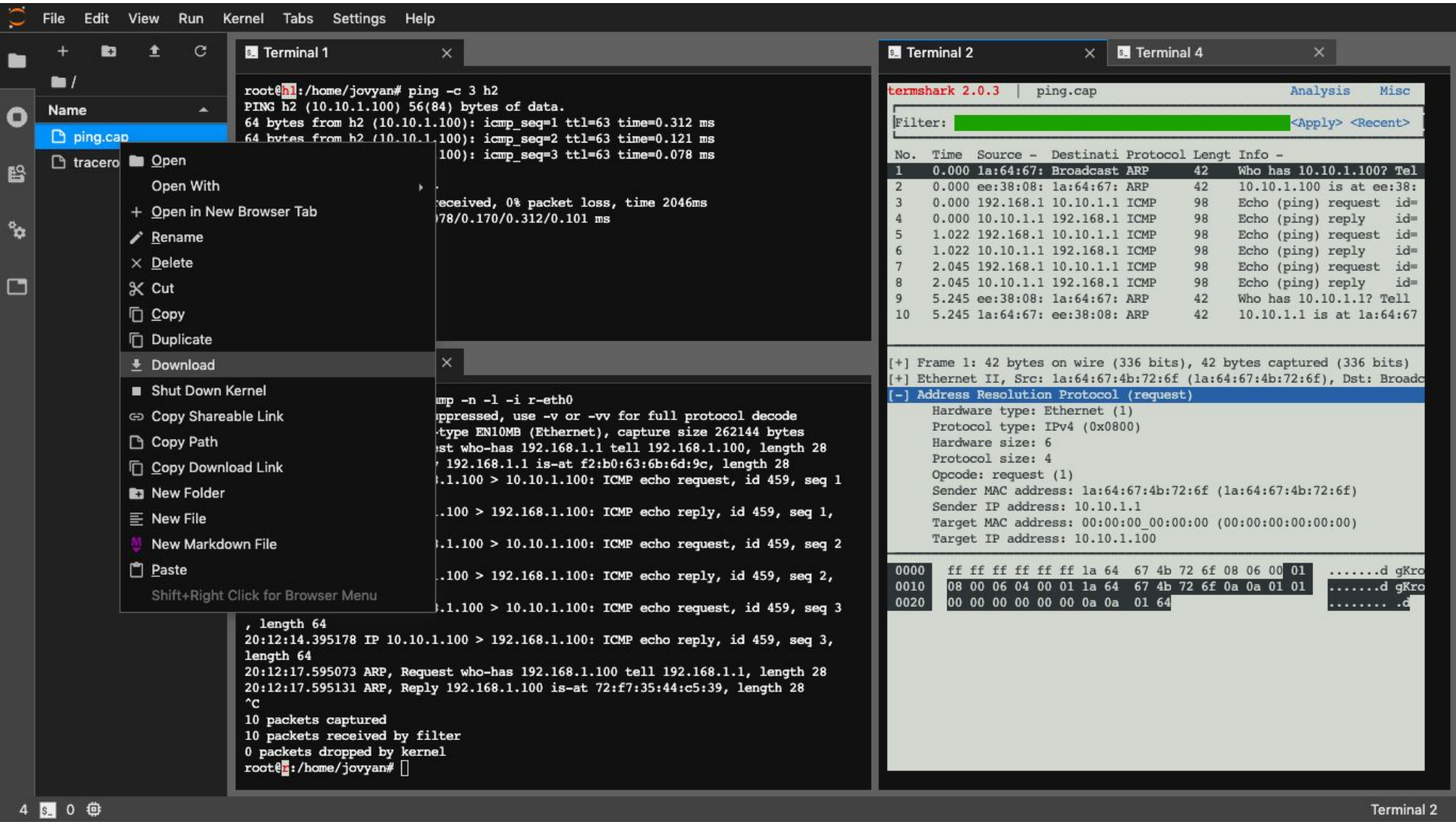
```
0000   ff ff ff ff ff ff 1a 64  67 4b 72 6f 08 06 00 01   .......d gKro....
0010   08 00 06 04 00 01 1a 64  67 4b 72 6f 0a 0a 01 01   .......d gKro....
0020   00 00 00 00 00 00 0a 0a  01 64                     ........ .d
```

4 $_ 0 Terminal 1

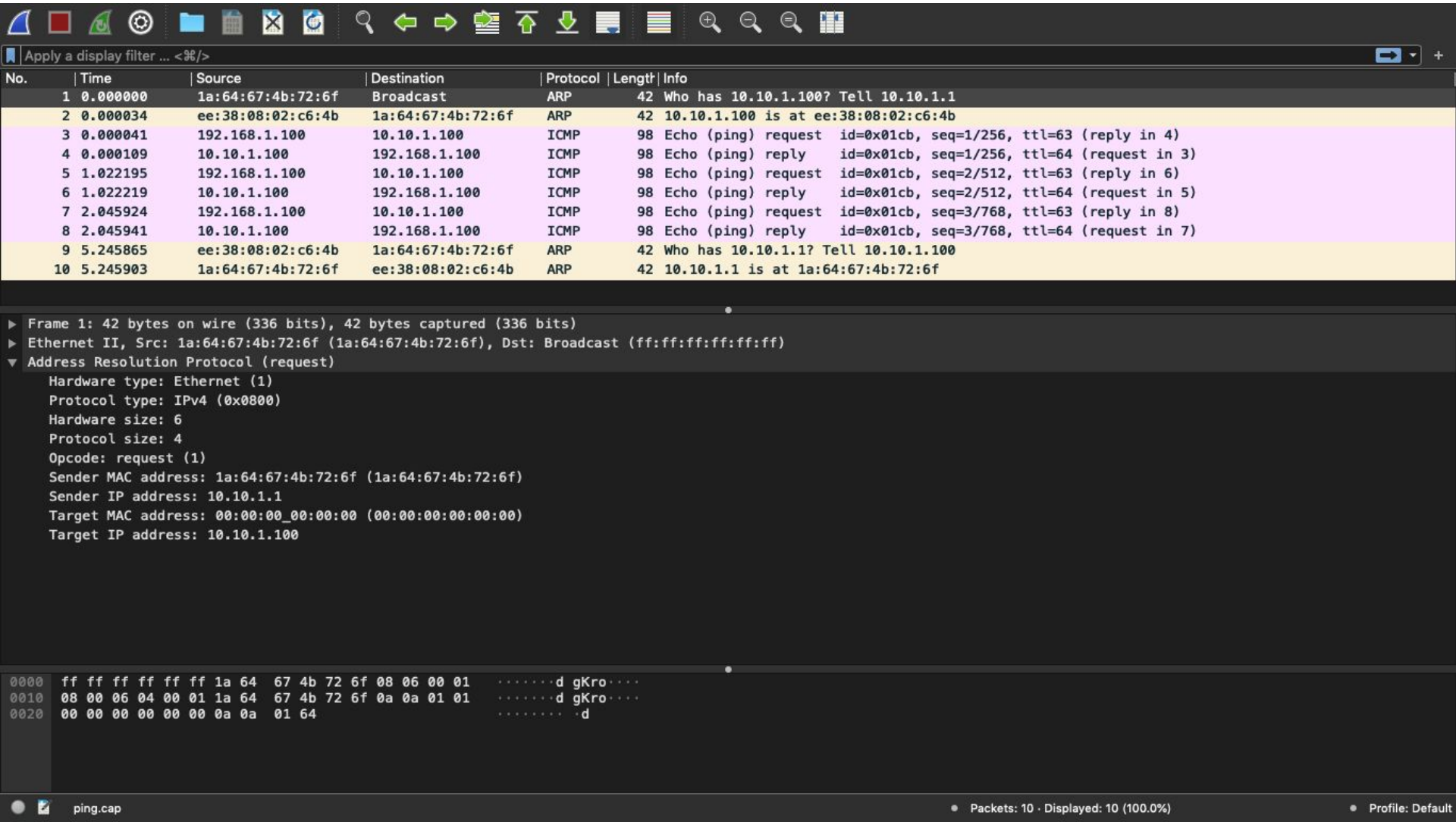
In text format

```
termshark 2.0.3  |  ping.cap                                                    Analysis    Misc
┌──────────────────────────────────────────────────────────────────────────────────────────────┐
│Filter:                                                                  <Apply> <Recent>     │
└──────────────────────────────────────────────────────────────────────────────────────────────┘
 No. - Time - Source -    Destination Protocol  Length Info -
 1     0.0000 1a:64:67:4b Broadcast   ARP       42     Who has 10.10.1.100? Tell 10.
 2     0.0000 ee:38:08:02 1a:64:67:4b ARP       42     10.10.1.100 is at ee:38:08:02
 3     0.0000 192.168.1.1 10.10.1.100 ICMP      98     Echo (ping) request  id=0x01c
 4     0.0001 10.10.1.100 192.168.1.1 ICMP      98     Echo (ping) reply    id=0x01c
 5     1.0221 192.168.1.1 10.10.1.100 ICMP      98     Echo (ping) request  id=0x01c
 6     1.0222 10.10.1.100 192.168.1.1 ICMP      98     Echo (ping) reply    id=0x01c
 7     2.0459 192.168.1.1 10.10.1.100 ICMP      98     Echo (ping) request  id=0x01c
 8     2.0459 10.10.1.100 192.168.1.1 ICMP      98     Echo (ping) reply    id=0x01c
 9     5.2458 ee:38:08:02 1a:64:67:4b ARP       42     Who has 10.10.1.1? Tell 10.10
 10    5.2459 1a:64:67:4b ee:38:08:02 ARP       42     10.10.1.1 is at 1a:64:67:4b:7
──────────────────────────────────────────────────────────────────────────────────────────────
[+] Frame 1: 42 bytes on wire (336 bits), 42 bytes captured (336 bits)
[+] Ethernet II, Src: 1a:64:67:4b:72:6f (1a:64:67:4b:72:6f), Dst: Broadcast (ff:ff:ff
[+] Address Resolution Protocol (request)
──────────────────────────────────────────────────────────────────────────────────────────────
 0000   ff ff ff ff ff ff 1a 64  67 4b 72 6f 08 06 00 01   .......d gKro....
 0010   08 00 06 04 00 01 1a 64  67 4b 72 6f 0a 0a 01 01   .......d gKro....
 0020   00 00 00 00 00 00 0a 0a  01 64                     ........ .d
```

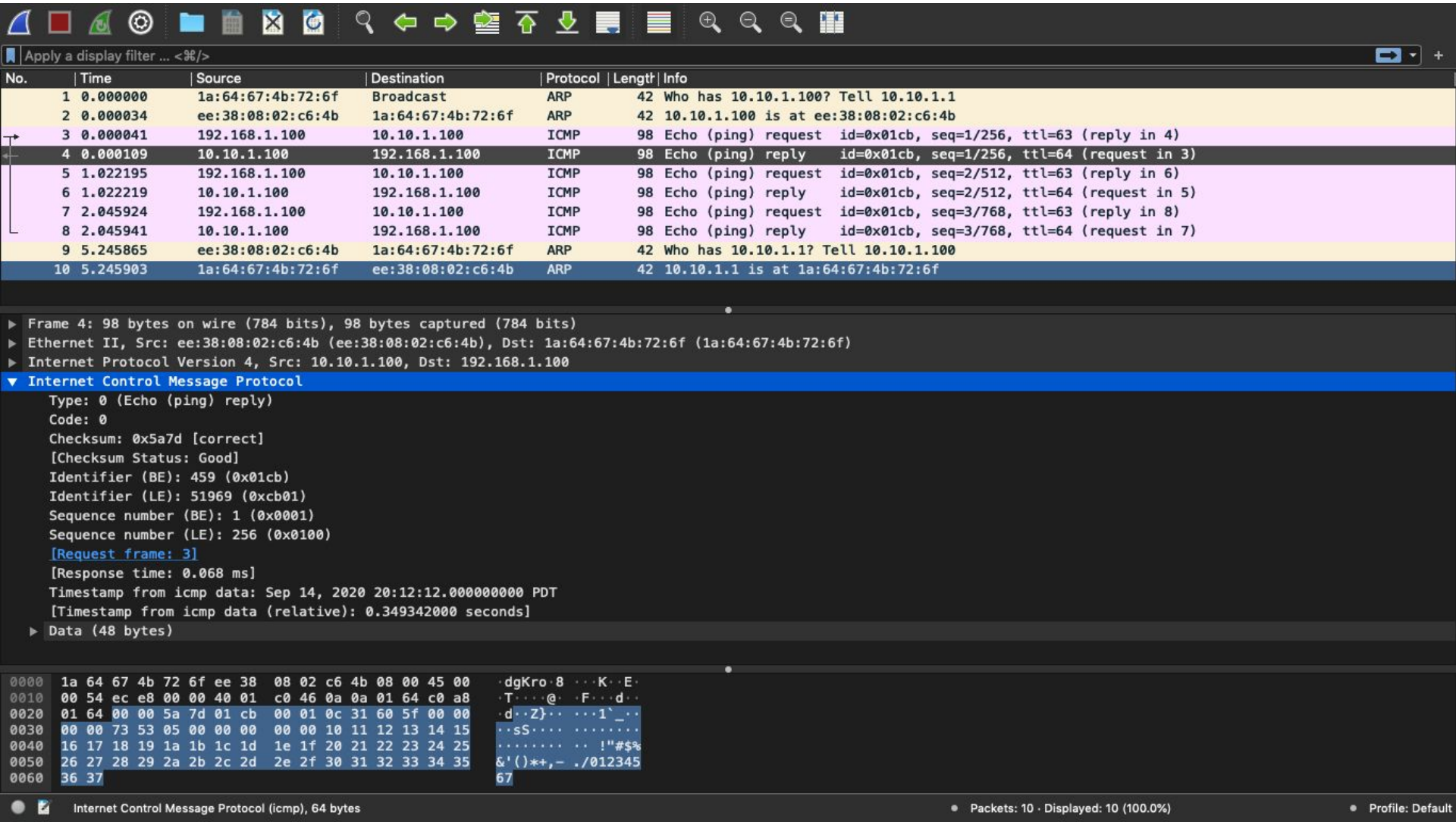
Or download to your local computer



Analyze in your Wireshark: ARP request



Ping request (and response in frame: 4)



Another practice if time allows: traceroute to Google

File Edit View Run Kernel Tabs Settings Help

Terminal 1 | Terminal 3

```
root@:/home/jovyan# traceroute -q 1 -N 1 google.com
traceroute to google.com (172.217.1.174), 30 hops max, 60 byte packets
 1  172.16.1.2 (172.16.1.2)  0.070 ms
 2  142.104.74.92 (142.104.74.92)  0.102 ms
 3  n-gateway.csc.uvic.ca (142.104.74.65)  0.772 ms
 4  192.168.9.5 (192.168.9.5)  0.520 ms
 5  192.168.10.1 (192.168.10.1)  0.747 ms
 6  192.168.8.6 (192.168.8.6)  0.709 ms
 7  142.104.100.149 (142.104.100.149)  0.877 ms
 8  emclclebi24-oran-ix.uvic.ca (142.104.100.97)  1.333 ms
 9  362-ix-cr1-uviccle.vctr3.bc.net (207.23.244.229)  1.319 ms
10  vctr3rtr2.canarie.ca (199.212.24.98)  1.460 ms
11  sttllrtr2.canarie.ca (206.81.80.189)  4.049 ms
12  pr01-et-0-3-0-0.sea09.net.google.com (206.81.80.17)  4.326 ms
13  74.125.243.195 (74.125.243.195)  3.972 ms
14  142.250.228.152 (142.250.228.152)  10.823 ms
15  *
16  *
17  172.253.67.239 (172.253.67.239)  57.063 ms
18  108.170.250.225 (108.170.250.225)  56.805 ms
19  108.170.226.223 (108.170.226.223)  56.723 ms
20  yyz10s04-in-f14.1e100.net (172.217.1.174)  56.943 ms
root@:/home/jovyan#
```

Terminal 2 | Terminal 4

```
pan@jupyter:~$ sudo tcpdump -i host-nat icmp or udp and not port 53 -Z root -w /tmp/traceroute.cap
tcpdump: listening on host-nat, link-type EN10MB (Ethernet), capture size 262144 bytes
^C38 packets captured
38 packets received by filter
0 packets dropped by kernel
pan@jupyter:~$ mv /tmp/traceroute.cap .
mv: failed to preserve ownership for './traceroute.cap': Permission denied
mv: cannot remove '/tmp/traceroute.cap': Operation not permitted
pan@jupyter:~$ tcpdump -n -l -r traceroute.cap
reading from file traceroute.cap, link-type EN10MB (Ethernet)
20:33:13.638062 IP 172.16.1.1.57822 > 172.217.1.174.33434: UDP, length 32
20:33:13.638113 IP 172.16.1.2 > 172.16.1.1: ICMP time exceeded in-transit, length 68
20:33:13.638664 IP 172.16.1.1.51147 > 172.217.1.174.33435: UDP, length 32
20:33:13.638754 IP 142.104.74.92 > 172.16.1.1: ICMP time exceeded in-transit, length 68
20:33:13.640335 IP 172.16.1.1.56855 > 172.217.1.174.33436: UDP, length 32
20:33:13.641091 IP 142.104.74.65 > 172.16.1.1: ICMP time exceeded in-transit, length 36
20:33:13.642354 IP 172.16.1.1.40066 > 172.217.1.174.33437: UDP, length 32
20:33:13.642854 IP 192.168.9.5 > 172.16.1.1: ICMP time exceeded in-transit, length 36
20:33:13.643494 IP 172.16.1.1.57906 > 172.217.1.174.33438: UDP, length 32
20:33:13.644224 IP 192.168.10.1 > 172.16.1.1: ICMP time exceeded in-transit, length 36
20:33:13.644774 IP 172.16.1.1.55414 > 172.217.1.174.33439: UDP, length 32
20:33:13.645470 IP 192.168.8.6 > 172.16.1.1: ICMP time exceeded in-transit, length 36
20:33:18.798857 IP 172.16.1.1.46257 > 172.217.1.174.33440: UDP, length 32
20:33:18.799710 IP 142.104.100.149 > 172.16.1.1: ICMP time exceeded in-transit, length 36
20:33:18.800522 IP 172.16.1.1.48769 > 172.217.1.174.33441: UDP, length 32
20:33:18.801831 IP 142.104.100.97 > 172.16.1.1: ICMP time exceeded in-transit, length 36
20:33:18.802822 IP 172.16.1.1.60811 > 172.217.1.174.33442: UDP, length 32
20:33:18.804119 IP 207.23.244.229 > 172.16.1.1: ICMP time exceeded in-transit, length 36
20:33:18.805117 IP 172.16.1.1.46703 > 172.217.1.174.33443: UDP, length 32
20:33:18.806558 IP 199.212.24.98 > 172.16.1.1: ICMP time exceeded in-transit, length 36
20:33:18.864667 IP 172.16.1.1.38859 > 172.217.1.174.33444: UDP, length 32
20:33:18.868686 IP 206.81.80.189 > 172.16.1.1: ICMP time exceeded in-transit, length 36
20:33:18.869753 IP 172.16.1.1.48042 > 172.217.1.174.33445: UDP, length 32
20:33:18.874021 IP 206.81.80.17 > 172.16.1.1: ICMP time exceeded in-transit, length 36
20:33:18.874965 IP 172.16.1.1.55662 > 172.217.1.174.33446: UDP, length 32
20:33:18.878913 IP 74.125.243.195 > 172.16.1.1: ICMP time exceeded in-transit, length 36
20:33:18.890371 IP 172.16.1.1.39890 > 172.217.1.174.33447: UDP, length 32
20:33:18.901167 IP 142.250.228.152 > 172.16.1.1: ICMP time exceeded in-transit, length 148
20:33:25.977084 IP 172.16.1.1.54088 > 172.217.1.174.33448: UDP, length 32
20:33:30.982750 IP 172.16.1.1.37528 > 172.217.1.174.33449: UDP, length 32
20:33:35.985131 IP 172.16.1.1.53149 > 172.217.1.174.33450: UDP, length 32
20:33:36.042168 IP 172.253.67.239 > 172.16.1.1: ICMP time exceeded in-transit, length 36
20:33:43.119526 IP 172.16.1.1.41808 > 172.217.1.174.33451: UDP, length 32
```

4 0 Terminal 4

On R (which has routed access to the Internet)

```
root@r:/home/jovyan# traceroute -q 1 -N 1 google.com
traceroute to google.com (172.217.1.174), 30 hops max, 60 byte
packets
 1  172.16.1.2 (172.16.1.2)  0.070 ms
 2  142.104.74.92 (142.104.74.92)  0.102 ms
 3  n-gateway.csc.uvic.ca (142.104.74.65)  0.772 ms
 4  192.168.9.5 (192.168.9.5)  0.520 ms
 5  192.168.10.1 (192.168.10.1)  0.747 ms
 6  192.168.8.6 (192.168.8.6)  0.709 ms
 7  142.104.100.149 (142.104.100.149)  0.877 ms
 8  emc1clebi24-oran-ix.uvic.ca (142.104.100.97)  1.333 ms
 9  362-ix-cr1-uviccle.vctr3.bc.net (207.23.244.229)  1.319 ms
10  vctr3rtr2.canarie.ca (199.212.24.98)  1.460 ms
11  sttl1rtr2.canarie.ca (206.81.80.189)  4.049 ms
12  pr01-et-0-3-0-0.sea09.net.google.com (206.81.80.17)  4.326 ms
13  74.125.243.195 (74.125.243.195)  3.972 ms
14  142.250.228.152 (142.250.228.152)  10.823 ms
15  *
16  *
17  172.253.67.239 (172.253.67.239)  57.063 ms
18  108.170.250.225 (108.170.250.225)  56.805 ms
19  108.170.226.223 (108.170.226.223)  56.723 ms
20  yyz10s04-in-f14.1e100.net (172.217.1.174)  56.943 ms
```

On Jupyter where the packets will go through

```
pan@jupyter:~$ sudo tcpdump -i host-nat icmp or udp and not port 53 -Z root -w
/tmp/traceroute.cap
tcpdump: listening on host-nat, link-type EN10MB (Ethernet), capture size 262144
bytes
^C38 packets captured
38 packets received by filter
0 packets dropped by kernel
pan@jupyter:~$ mv /tmp/traceroute.cap .
mv: failed to preserve ownership for './traceroute.cap': Permission denied
mv: cannot remove '/tmp/traceroute.cap': Operation not permitted
pan@jupyter:~$ tcpdump -n -l -r traceroute.cap
reading from file traceroute.cap, link-type EN10MB (Ethernet)
20:33:13.638062 IP 172.16.1.1.57822 > 172.217.1.174.33434: UDP, length 32
20:33:13.638113 IP 172.16.1.2 > 172.16.1.1: ICMP time exceeded in-transit, length 68
20:33:13.638664 IP 172.16.1.1.51147 > 172.217.1.174.33435: UDP, length 32
20:33:13.638754 IP 142.104.74.92 > 172.16.1.1: ICMP time exceeded in-transit, length
68
20:33:13.640335 IP 172.16.1.1.56855 > 172.217.1.174.33436: UDP, length 32
20:33:13.641091 IP 142.104.74.65 > 172.16.1.1: ICMP time exceeded in-transit, length
36
20:33:13.642354 IP 172.16.1.1.40066 > 172.217.1.174.33437: UDP, length 32
20:33:13.642854 IP 192.168.9.5 > 172.16.1.1: ICMP time exceeded in-transit, length
36
20:33:13.643494 IP 172.16.1.1.57906 > 172.217.1.174.33438: UDP, length 32
20:33:13.644224 IP 192.168.10.1 > 172.16.1.1: ICMP time exceeded in-transit, length
36
```

Get to Google

```
20:33:13.644774 IP 172.16.1.1.55414 > 172.217.1.174.33439: UDP, length 32
20:33:13.645470 IP 192.168.8.6 > 172.16.1.1: ICMP time exceeded in-transit, length
36
20:33:18.798857 IP 172.16.1.1.46257 > 172.217.1.174.33440: UDP, length 32
20:33:18.799710 IP 142.104.100.149 > 172.16.1.1: ICMP time exceeded in-transit,
length 36
20:33:18.800522 IP 172.16.1.1.48769 > 172.217.1.174.33441: UDP, length 32
20:33:18.801831 IP 142.104.100.97 > 172.16.1.1: ICMP time exceeded in-transit,
length 36
20:33:18.802822 IP 172.16.1.1.60811 > 172.217.1.174.33442: UDP, length 32
20:33:18.804119 IP 207.23.244.229 > 172.16.1.1: ICMP time exceeded in-transit,
length 36
20:33:18.805117 IP 172.16.1.1.46703 > 172.217.1.174.33443: UDP, length 32
20:33:18.806558 IP 199.212.24.98 > 172.16.1.1: ICMP time exceeded in-transit, length
36
20:33:18.864667 IP 172.16.1.1.38859 > 172.217.1.174.33444: UDP, length 32
20:33:18.868686 IP 206.81.80.189 > 172.16.1.1: ICMP time exceeded in-transit, length
36
20:33:18.869753 IP 172.16.1.1.48042 > 172.217.1.174.33445: UDP, length 32
20:33:18.874021 IP 206.81.80.17 > 172.16.1.1: ICMP time exceeded in-transit, length
36
```

From Google Seattle to Google Toronto: why?

```
20:33:18.874965 IP 172.16.1.1.55662 > 172.217.1.174.33446: UDP, length 32
20:33:18.878913 IP 74.125.243.195 > 172.16.1.1: ICMP time exceeded in-transit,
length 36
20:33:18.890371 IP 172.16.1.1.39890 > 172.217.1.174.33447: UDP, length 32
20:33:18.901167 IP 142.250.228.152 > 172.16.1.1: ICMP time exceeded in-transit,
length 148
20:33:25.977084 IP 172.16.1.1.54088 > 172.217.1.174.33448: UDP, length 32
20:33:30.982750 IP 172.16.1.1.37528 > 172.217.1.174.33449: UDP, length 32
20:33:35.985131 IP 172.16.1.1.53149 > 172.217.1.174.33450: UDP, length 32
20:33:36.042168 IP 172.253.67.239 > 172.16.1.1: ICMP time exceeded in-transit,
length 36
20:33:43.119526 IP 172.16.1.1.41808 > 172.217.1.174.33451: UDP, length 32
20:33:43.176306 IP 108.170.250.225 > 172.16.1.1: ICMP time exceeded in-transit,
length 68
20:33:43.277325 IP 172.16.1.1.33030 > 172.217.1.174.33452: UDP, length 32
20:33:43.334024 IP 108.170.226.223 > 172.16.1.1: ICMP time exceeded in-transit,
length 68
20:33:50.401746 IP 172.16.1.1.56245 > 172.217.1.174.33453: UDP, length 32
20:33:50.458637 IP 172.217.1.174 > 172.16.1.1: ICMP 172.217.1.174 udp port 33453
unreachable, length 36
```

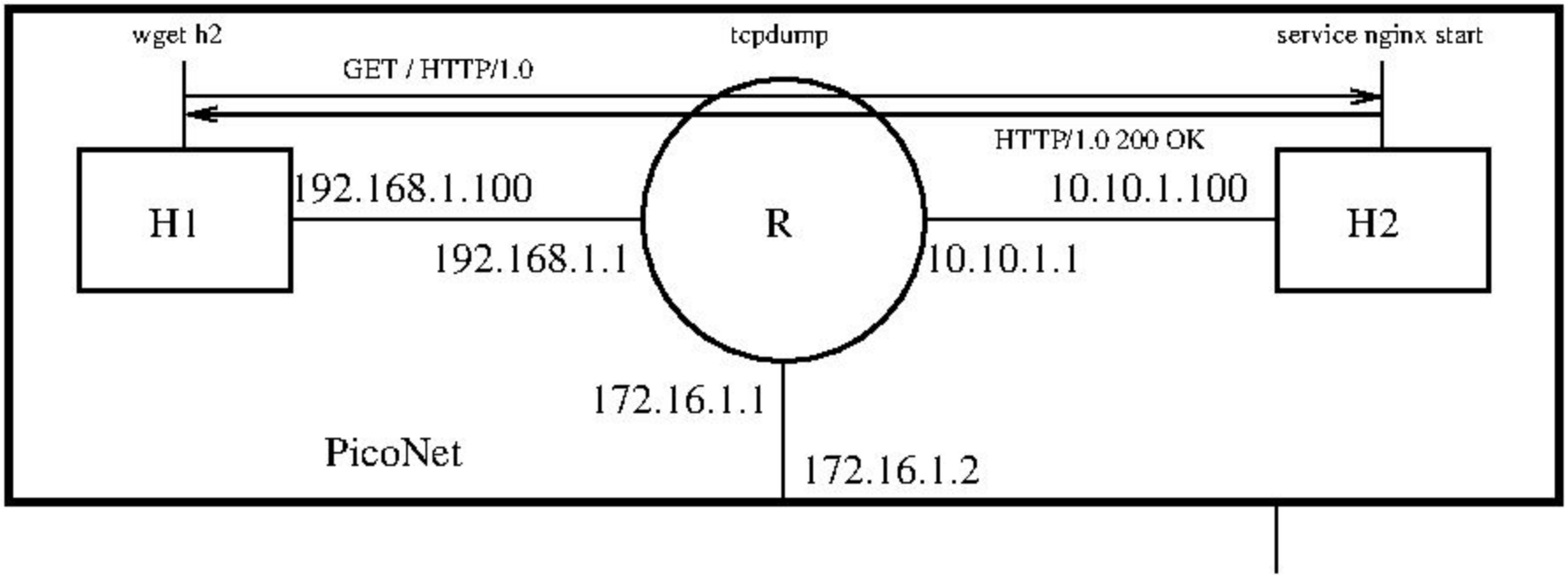
A lot of FUN in PicoNet!

PicoHub: ***our*** New CSc361 Lab platform (2020--)

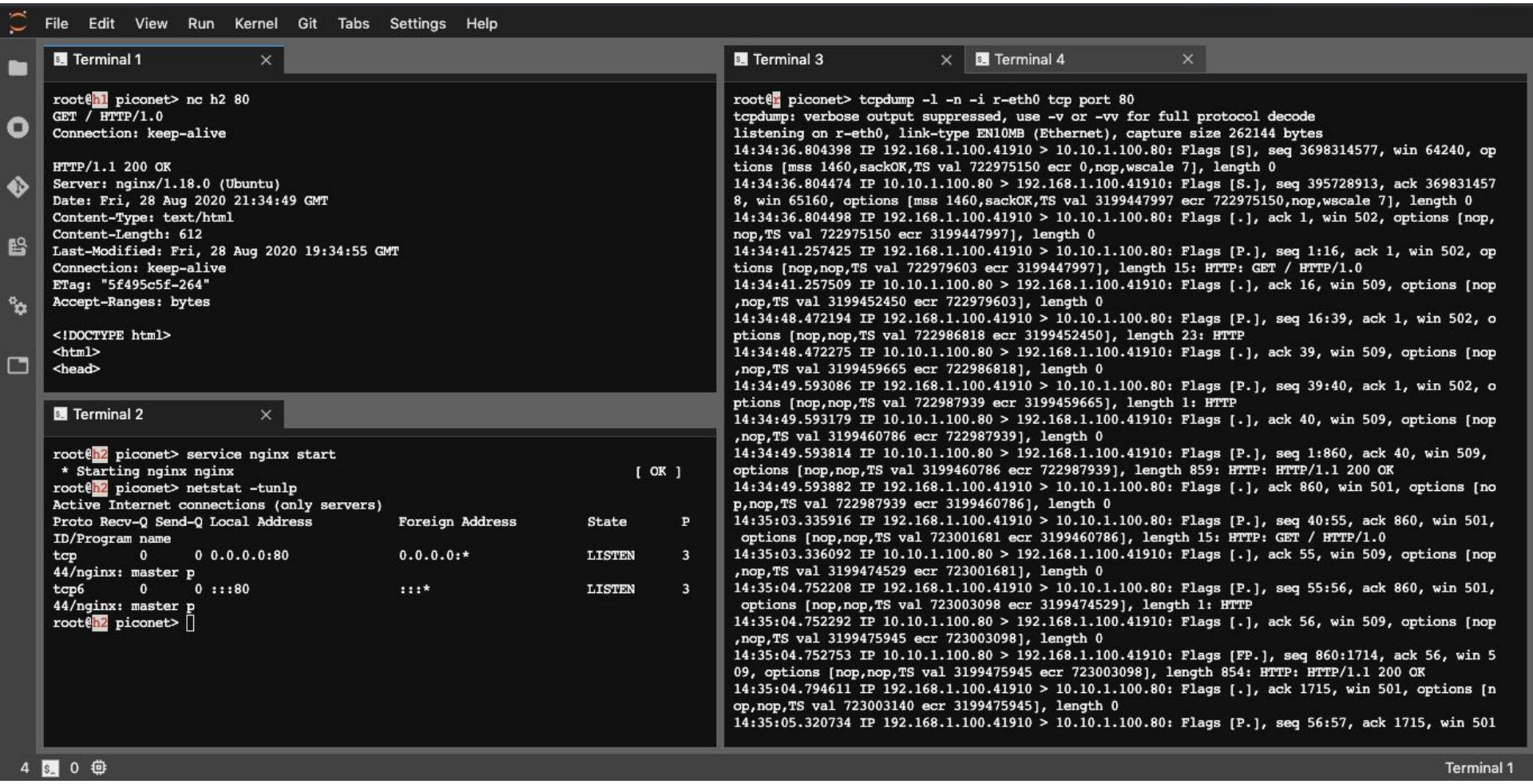
- Ready for the ***first*** full-class use in Fall 2020
  - But may still have issues to be ironed out
    - Functionality and performance issues
    - Compatibility and accessibility issues
  - So please let us know if you have any comments, questions and suggestions
    - Tell us how to reproduce the problems you encountered
    - **You can make the PicoNet/Hub better too**
  - Please be patient, as our team is working with a large class and on other things as well
- Thanks to the team!
  - Mantis: Pushed and started the transition in 2018
  - Miguel: HCI expert (advocate of JupyterLab)
  - **Tomas: Tech Support (PicoNet development/deployment/operation)**
  - Victoria: Senior Lab Instructor (CSc361 Lab/Tutorial instructor and Lab Manual developer)
  - Zhiming, Rui and Wenjun: Grad Teaching Assistants, as well as many student beta-testers!
- Contact
  - Jianping Pan <Pan@UVic.CA>: Team leader, overall design and initial prototyping/testing
  - Supported by a LTSI Course Redesign/Development grant to hire TA in Summer 2020

Extra: HTTP Lab

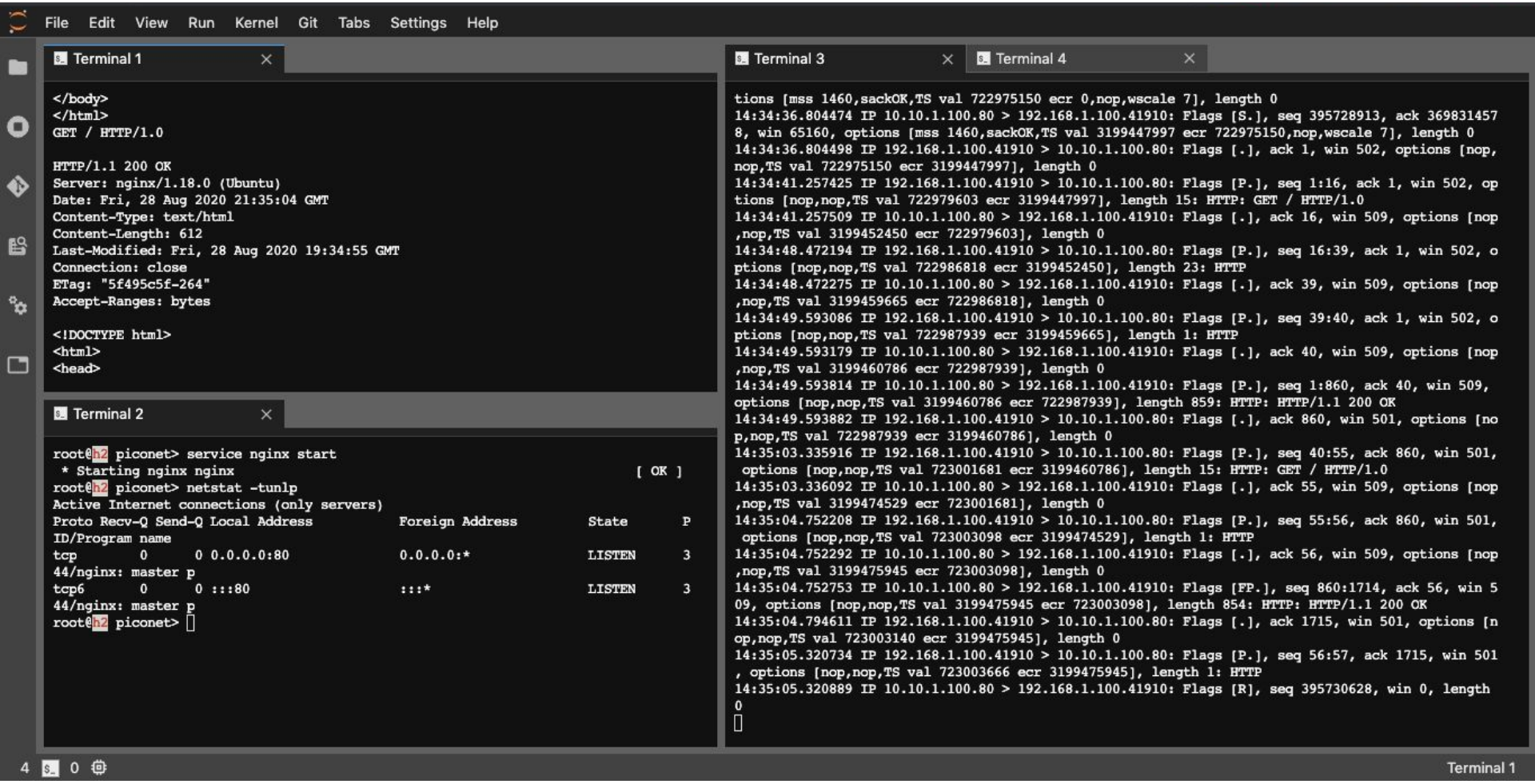
- H1: HTTP client (web browser, e.g., wget)
- H2: HTTP server (web server, e.g., nginx-light)
- R: TCPdump (packet sniffer to observe the interaction between H1 and H2)
  - Can output to screen for real-time observation or write to file for follow-up analysis



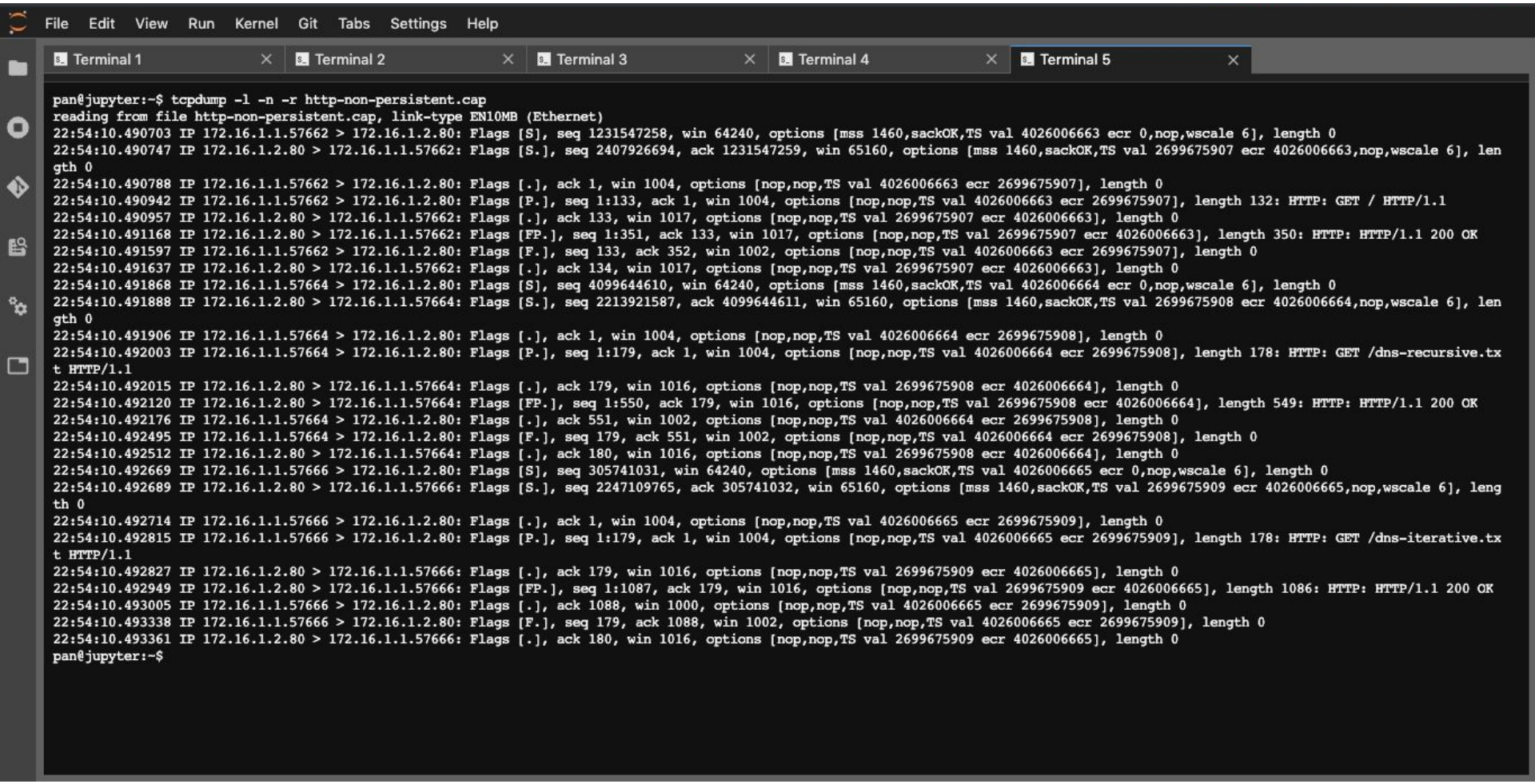
Persistent HTTP: “Connection: keep-alive”



Non-persistent HTTP connection



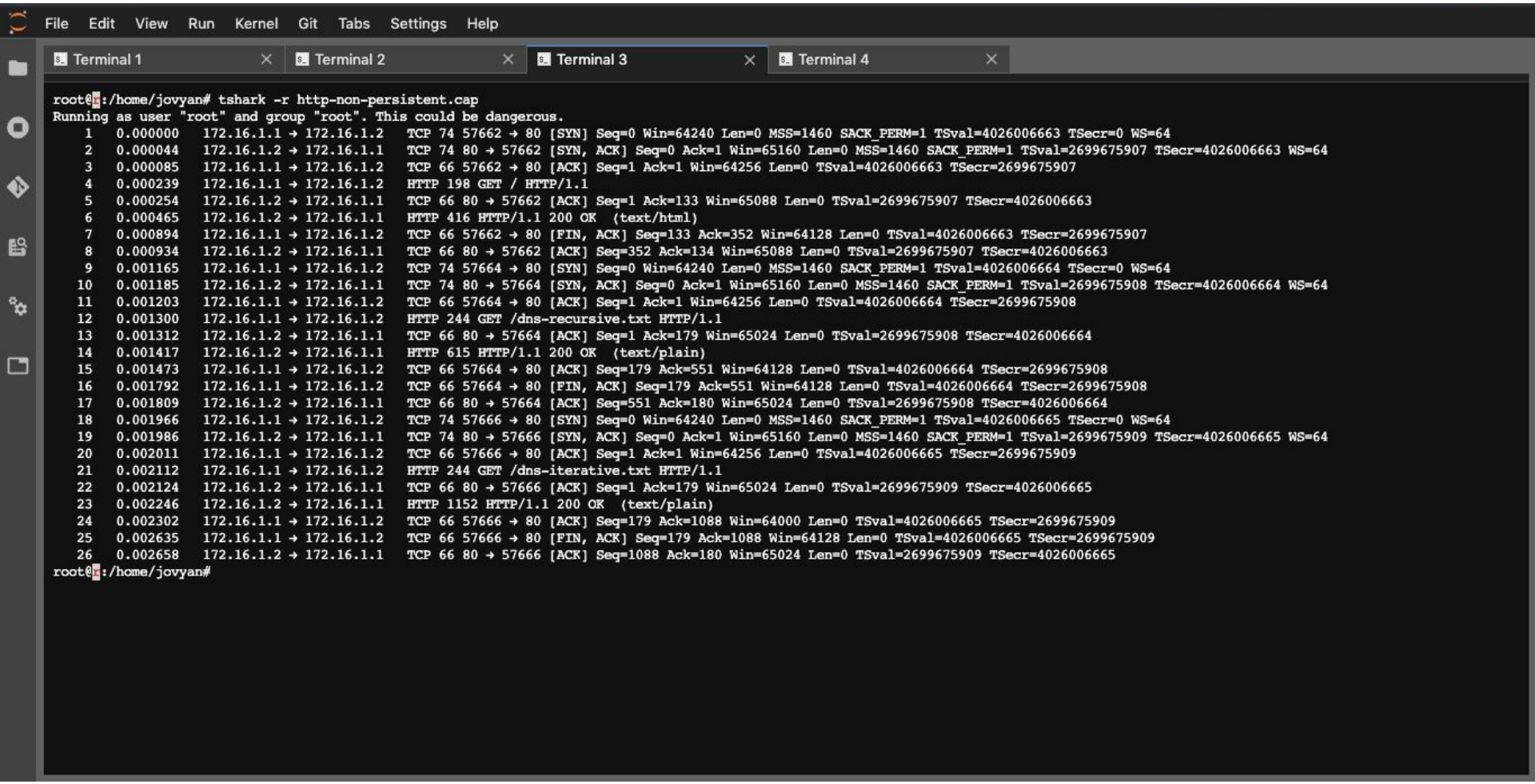
TCPdump: remote raw analysis in PicoNet



TCPdump: text output

reading from file http-non-persistent.cap, link-type EN10MB (Ethernet)
22:54:10.490703 IP 172.16.1.1.57662 > 172.16.1.2.80: Flags [**S**], seq 1231547258, win 64240, options [mss 1460,sackOK,TS val 4026006663 ecr 0,nop,wscale 6], length 0
22:54:10.490747 IP 172.16.1.2.80 > 172.16.1.1.57662: Flags [**S.**], seq 2407926694, ack 1231547259, win 65160, options [mss 1460,sackOK,TS val 2699675907 ecr 4026006663,nop,wscale 6], length 0
22:54:10.490788 IP 172.16.1.1.57662 > 172.16.1.2.80: Flags [**.**], ack 1, win 1004, options [nop,nop,TS val 4026006663 ecr 2699675907], length 0
22:54:10.490942 IP 172.16.1.1.57662 > 172.16.1.2.80: Flags [P.], seq 1:133, ack 1, win 1004, options [nop,nop,TS val 4026006663 ecr 2699675907], length 132: HTTP: **GET / HTTP/1.1**
22:54:10.490957 IP 172.16.1.2.80 > 172.16.1.1.57662: Flags [.], ack 133, win 1017, options [nop,nop,TS val 2699675907 ecr 4026006663], length 0
22:54:10.491168 IP 172.16.1.2.80 > 172.16.1.1.57662: Flags [**FP.**], seq 1:351, ack 133, win 1017, options [nop,nop,TS val 2699675907 ecr 4026006663], length 350: HTTP: **HTTP/1.1 200 OK**
22:54:10.491597 IP 172.16.1.1.57662 > 172.16.1.2.80: Flags [**F.**], seq 133, ack 352, win 1002, options [nop,nop,TS val 4026006663 ecr 2699675907], length 0
22:54:10.491637 IP 172.16.1.2.80 > 172.16.1.1.57662: Flags [**.**], ack 134, win 1017, options [nop,nop,TS val 2699675907 ecr 4026006663], length 0
22:54:10.491868 IP 172.16.1.1.57664 > 172.16.1.2.80: Flags [***S***], seq 4099644610, win 64240, options [mss 1460,sackOK,TS val 4026006664 ecr 0,nop,wscale 6], length 0
22:54:10.491888 IP 172.16.1.2.80 > 172.16.1.1.57664: Flags [***S.***], seq 2213921587, ack 4099644611, win 65160, options [mss 1460,sackOK,TS val 2699675908 ecr 4026006664,nop,wscale 6], length 0
22:54:10.491906 IP 172.16.1.1.57664 > 172.16.1.2.80: Flags [**.**], ack 1, win 1004, options [nop,nop,TS val 4026006664 ecr 2699675908], length 0
22:54:10.492003 IP 172.16.1.1.57664 > 172.16.1.2.80: Flags [P.], seq 1:179, ack 1, win 1004, options [nop,nop,TS val 4026006664 ecr 2699675908], length 178: HTTP: ***GET /dns-recursive.txt HTTP/1.1***

Tshark: remote summary analysis in PicoNet

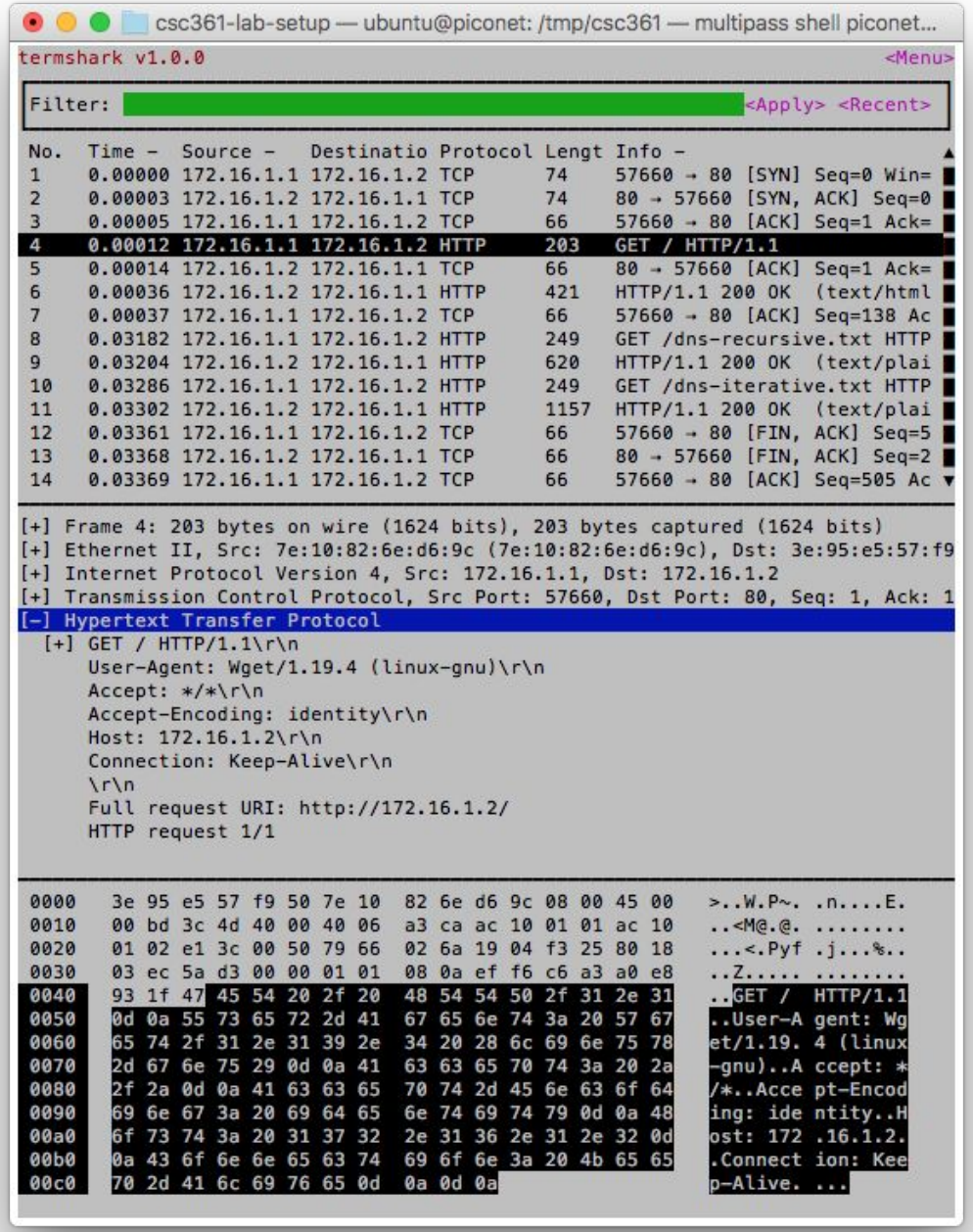


Tshark: text output

1 0.000000 172.16.1.1 → 172.16.1.2 TCP 74 57662 → 80 [**SYN**] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=4026006663 TSecr=0 WS=64
2 0.000044 172.16.1.2 → 172.16.1.1 TCP 74 80 → 57662 [**SYN, ACK**] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM=1 TSval=2699675907 TSecr=4026006663 WS=64
3 0.000085 172.16.1.1 → 172.16.1.2 TCP 66 57662 → 80 [**ACK**] Seq=1 Ack=1 Win=64256 Len=0 TSval=4026006663 TSecr=2699675907
4 0.000239 172.16.1.1 → 172.16.1.2 HTTP 198 **GET / HTTP/1.1**
5 0.000254 172.16.1.2 → 172.16.1.1 TCP 66 80 → 57662 [ACK] Seq=1 Ack=133 Win=65088 Len=0 TSval=2699675907 TSecr=4026006663
6 0.000465 172.16.1.2 → 172.16.1.1 HTTP 416 **HTTP/1.1 200 OK** (text/html)
7 0.000894 172.16.1.1 → 172.16.1.2 TCP 66 57662 → 80 [**FIN, ACK**] Seq=133 Ack=352 Win=64128 Len=0 TSval=4026006663 TSecr=2699675907
8 0.000934 172.16.1.2 → 172.16.1.1 TCP 66 80 → 57662 [**ACK**] Seq=352 Ack=134 Win=65088 Len=0 TSval=2699675907 TSecr=4026006663
9 0.001165 172.16.1.1 → 172.16.1.2 TCP 74 57664 → 80 [***SYN***] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=4026006664 TSecr=0 WS=64
10 0.001185 172.16.1.2 → 172.16.1.1 TCP 74 80 → 57664 [***SYN, ACK***] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM=1 TSval=2699675908 TSecr=4026006664 WS=64
11 0.001203 172.16.1.1 → 172.16.1.2 TCP 66 57664 → 80 [***ACK***] Seq=1 Ack=1 Win=64256 Len=0 TSval=4026006664 TSecr=2699675908
12 0.001300 172.16.1.1 → 172.16.1.2 HTTP 244 ***GET /dns-recursive.txt HTTP/1.1***
13 0.001312 172.16.1.2 → 172.16.1.1 TCP 66 80 → 57664 [ACK] Seq=1 Ack=179 Win=65024 Len=0 TSval=2699675908 TSecr=4026006664
14 0.001417 172.16.1.2 → 172.16.1.1 HTTP 615 ***HTTP/1.1 200 OK*** (text/plain)

Termshark

- Remote terminal analysis
  - Available in PicoNet already
  - Screen reader friendliness



Termshark: text output

termshark 2.0.3 | http-non-persistent.cap Analysis Misc

Filter: <Apply> <Recent>

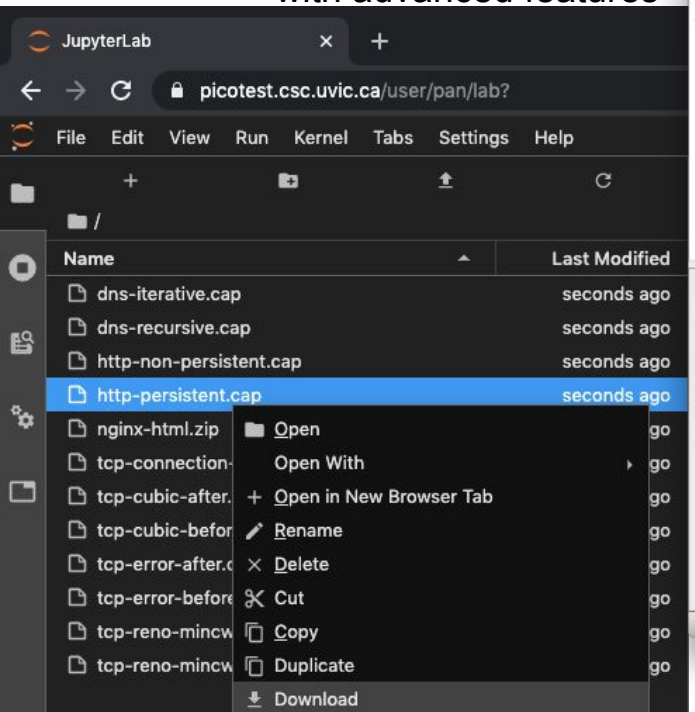
No. -	Time -	Source -	Destination -	Protocol -	Length -	Info -
1	0.000000	172.16.1.1	172.16.1.2	TCP	74	57662 → 80 [**SYN**] Seq=0
2	0.000044	172.16.1.2	172.16.1.1	TCP	74	80 → 57662 [**SYN, ACK**] Seq=0
3	0.000085	172.16.1.1	172.16.1.2	TCP	66	57662 → 80 [**ACK**] Seq=1 Ack=1
4	0.000239	172.16.1.1	172.16.1.2	HTTP	198	**GET / HTTP/1.1**
5	0.000254	172.16.1.2	172.16.1.1	TCP	66	80 → 57662 [ACK] Seq=1 Ack=133
6	0.000465	172.16.1.2	172.16.1.1	HTTP	416	**HTTP/1.1 200 OK** (text/html)
7	0.000894	172.16.1.1	172.16.1.2	TCP	66	57662 → 80 [**FIN, ACK**] Seq=133
8	0.000934	172.16.1.2	172.16.1.1	TCP	66	80 → 57662 [**ACK**] Seq=352
9	0.001165	172.16.1.1	172.16.1.2	TCP	74	57664 → 80 [***SYN***] Seq=0
10	0.001185	172.16.1.2	172.16.1.1	TCP	74	80 → 57664 [***SYN, ACK***] Seq=0

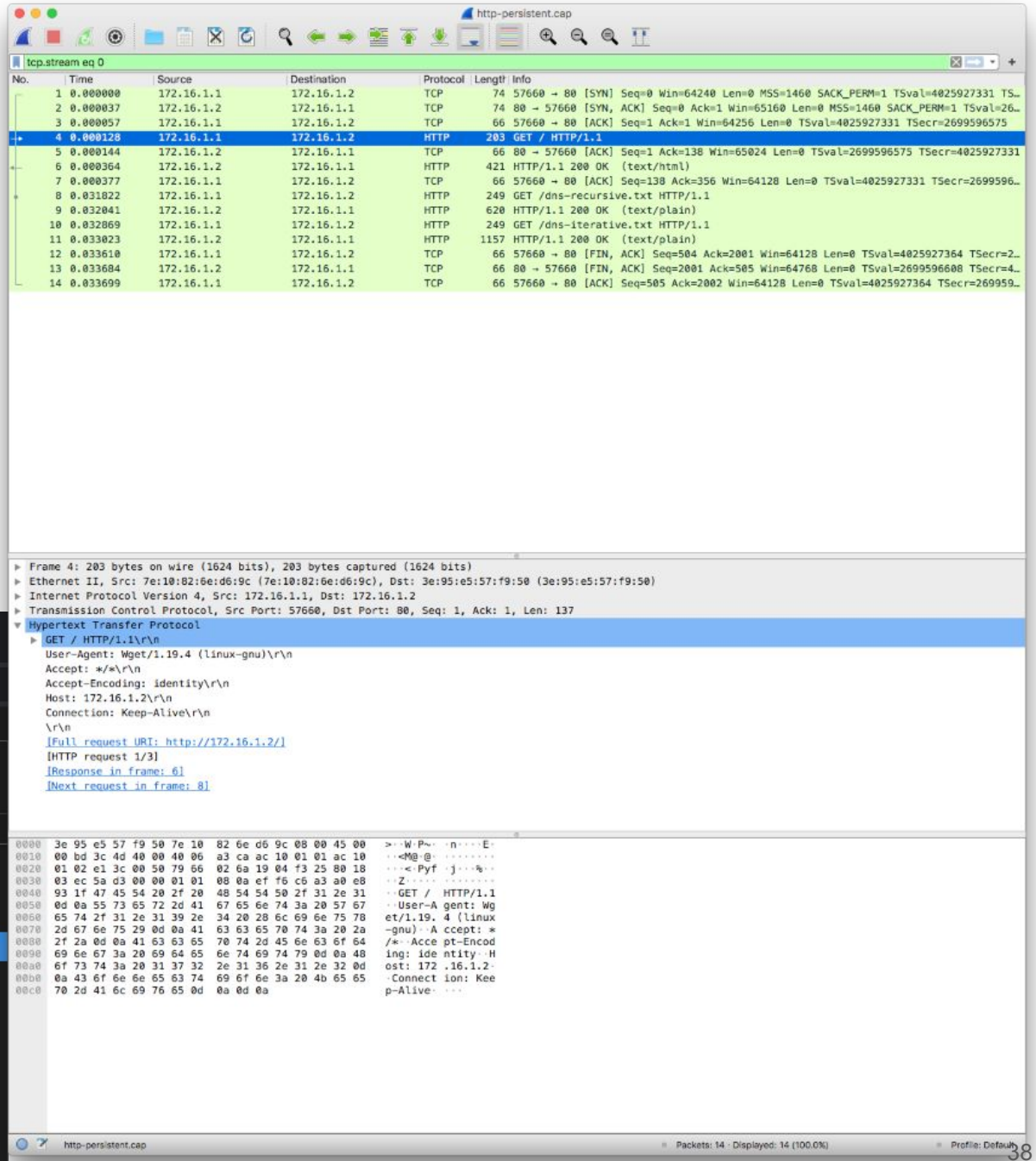
[+] Frame 4: 198 bytes on wire (1584 bits), 198 bytes captured (1584 bits)
[+] Ethernet II, Src: 7e:10:82:6e:d6:9c (7e:10:82:6e:d6:9c), Dst: 3e:95:e5:57:f9:50 (3e:95:e5:57:f9:50)
[+] Internet Protocol Version 4, Src: 172.16.1.1, Dst: 172.16.1.2
[+] Transmission Control Protocol, Src Port: 57662, Dst Port: 80, Seq: 1, Ack: 1, Len: 132
[+] **Hypertext Transfer Protocol**

```
0000   3e 95 e5 57 f9 50 7e 10  82 6e d6 9c 08 00 45 00   >..W.P~. .n....E.
0010   00 b8 8a 57 40 00 40 06  55 c5 ac 10 01 01 ac 10   ...W@.@. U.......
0020   01 02 e1 3e 00 50 49 67  eb 7b 8f 86 0b a7 80 18   ...>.PIg .{......
```

Wireshark

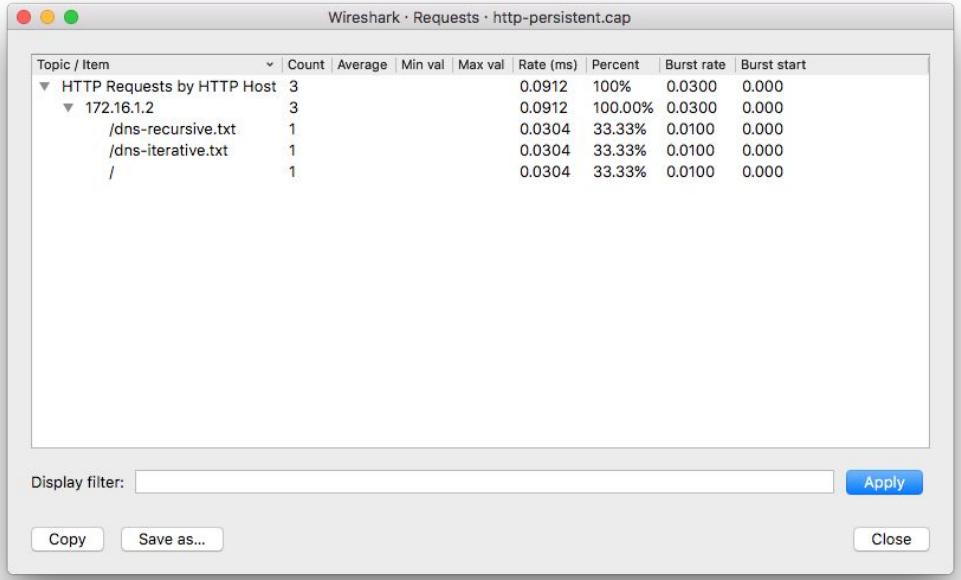
- Download the tracefile
  - through JupyterLab
- To student's laptop
  - e.g., http-persistent.cap
- To load in Wireshark
  - for detailed analysis
  - with advanced features

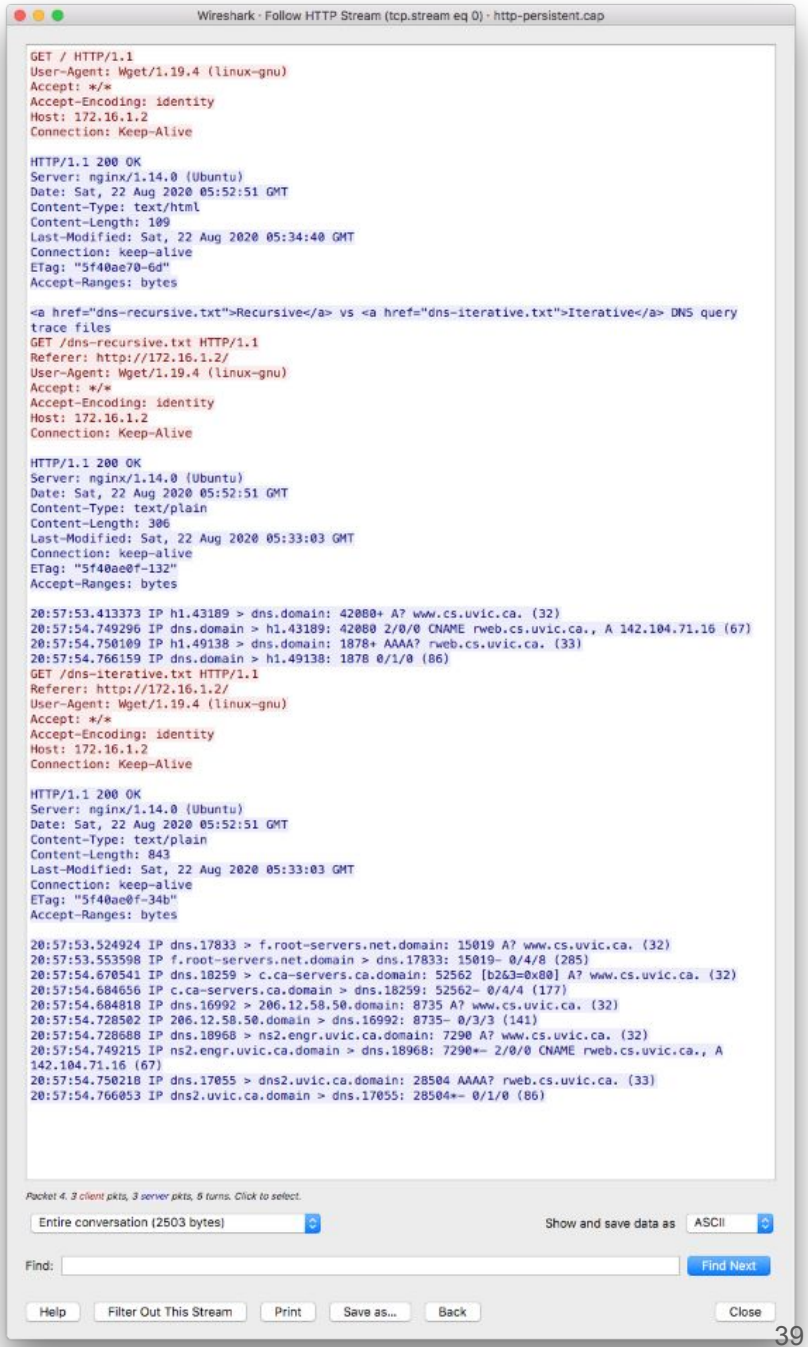




More advanced features

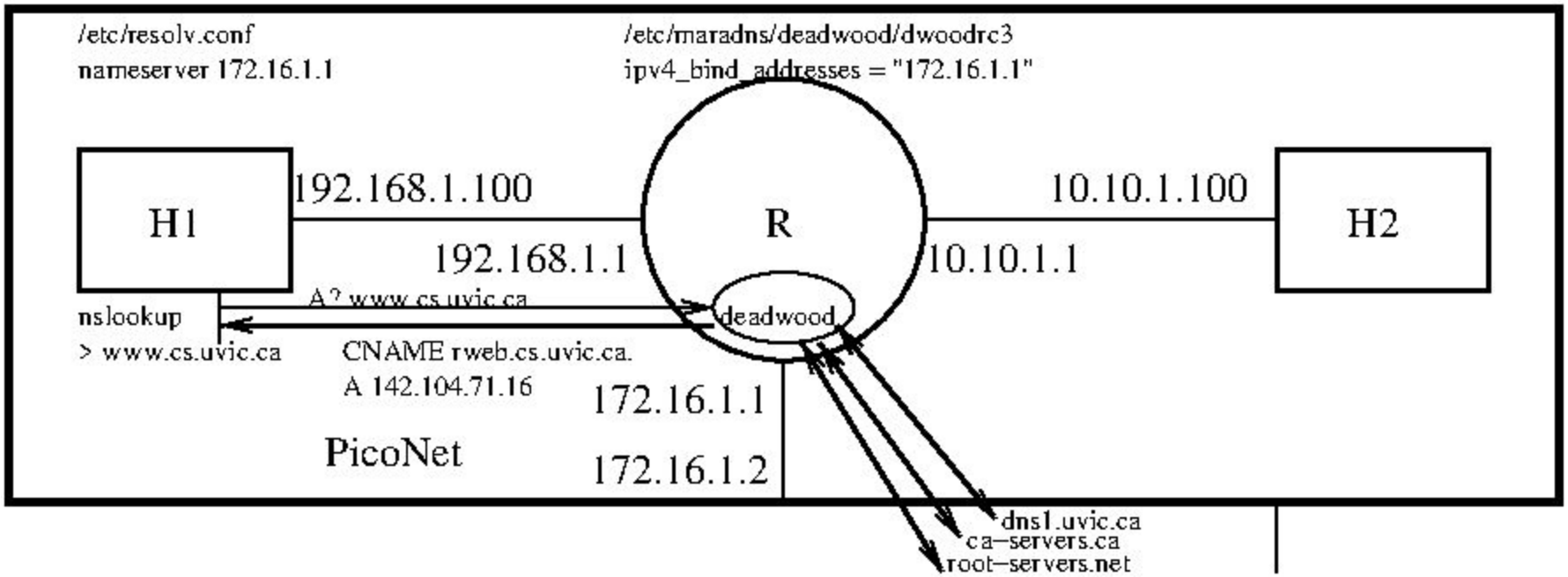
- Wireshark local analysis
  - E.g., Analyze -> Follow -> HTTP Stream
  - Statistics -> HTTP -> Requests
  - And many more



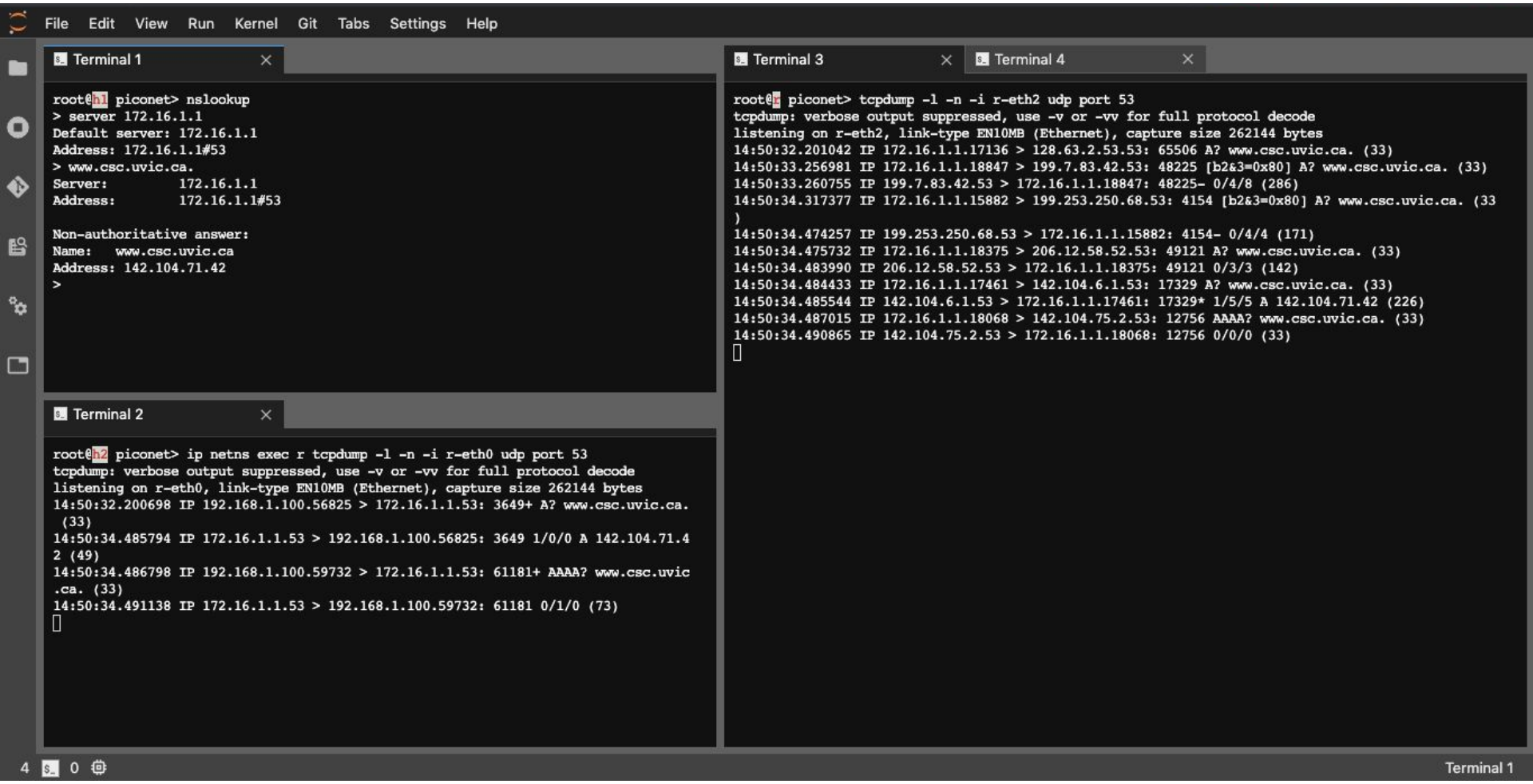


DNS Lab

- H1: DNS client (e.g., nslookup)
- R: Local DNS server (e.g., deadwood)
- Root DNS server: root-servers.net; TLD (top-level domain) DNS server: e.g., ca-servers.ca; authoritative DNS server: e.g., dns1.uvic.ca

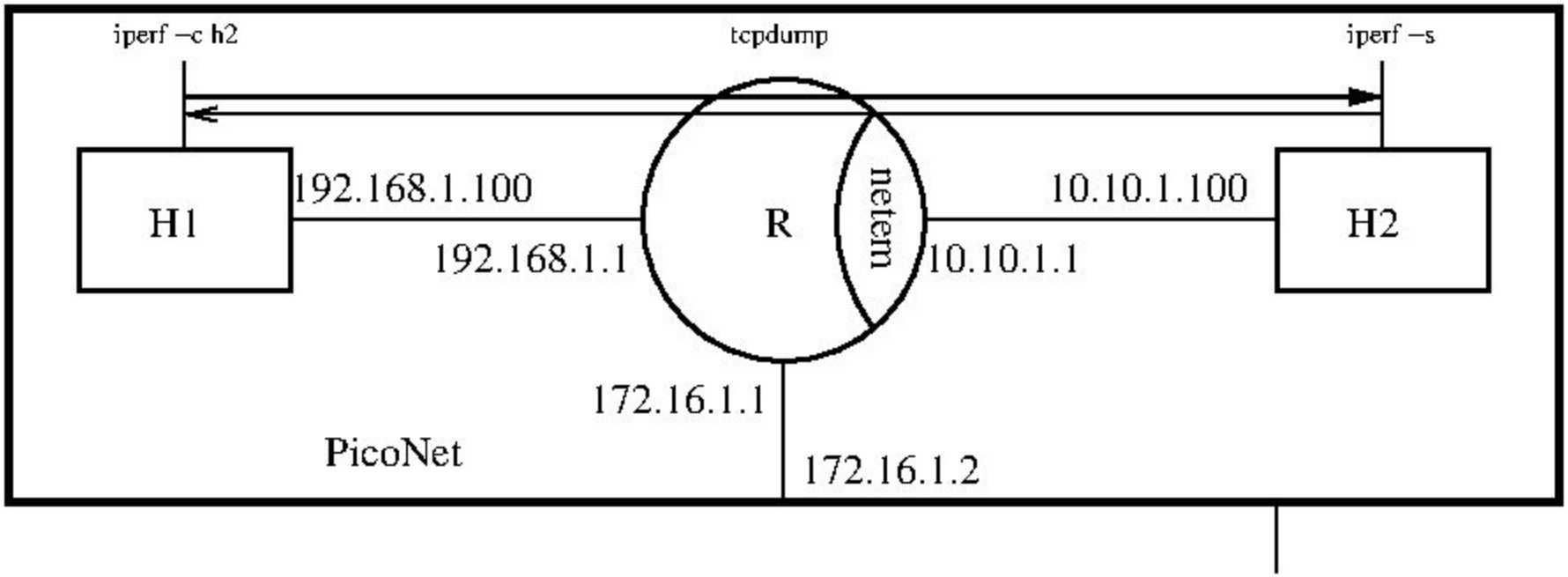


Recursive vs iterative DNS query

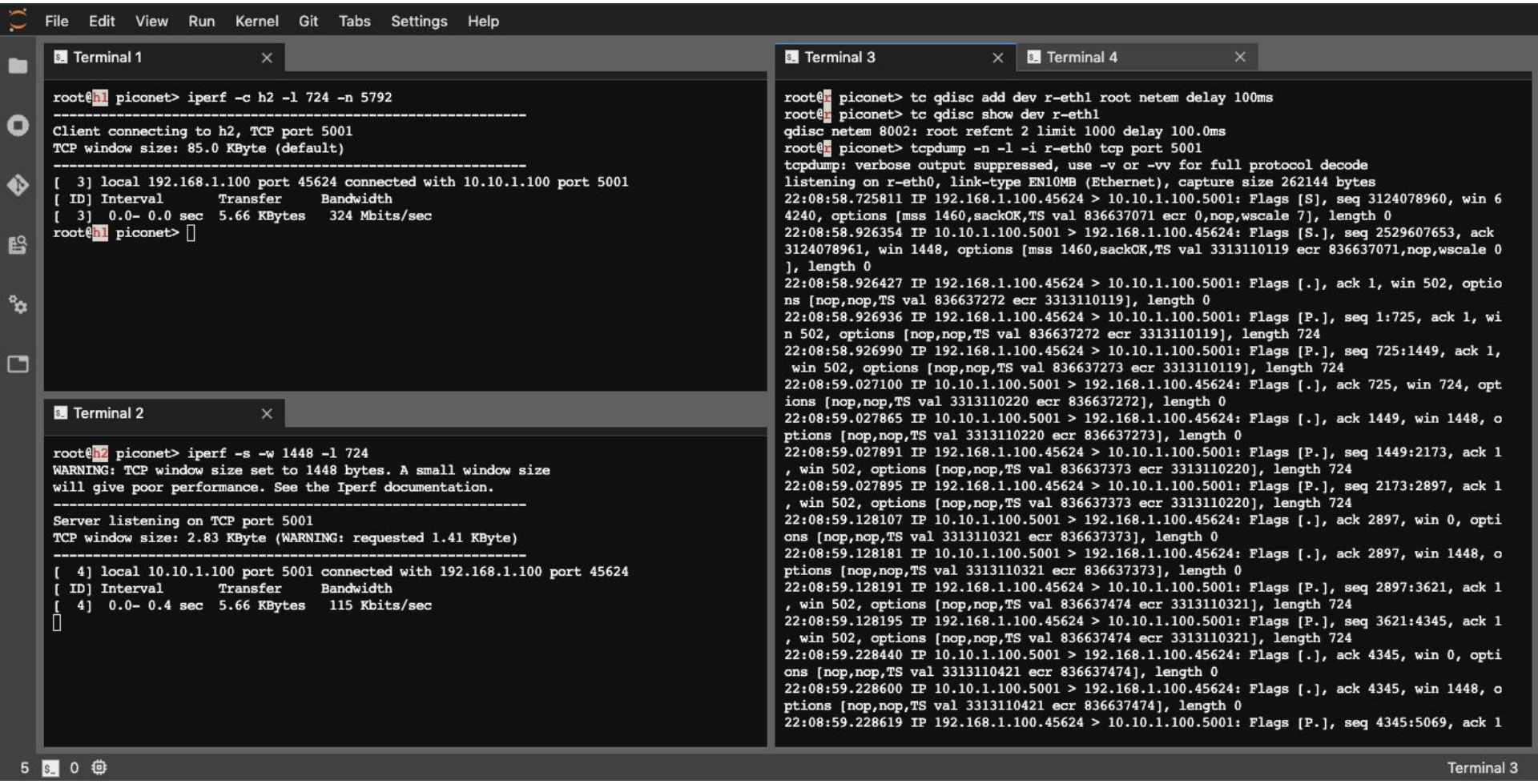


TCP Lab

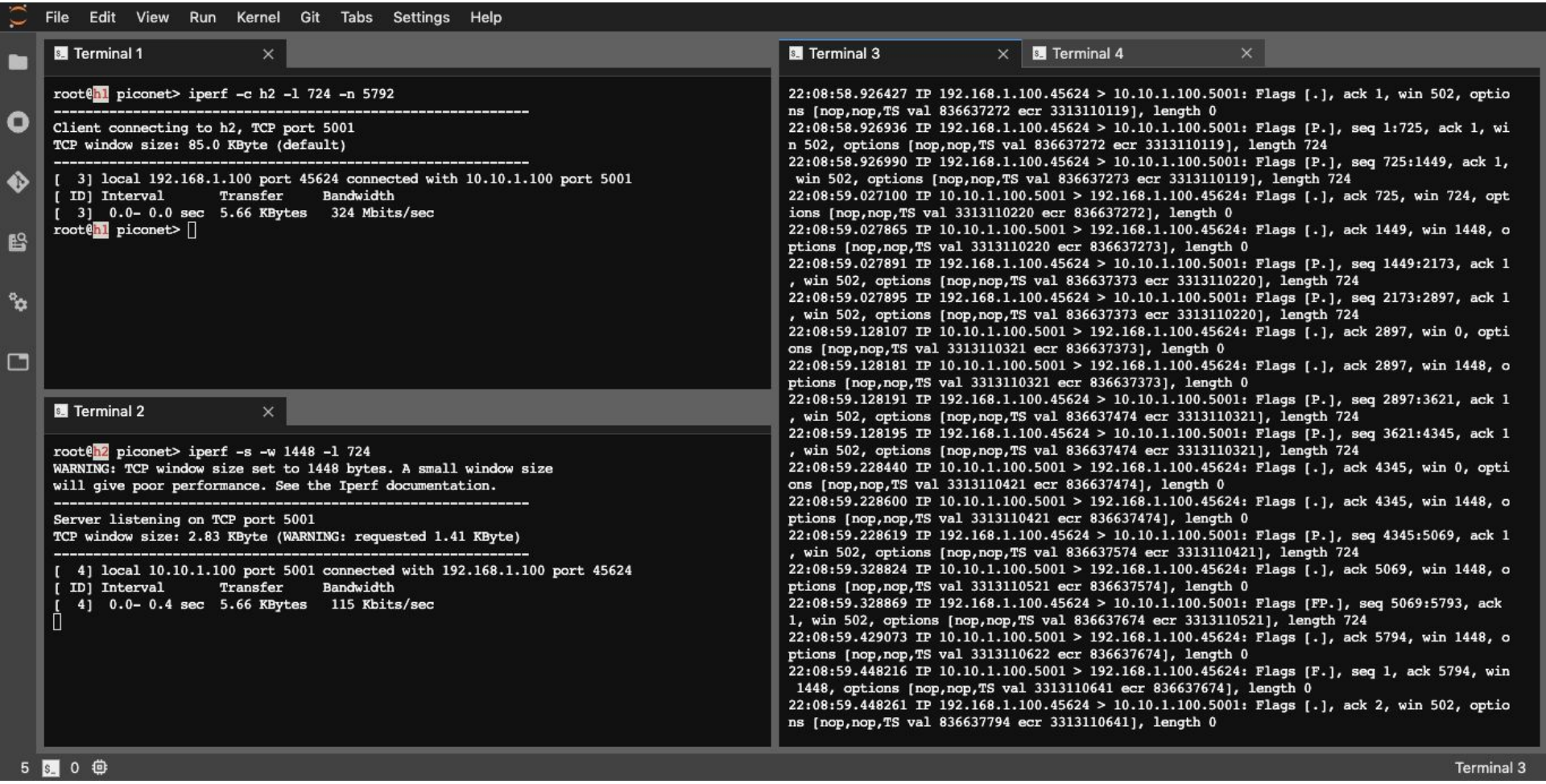
- H1: TCP connection initiator and test data sender/client (e.g., iperf -c)
- H2: TCP connection responder and test data receiver/server (e.g., iperf -s)
- R: netem to emulate packet delay, loss, duplication, corruption and reordering
  - Appear to be a real Internet (wide-area network)



TCP connection establishment (3-way handshake)

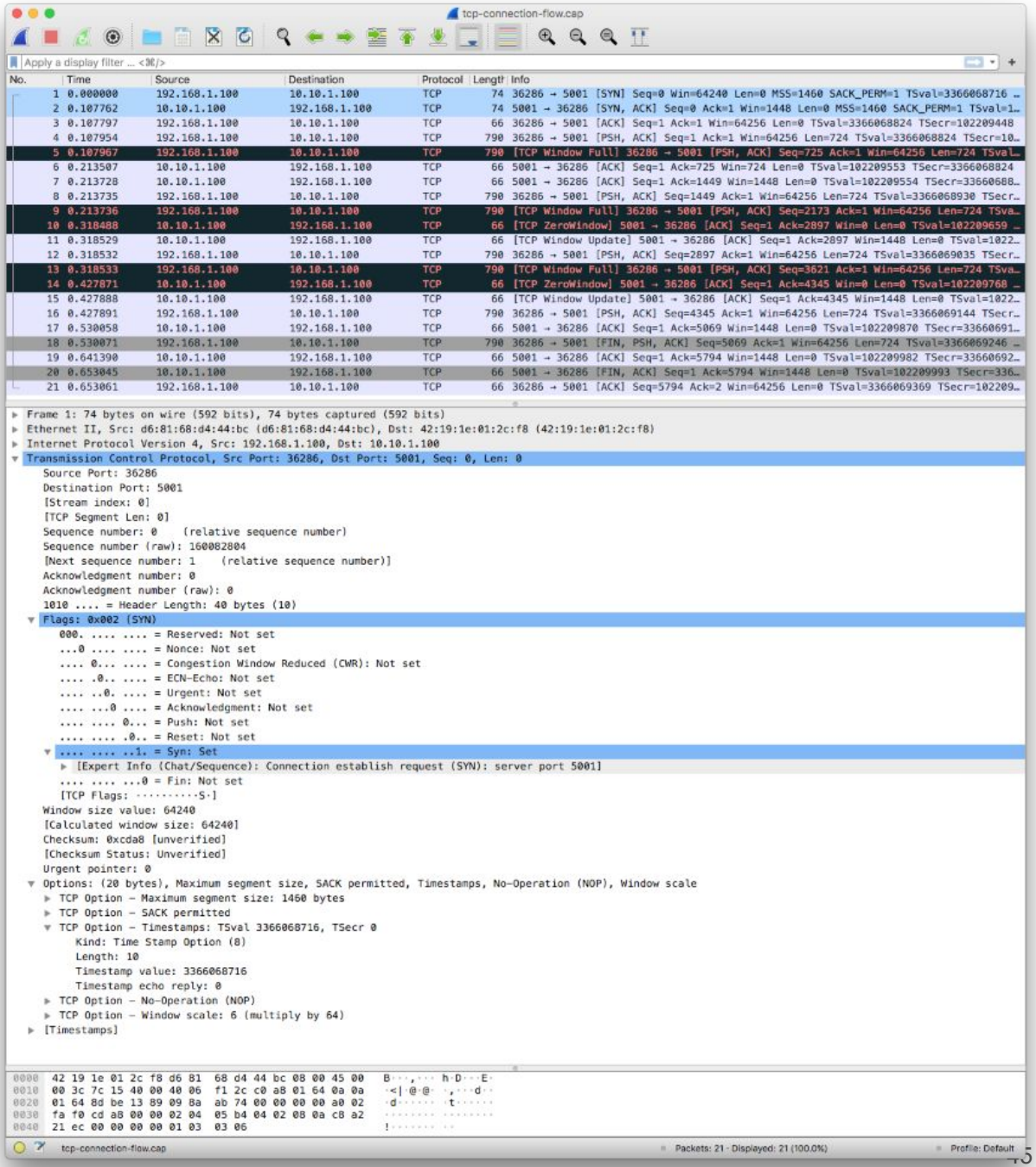


TCP connection release (4-way handshake)



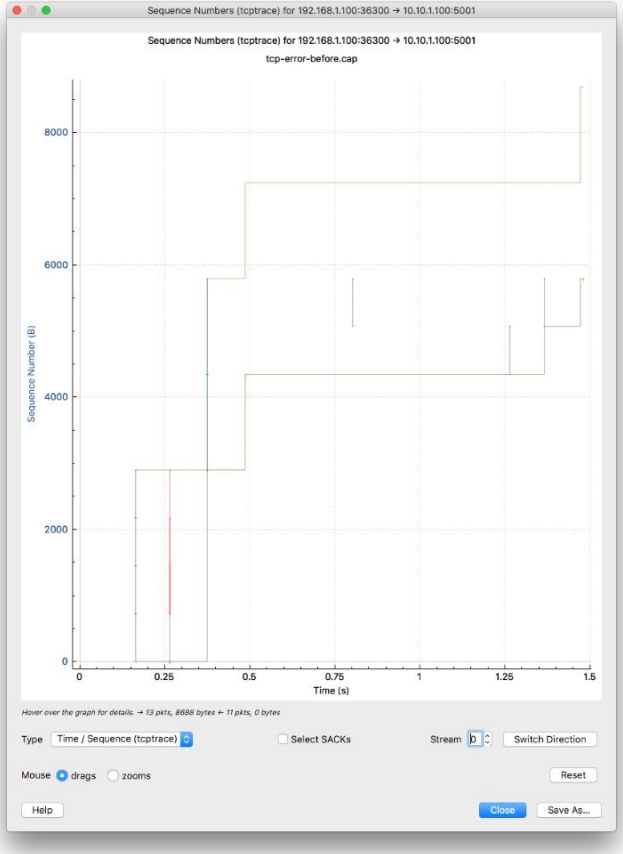
TCP flow control

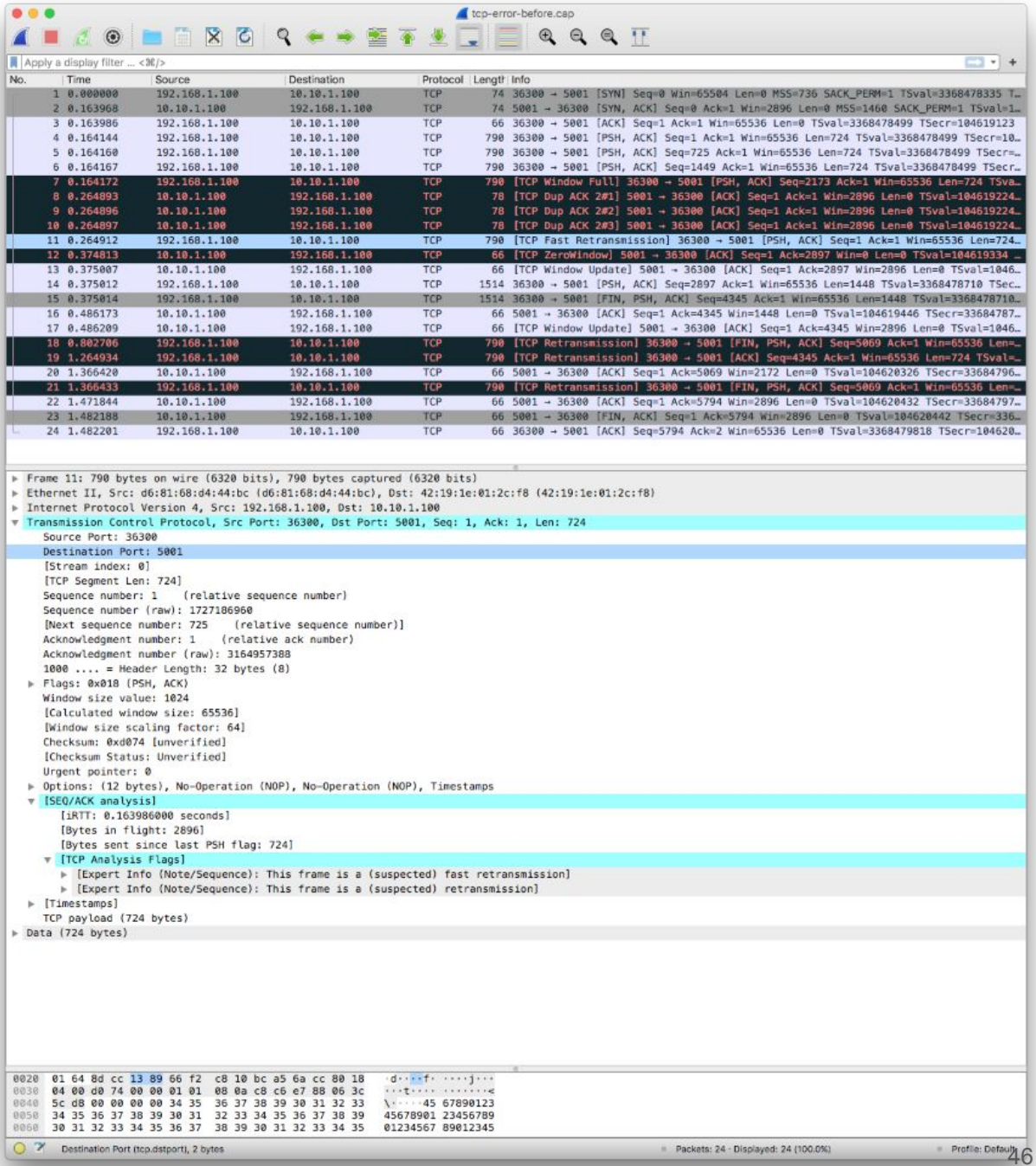
- Variable sliding window
  - “TCP Window Full”
  - “TCP Zero Window”
  - “TCP Window Update”
  - ...



TCP error control

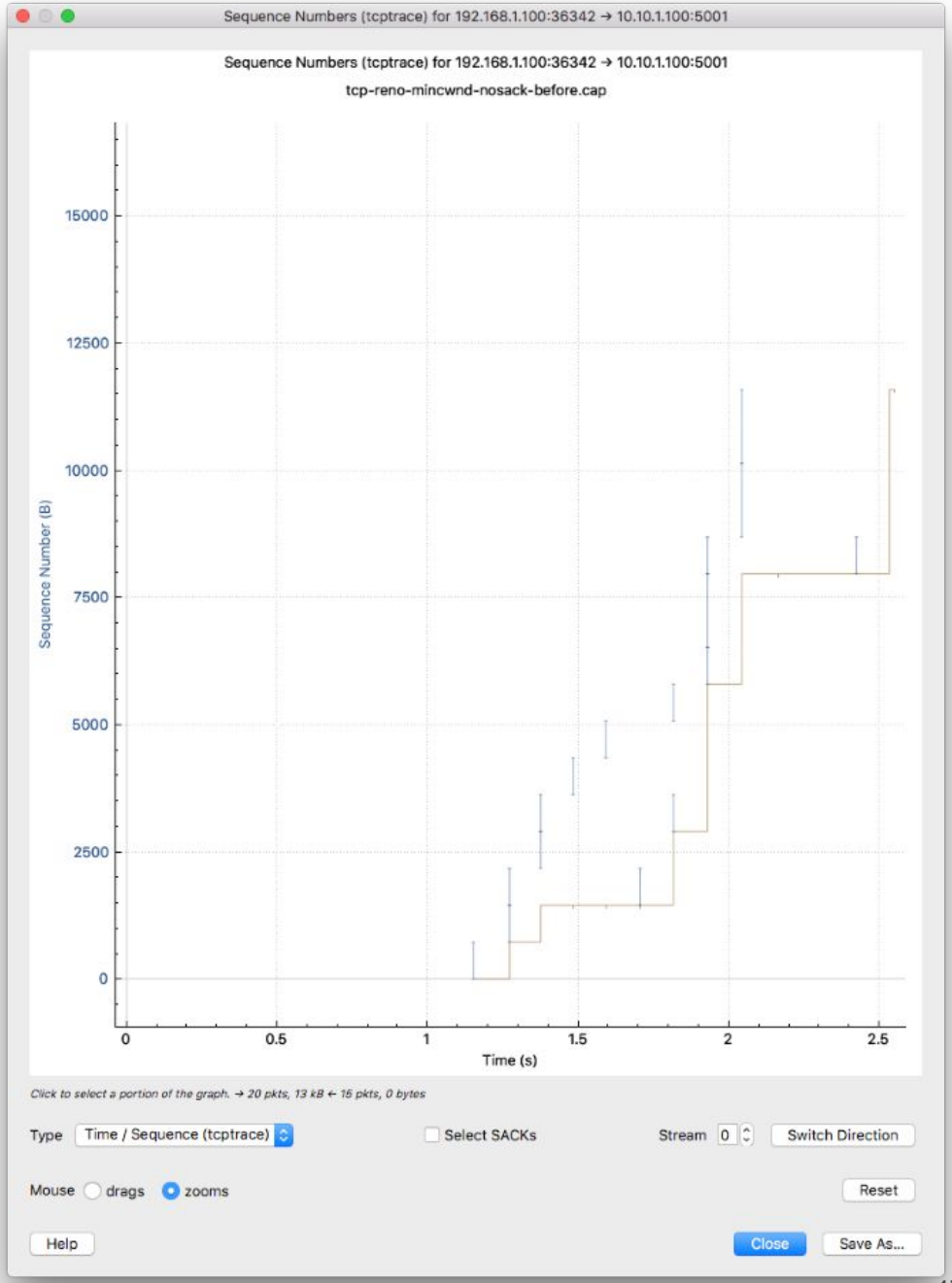
- Loss vs retransmission
  - “TCP Dup ACK”
  - “Fast Retransmission”
  - “TCP Retransmission”





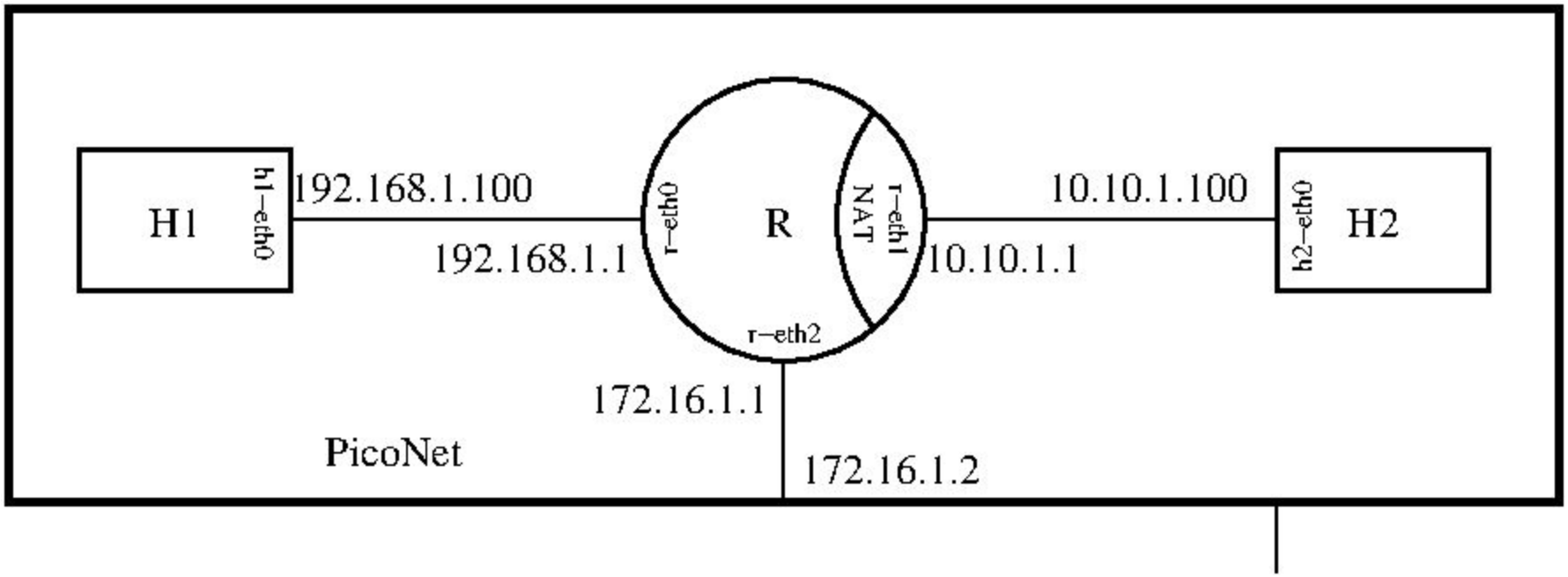
TCP congestion control

- What keeps the Internet still alive
  - “Slow Start”
  - “Congestion Avoidance”
  - “Timeout Retransmit”
  - “Fast Retransmit”
  - “Fast Recovery”
  - And many more...

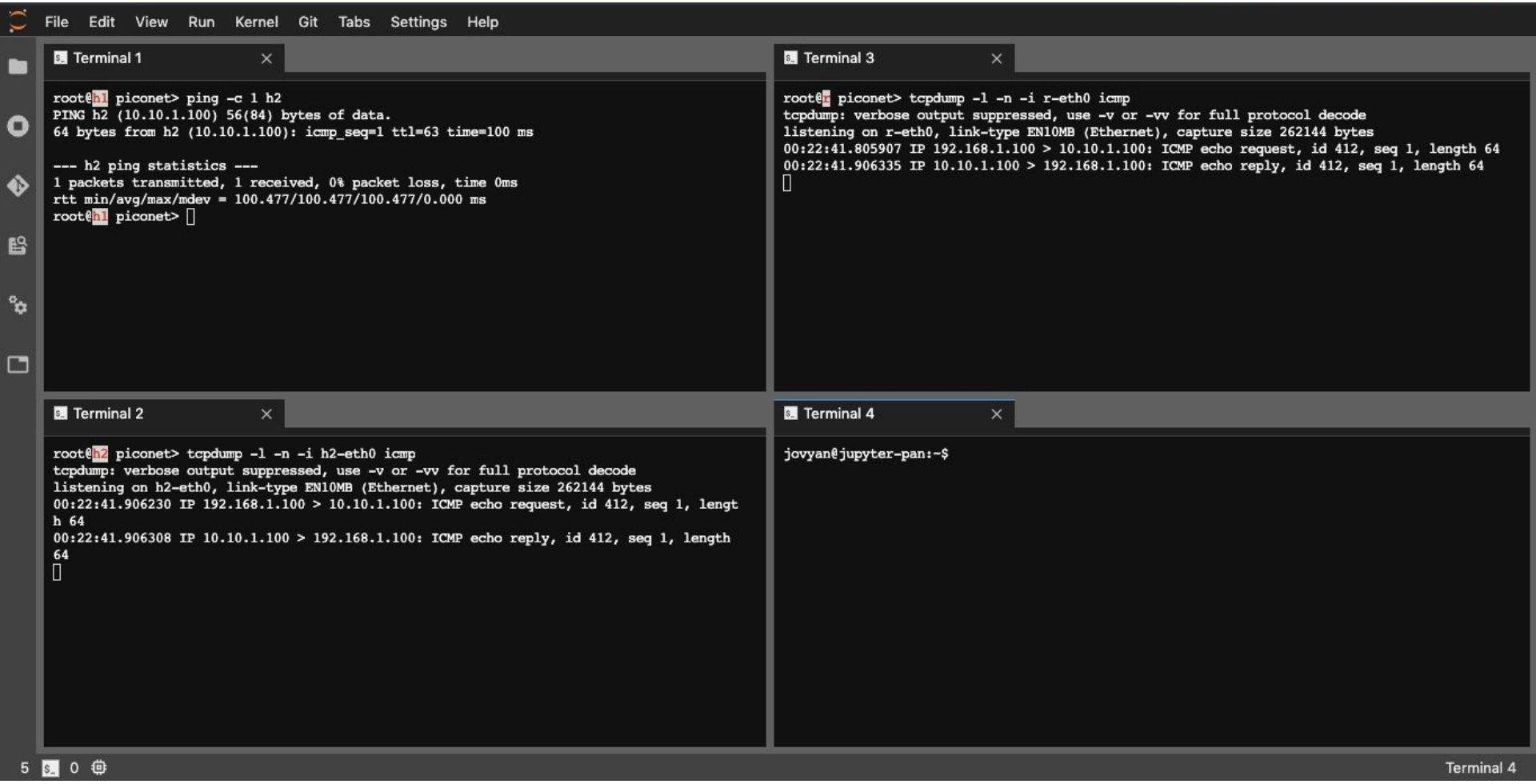


NAT (network address/translation) Lab

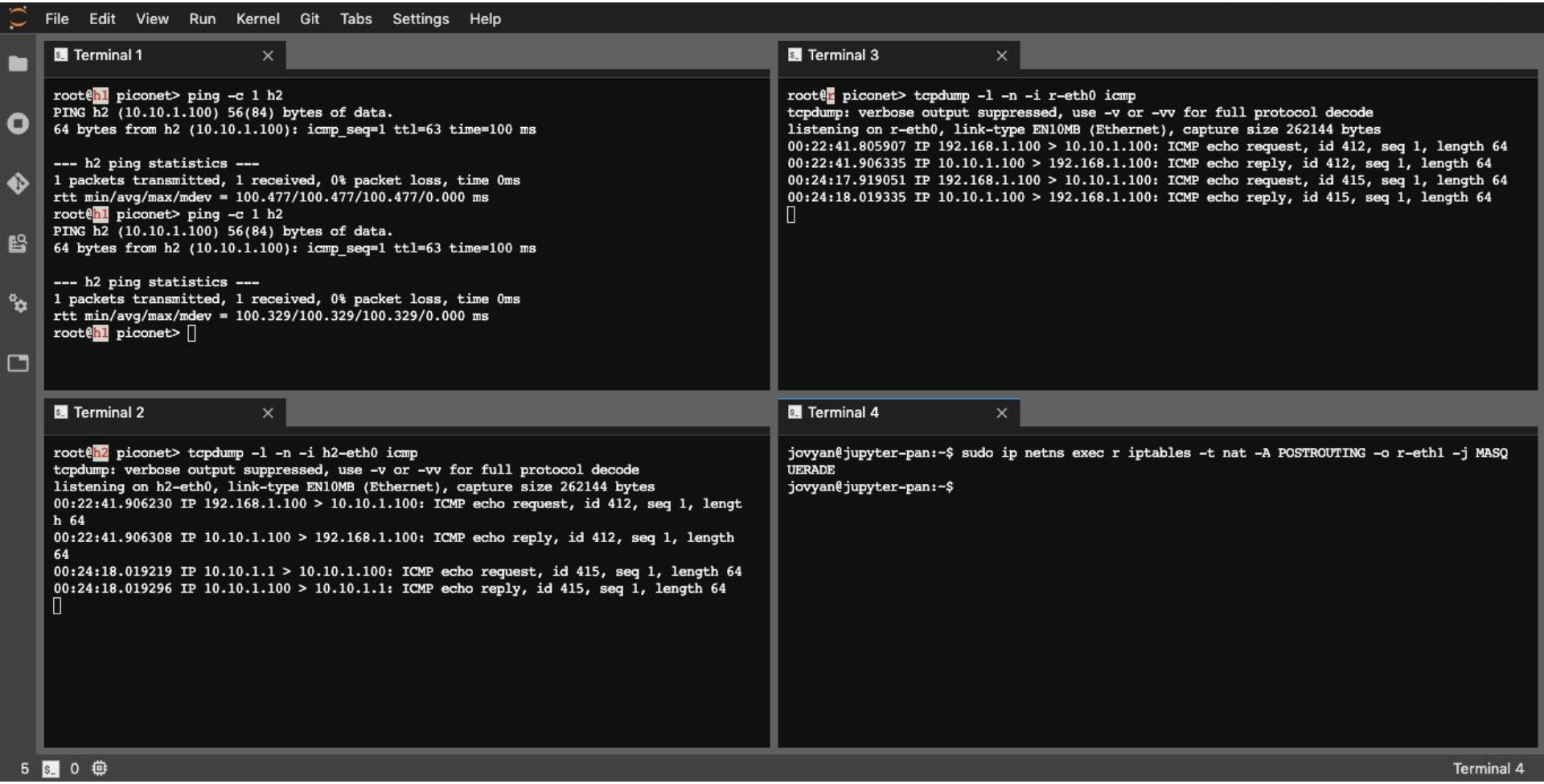
- H1: one of your home computers
- H2: the server on the Internet
- R: your home router, most likely running a network address translator too
  - Only need to pay for one public IP address to your Internet service provider



Before NAT

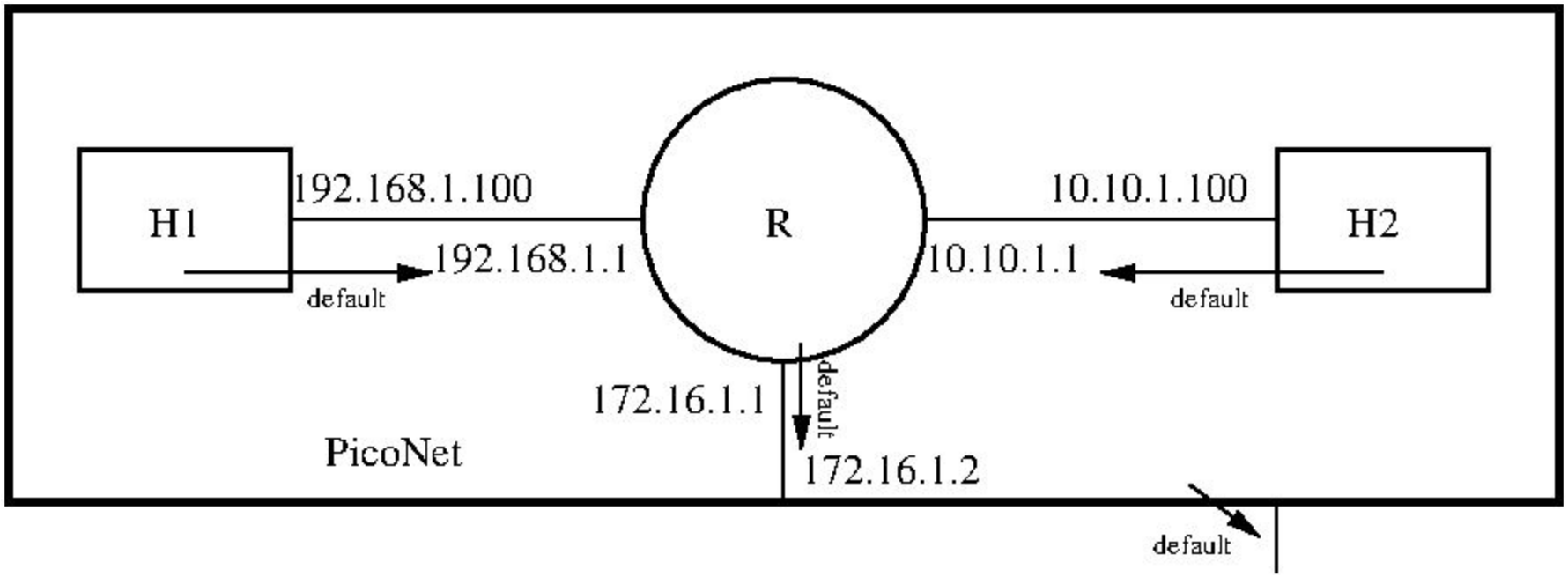

File Edit View Run Kernel Git Tabs Settings Help
Terminal 1
root@h1 piconet> ping -c 1 h2
PING h2 (10.10.1.100) 56(84) bytes of data.
64 bytes from h2 (10.10.1.100): icmp_seq=1 ttl=63 time=100 ms
--- h2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 100.477/100.477/100.477/0.000 ms
root@h1 piconet>
Terminal 3
root@r piconet> tcpdump -l -n -i r-eth0 icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on r-eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
00:22:41.805907 IP 192.168.1.100 > 10.10.1.100: ICMP echo request, id 412, seq 1, length 64
00:22:41.906335 IP 10.10.1.100 > 192.168.1.100: ICMP echo reply, id 412, seq 1, length 64
Terminal 2
root@h2 piconet> tcpdump -l -n -i h2-eth0 icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on h2-eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
00:22:41.906230 IP 192.168.1.100 > 10.10.1.100: ICMP echo request, id 412, seq 1, length 64
00:22:41.906308 IP 10.10.1.100 > 192.168.1.100: ICMP echo reply, id 412, seq 1, length 64
Terminal 4
jovyan@jupyter-pan:~$
5 0
Terminal 4

After NAT

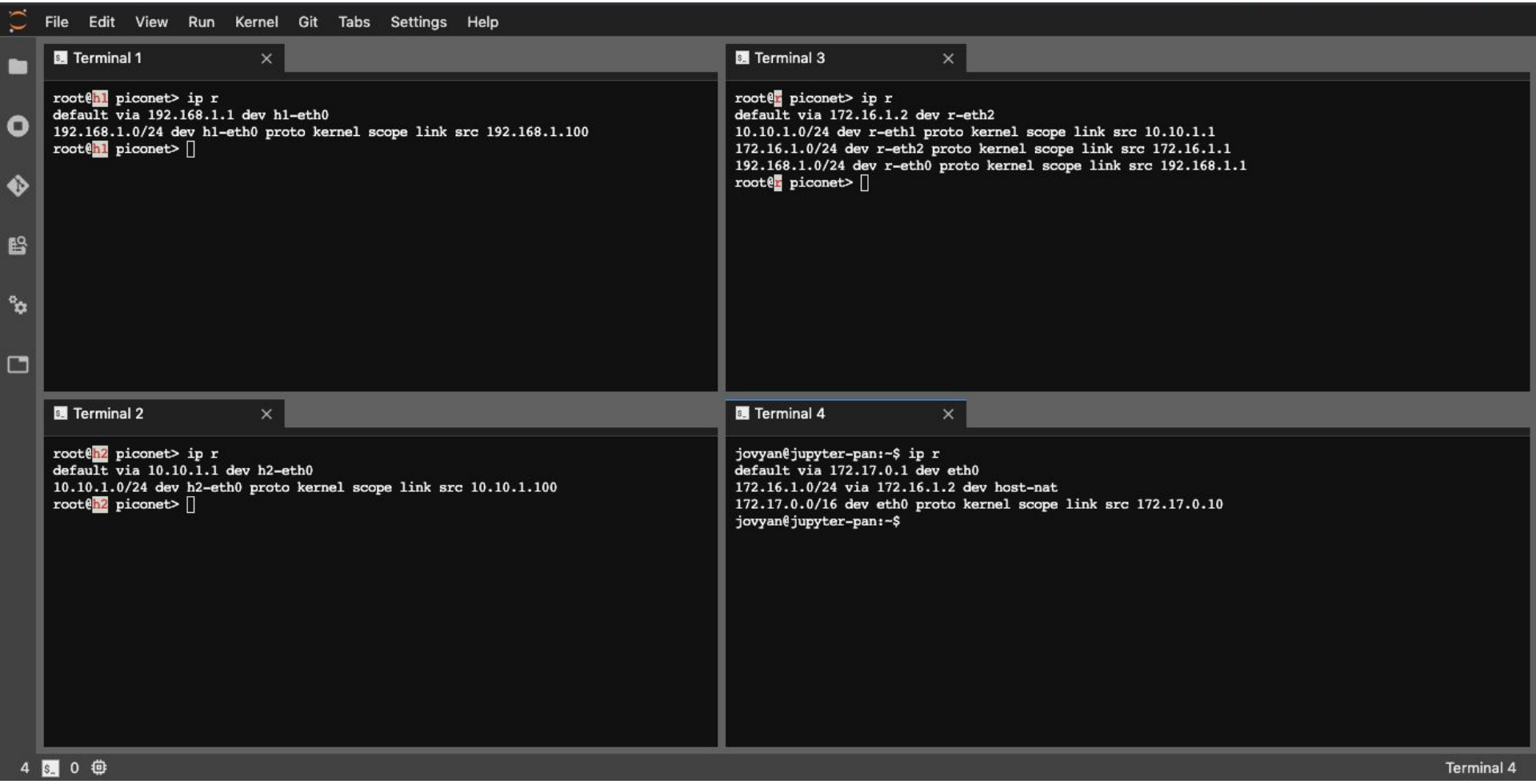

File Edit View Run Kernel Git Tabs Settings Help
Terminal 1
root@h1 piconet> ping -c 1 h2
PING h2 (10.10.1.100) 56(84) bytes of data.
64 bytes from h2 (10.10.1.100): icmp_seq=1 ttl=63 time=100 ms
--- h2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 100.477/100.477/100.477/0.000 ms
root@h1 piconet> ping -c 1 h2
PING h2 (10.10.1.100) 56(84) bytes of data.
64 bytes from h2 (10.10.1.100): icmp_seq=1 ttl=63 time=100 ms
--- h2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 100.329/100.329/100.329/0.000 ms
root@h1 piconet>
Terminal 3
root@r piconet> tcpdump -l -n -i r-eth0 icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on r-eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
00:22:41.805907 IP 192.168.1.100 > 10.10.1.100: ICMP echo request, id 412, seq 1, length 64
00:22:41.906335 IP 10.10.1.100 > 192.168.1.100: ICMP echo reply, id 412, seq 1, length 64
00:24:17.919051 IP 192.168.1.100 > 10.10.1.100: ICMP echo request, id 415, seq 1, length 64
00:24:18.019335 IP 10.10.1.100 > 192.168.1.100: ICMP echo reply, id 415, seq 1, length 64
Terminal 2
root@h2 piconet> tcpdump -l -n -i h2-eth0 icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on h2-eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
00:22:41.906230 IP 192.168.1.100 > 10.10.1.100: ICMP echo request, id 412, seq 1, length 64
00:22:41.906308 IP 10.10.1.100 > 192.168.1.100: ICMP echo reply, id 412, seq 1, length 64
00:24:18.019219 IP 10.10.1.1 > 10.10.1.100: ICMP echo request, id 415, seq 1, length 64
00:24:18.019296 IP 10.10.1.100 > 10.10.1.1: ICMP echo reply, id 415, seq 1, length 64
Terminal 4
jovyan@jupyter-pan:~$ sudo ip netns exec r iptables -t nat -A POSTROUTING -o r-eth1 -j MASQUERADE
jovyan@jupyter-pan:~$
5 0
Terminal 4

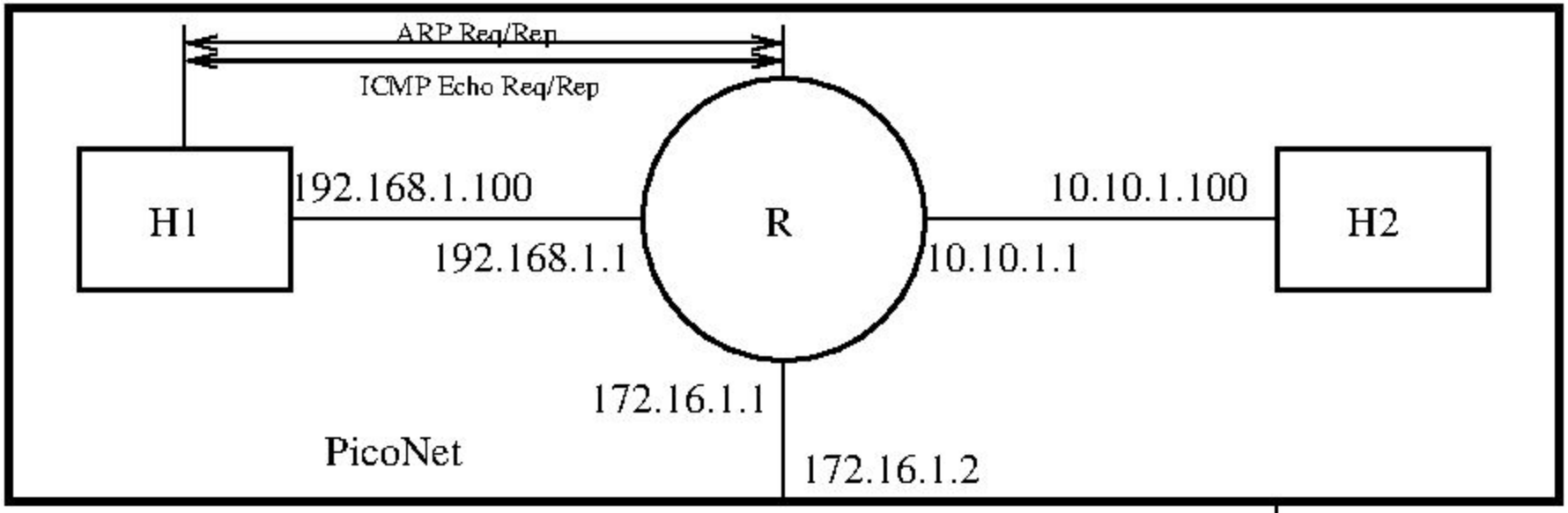
Routing Lab

- H1: 192.168.1.100/24 local and default (all others) to 192.168.1.1
- H2: 10.10.1.100/24 local and default to 10.10.1.1
- R: all 192.168.1.1/24, 10.10.1.1/24 and 172.16.1.1/24 local, default to 172.16.1.2 (and then through the PicoNet host to the Internet)

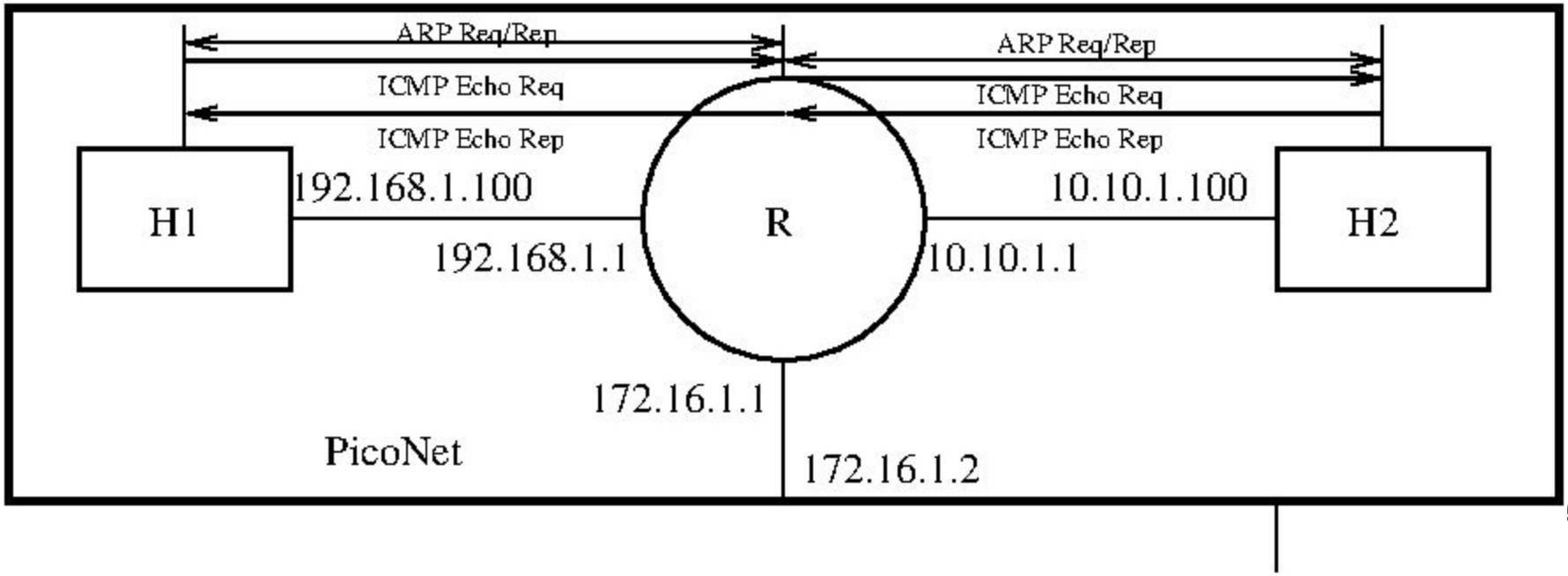


Default routing

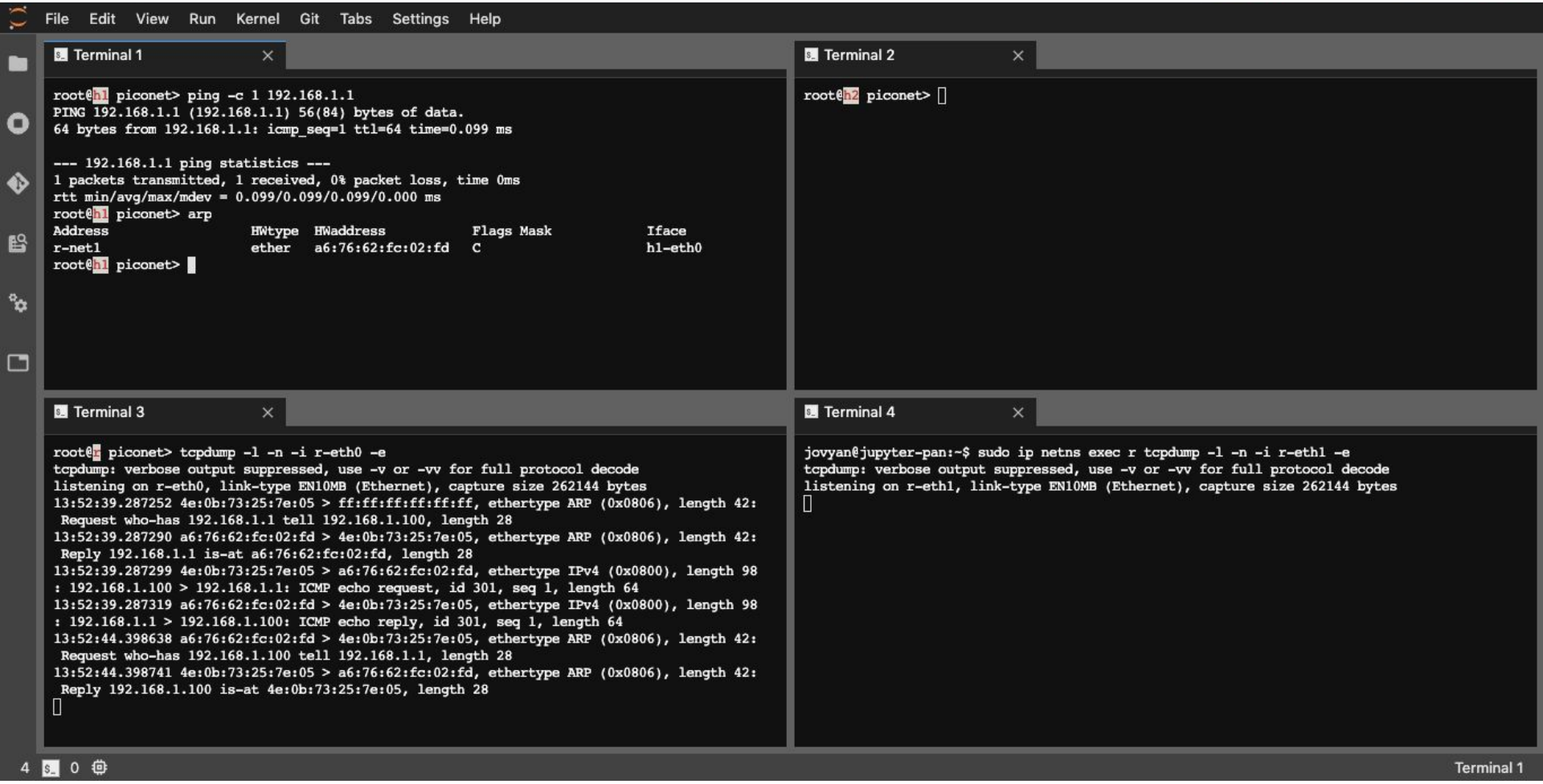




ARP/Interworking Lab



Within the same subnet (routing not involved)



Between different subnets (routing involved too)