



* readandspell.com

PicoHub: the PicoNet in a Hosted JupyterLab Cloud

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



* dreamstime.com

Why? How?

Old Physical Lab



Intermediate Virtual Lab

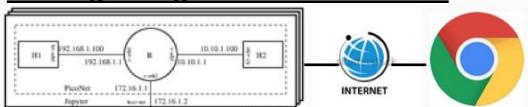


Works well in a Physical Lab, *not* on students laptop



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

New Lightweight Virtual Hosted Lab



Hosted in the Cloud

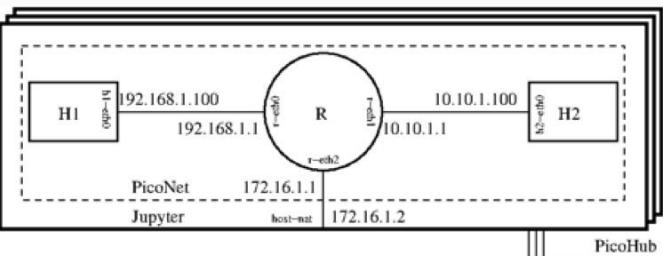


Just a web browser on students laptop



Learning and Teaching
Support and Innovation

PicoNet: the Minimized MiniNet



<https://picohub.csc.uvic.ca>

- Lightweight on students laptop
- Lightweight on lab provisioning

3. Fun labs!

```

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 bytes
20:12:12.349144 ARP. Request who-has 192.168.1.1 tell 192.168.1.100,
20:12:12.349212 ARP. Reply 192.168.1.1 is-at 10.10.1.100: ICMP echo request, id 20
20:12:12.349312 IP 10.10.1.100 > 192.168.1.100: ICMP echo reply, id 20
20:12:13.371383 IP 10.10.1.100 > 10.10.1.100: ICMP echo request, id 21
20:12:13.371459 IP 10.10.1.100 > 192.168.1.100: ICMP echo reply, id 21
20:12:14.395130 IP 10.10.1.100 > 10.10.1.100: ICMP echo request, id 22
20:12:14.395178 IP 10.10.1.100 > 192.168.1.100: ICMP echo reply, id 22
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 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 10.10.1.100
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 10.10.1.100
10. 5.245983 1a:64:67:4b:72:6f -> ee:38:08:02:c6:4b ARP 42 10.10.1.100
  
```

tshark 2.0.3 | ping.cap Analysis Misc

Filter: <Apply> <Reset>

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	1a:64:67:4b Broadcast	ARP	42	Who has 10.10.1.100 Tell 10.10.1.100	Tell 10.10.1.100 is-at 1a:64:67:4b
2	0.000000	ee:38:08:02:c6:4b	ARP	42	10.10.1.100 is-at ee:38:08:02:c6:4b	
3	0.000000	192.168.1.100	10.10.1.100	ICMP	98	Echo (ping) request id=0x01c
4	0.000109	10.10.1.100	192.168.1.100	ICMP	98	Echo (ping) reply id=0x01c
5	1.022195	192.168.1.100	10.10.1.100	ICMP	98	Echo (ping) request id=0x01c
6	1.022219	10.10.1.100	192.168.1.100	ICMP	98	Echo (ping) reply id=0x01c
7	2.045924	192.168.1.100	10.10.1.100	ICMP	98	Echo (ping) request id=0x01c
8	2.045941	10.10.1.100	192.168.1.100	ICMP	98	Echo (ping) reply id=0x01c
9	5.245865	ee:38:08:02:c6:4b	1a:64:67:4b:72:6f	ARP	42	Who has 10.10.1.100 Tell 10.10.1.100 is-at ee:38:08:02:c6:4b
10	5.245983	1a:64:67:4b:72:6f	ee:38:08:02:c6:4b	ARP	42	10.10.1.100 is-at 1a:64:67:4b:72:6f

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



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)



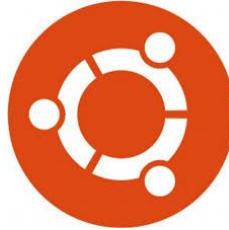
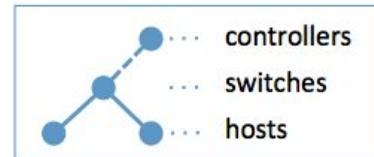
- Minimized OpenWRT kernel (firmware image)
 - 10 lab modules on **HTTP, DNS, TCP, UDP, IP, ICMP, routing, Ethernet, forwarding** and **ARP**
 - **netem** (network emulator) to introduce **packet delay, loss, duplication** and **reordering**
- Running on Linksys WRT54GL (customized firmware)
 - Students 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 (reflashed 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

> sudo mn

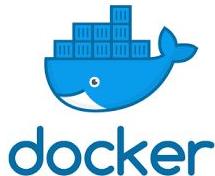


* <http://blog.asiantuntijakaveri.fi/2012/08/send-to-self-on-linux.html>



PicoHub: New CSc361 Lab platform (2020--)

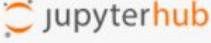
- New requirements
 - Lightweight on student's laptop
 - Only a web browser (e.g., Chrome) needed
 - Lightweight on lab provisioning
 - Virtual lab hosted in the cloud by the dept
 - Tested with minimized Ubuntu VM first
 - Settled down with the lightest Docker container
 - A powerful server can host 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>

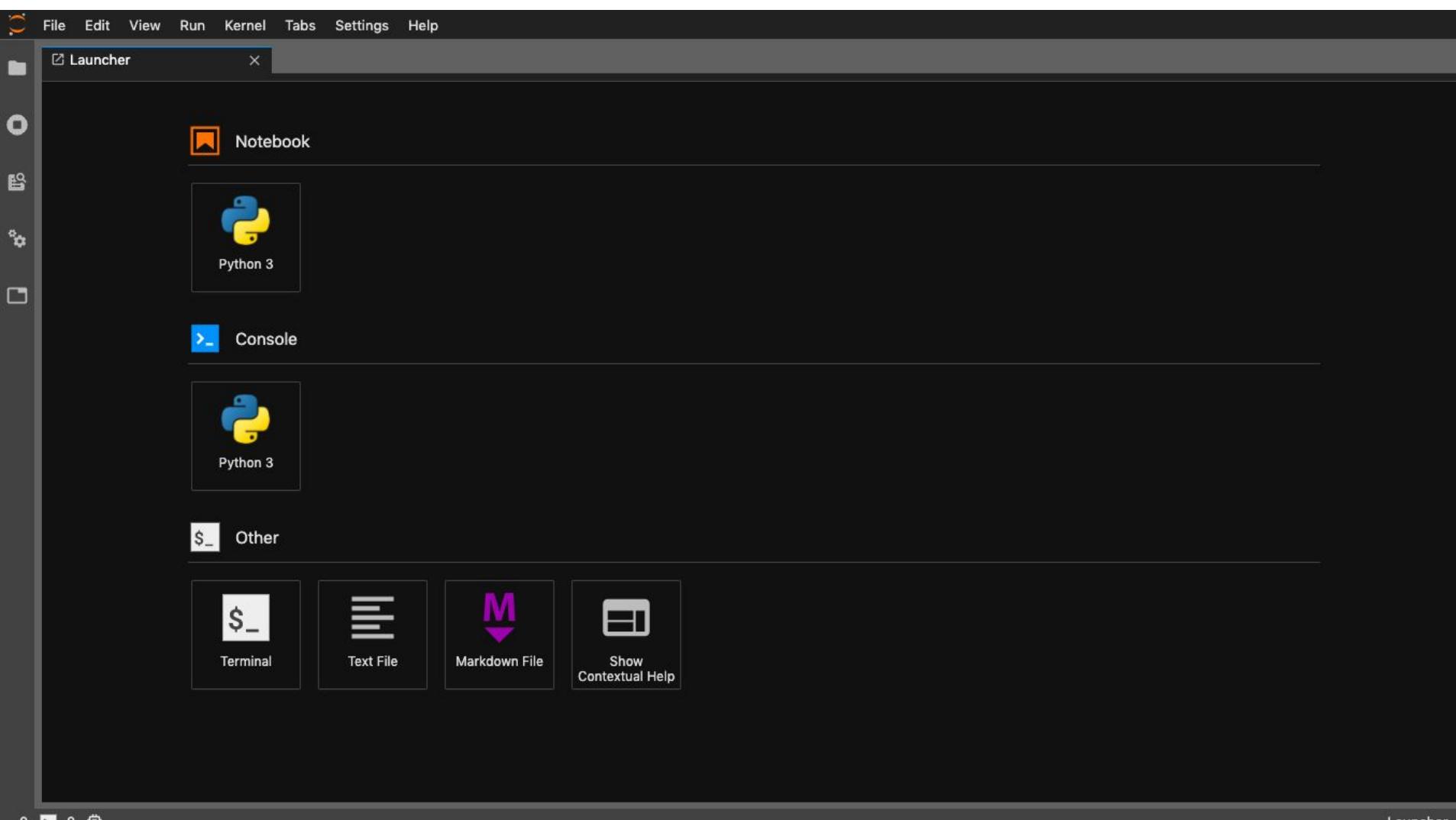
- Please connect to UVic through **UVic VPN** first

- And login PicoHub with your UVic Netlink ID and password

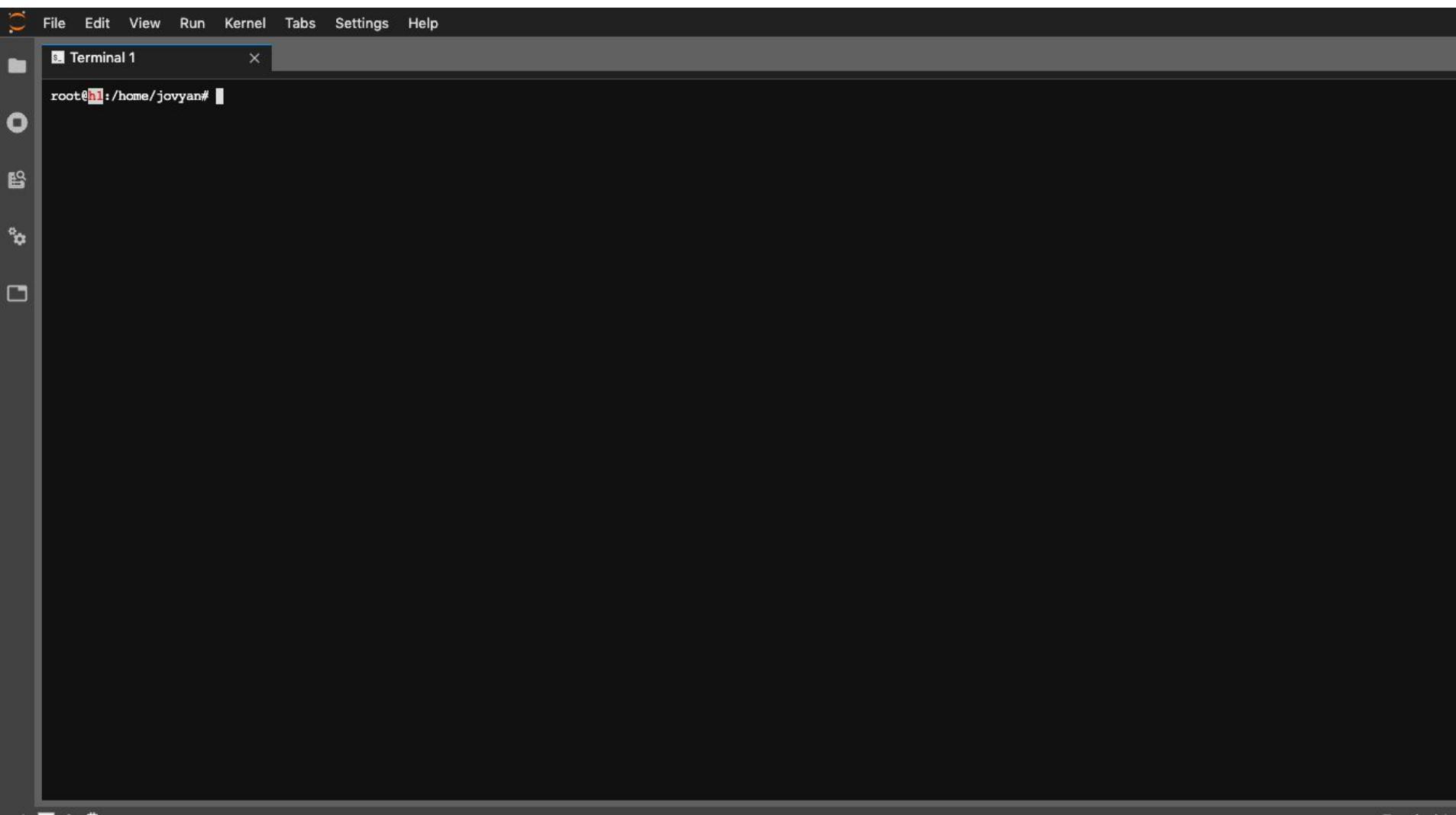


A screenshot of a "Sign in" form. The form has an orange header bar with the word "Sign in". Below the header are two input fields: one for "Username" and one for "Password", both represented by white input boxes with black outlines. At the bottom of the form is an orange rectangular button labeled "Sign In".

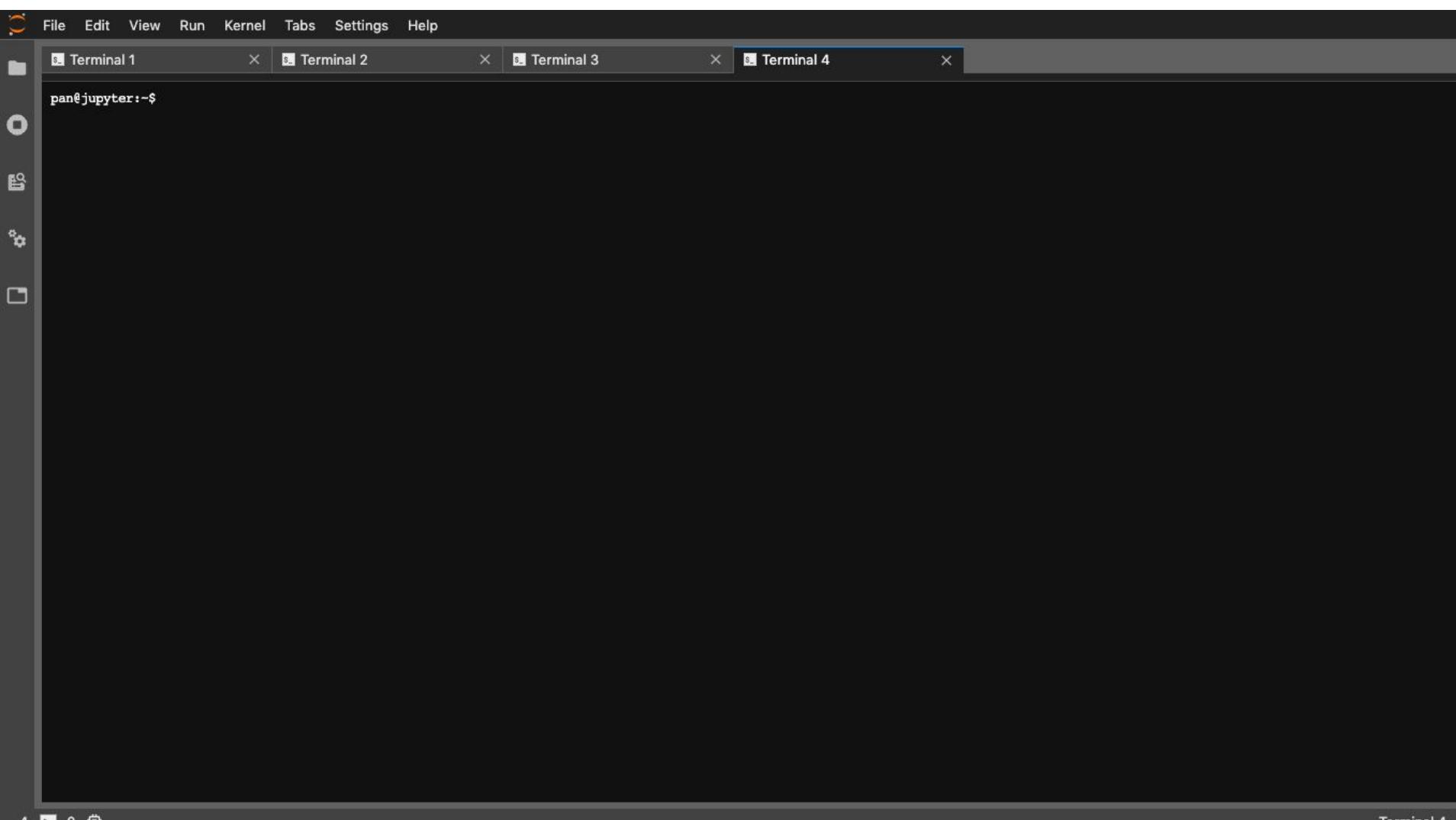
You will see a landing page like this



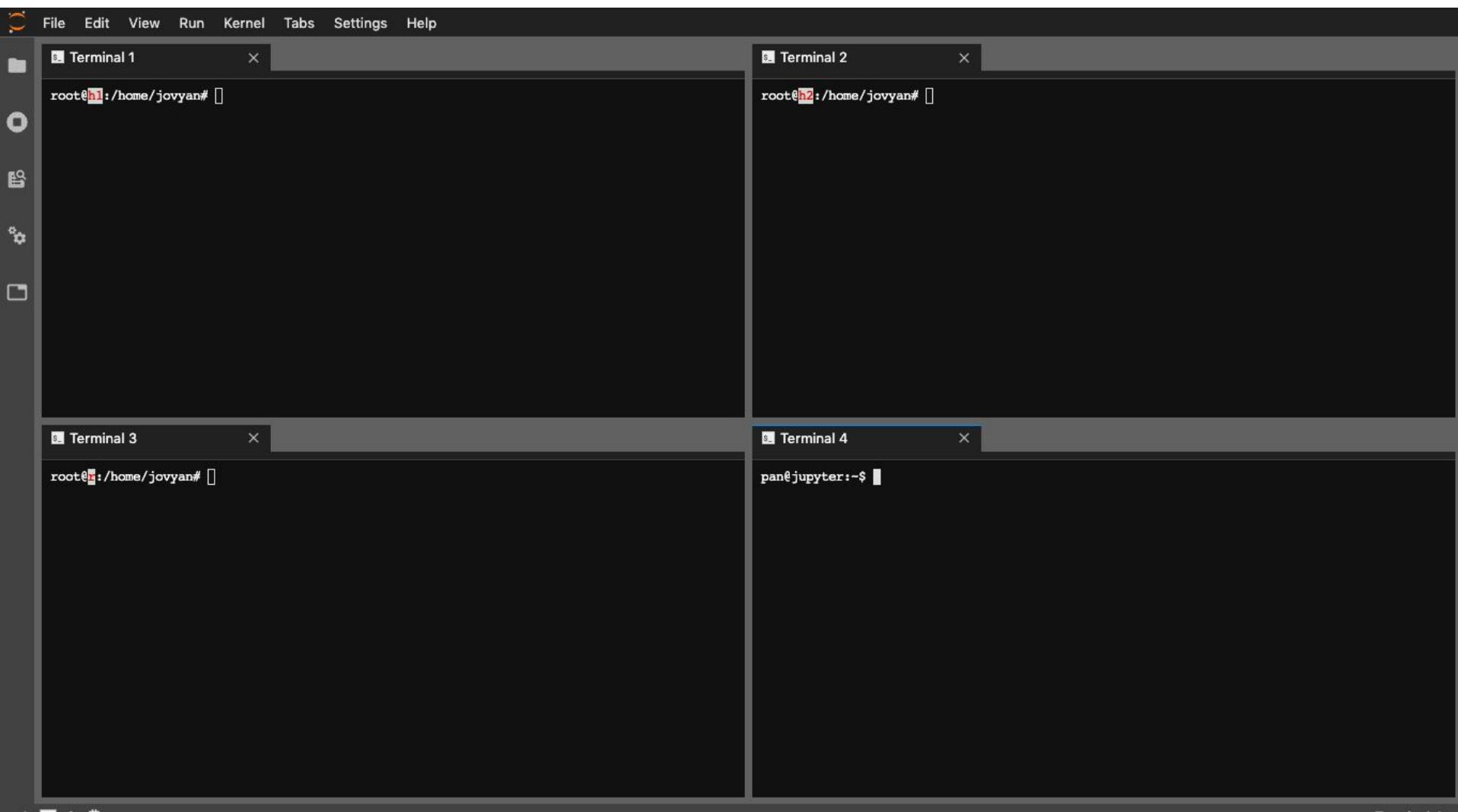
“Launcher -> Terminal” to get the client host (H1)



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

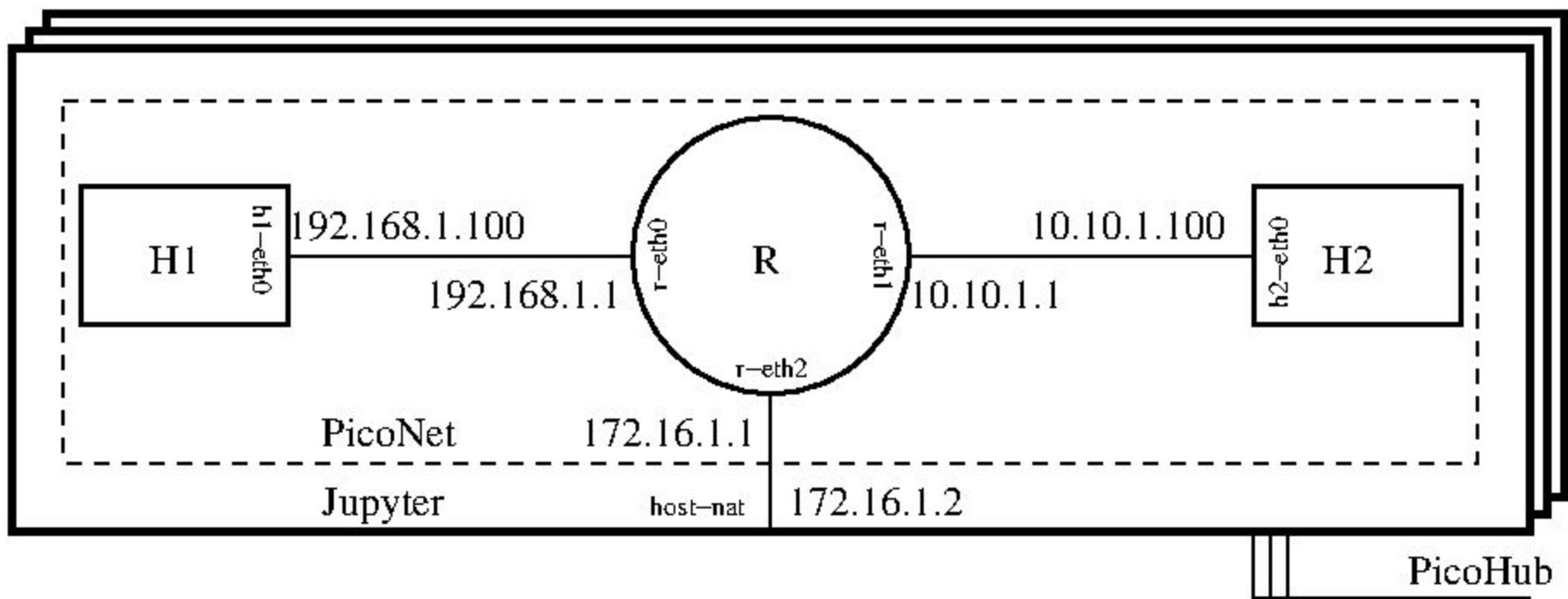


Grab Terminal tab and move to edge/concern to tile



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)

The screenshot shows a terminal window with four panes, each displaying the output of the 'ip a' command on different hosts.

- Terminal 1 (Host H1):** Shows the network configuration for host h1. It includes interfaces lo, h1-eth0, and h1-eth0@if4. The h1-eth0 interface has an IP address 192.168.1.100/24 and a MAC address fe80::fe00:10ff:fe00:100. The h1-eth0@if4 interface is a link-layer duplicate of h1-eth0.
- Terminal 2 (Host H2):** Shows the network configuration for host h2. It includes interfaces lo, h2-eth0, and h2-eth0@if6. The h2-eth0 interface has an IP address 10.10.1.100/24 and a MAC address fe80::ec38:8fff:fe02:c64b/64. The h2-eth0@if6 interface is a link-layer duplicate of h2-eth0.
- Terminal 3 (Host R):** Shows the network configuration for host r. It includes interfaces r-eth0, r-eth1, r-eth2, and r-eth2@if7. The r-eth0 interface has an IP address 192.168.1.1/24 and a MAC address fe80::f0b0:63ff:fe6b:6d9c/64. The r-eth1 and r-eth2 interfaces have IP addresses 10.10.1.1/24 and 172.16.1.1/24 respectively, and MAC addresses fe80::f0b0:63ff:fe6b:6d9c/64 and fe80::1864:67ff:fedb:726f/64. The r-eth2@if7 interface is a link-layer duplicate of r-eth2.
- Terminal 4 (Jupyter Notebook):** Shows the network configuration for a Jupyter notebook instance on host r. It includes interfaces lo, eth0, and host-nat@if9. The lo interface is a loopback interface. The eth0 interface has an IP address 10.42.10.106/32 and a MAC address b6:cc:00:1a:36:85. The host-nat@if9 interface is a link-layer duplicate of eth0.

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

```
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 2
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 la:64:67:4b:72:6f, length 28
root@h2:/home/jovyan# tsshark -r ping.cap
Running as user "root" and group "root". This could be dangerous.
1 0.000000 la: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 → la: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 → la:64:67:4b:72:6f ARP 42 Who has 10.10.1.1? Tell 10.10.1.100
10 5.245903 la:64:67:4b:72:6f → ee:38:08:02:c6:4b ARP 42 10.10.1.1 is at la:64:67:4b:72:6f
root@h2:/home/jovyan# 

Terminal 3
root@h3:/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
root@h3:/home/jovyan# 
```

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 2

```
termshark 2.0.3 | ping.cap
```

Analysis Misc

Filter: <Apply> <Recent>

No.	Time	Source	Destination	Protocol	Length	Info
1	0.00000	la:64:67:4b: Broadcast	ARP	42	Who has 10.10.1.100? Tell 10.10	
2	0.00003	ee:38:08:02: la: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: la:64:67:4b:	ARP	42	Who has 10.10.1.1? Tell 10.10.1	
10	5.24590	la:64:67:4b: ee:38:08:02:	ARP	42	10.10.1.1 is at la:64:67:4b:72:	

[+] Frame 1: 42 bytes on wire (336 bits), 42 bytes captured (336 bits)
[+] Ethernet II, Src: la:64:67:4b:72:6f (la:64:67:4b:72:6f), Dst: Broadcast (ff: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: la:64:67:4b:72:6f (la: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 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 01d gKro...
0020 00 00 00 00 00 00 0a 0a 01 64d

Terminal 3

```
root@h2:/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
root@h2:/home/jovyan#
```

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

The screenshot shows a Linux desktop environment with several windows open:

- Terminal 1:** Shows a root shell session where a ping command is run against host h2 (IP 10.10.1.100). The output shows three ICMP echo requests being sent and their responses.
- Terminal 2:** Shows the Wireshark interface (version 2.0.3) with a capture file named "ping.cap". The packet list pane shows 10 captured frames, all of which are ARP requests. The details and bytes panes show the structure of an ARP frame.
- File Manager:** Shows a file named "ping.cap" selected in a folder named "tracero". A context menu is open for this file, listing options like Open, Rename, Delete, Copy, Duplicate, Download, Shut Down Kernel, Copy Shareable Link, Copy Path, Copy Download Link, New Folder, New File, New Markdown File, Paste, and Shift+Right Click for Browser Menu.

Analyze in your Wireshark: ARP request

Wireshark interface showing network traffic analysis.

Selected packet details:

No.	Time	Source	Destination	Protocol	Length	Info
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 (reply in 4)
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 (reply in 6)
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 (reply in 8)
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

Selected packet bytes:

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

Selected packet details pane:

- 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: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

Packets: 10 · Displayed: 10 (100.0%) · Profile: Default

Ping request (and response in frame: 4)

Screenshot of Wireshark showing network traffic analysis.

Display Filter: Apply a display filter ... <%>

No.	Time	Source	Destination	Protocol	Length	Info
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 (reply in 4)
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 (reply in 6)
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 (reply in 8)
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

Frame 4: 98 bytes on wire (784 bits), 98 bytes captured (784 bits)

Ethernet II, Src: ee:38:08:02:c6:4b (ee:38:08:02:c6:4b), Dst: 1a:64:67:4b:72:6f (1a:64:67:4b:72:6f)

Internet Protocol Version 4, Src: 10.10.1.100, Dst: 192.168.1.100

Internet Control Message Protocol

- Type: 0 (Echo (ping) reply)
- Code: 0
- Checksum: 0x5a7d [correct]
- [Checksum Status: Good]
- Identifier (BE): 459 (0x01cb)
- Identifier (LE): 51969 (0xcb01)
- Sequence number (BE): 1 (0x0001)
- Sequence number (LE): 256 (0x0100)
- [Request frame: 3]
- [Response time: 0.068 ms]
- Timestamp from icmp data: Sep 14, 2020 20:12:12.000000000 PDT
- [Timestamp from icmp data (relative): 0.349342000 seconds]

Data (48 bytes)

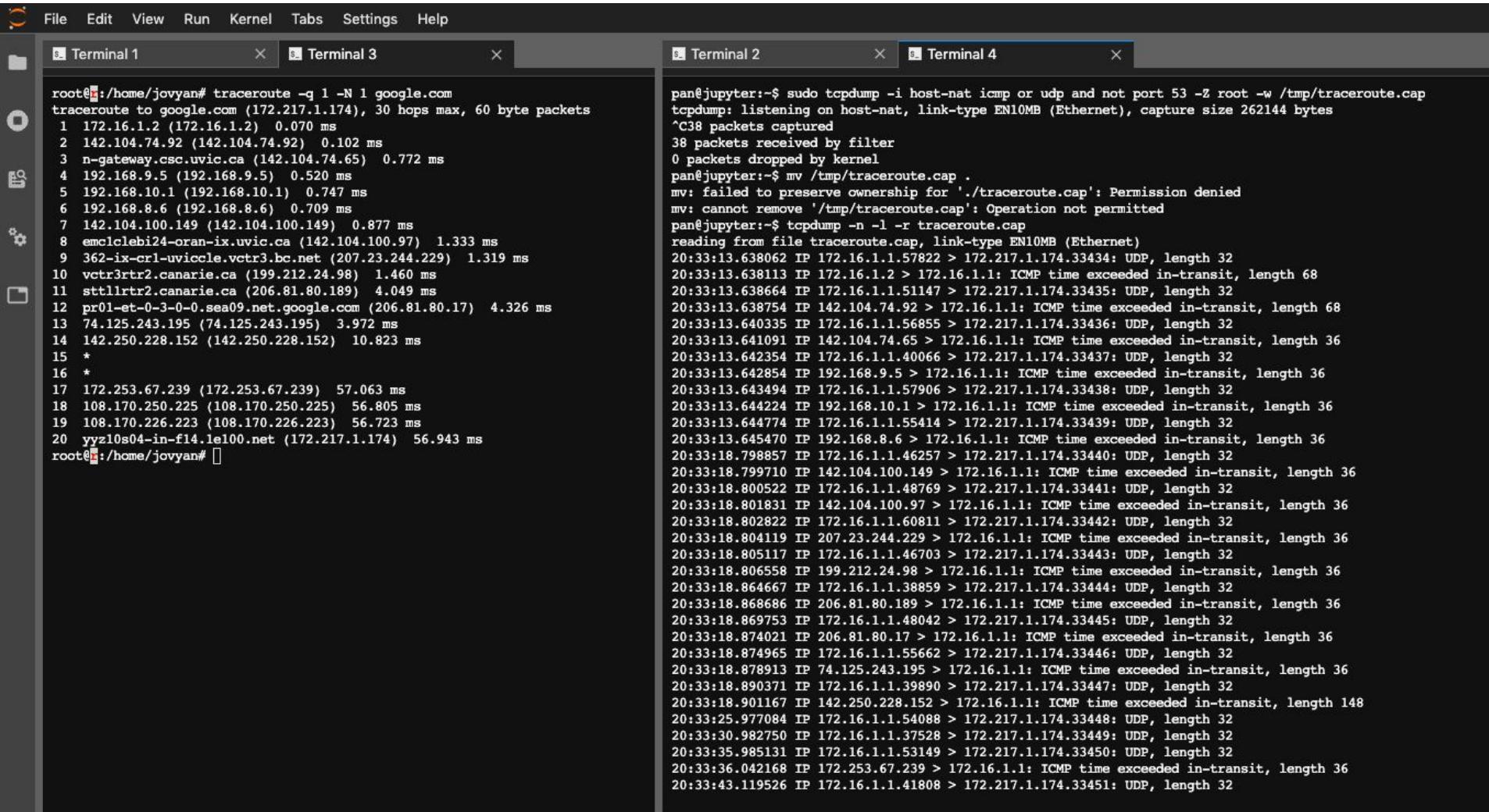
0000	1a 64 67 4b 72 6f ee 38 08 02 c6 4b 08 00 45 00	dgKro·8 ···K·E·
0010	00 54 ec e8 00 00 40 01 c0 46 0a 0a 01 64 c0 a8	·T···@ ·F··d··
0020	01 64 00 00 5a 7d 01 cb 00 01 0c 31 60 5f 00 00	·d··Z}·····1`_··
0030	00 00 73 53 05 00 00 00 00 00 10 11 12 13 14 15	··sS·····.....
0040	16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 22 23 24 25	·..... ···!#%·
0050	26 27 28 29 2a 2b 2c 2d 2e 2f 30 31 32 33 34 35	&'()**+, - ./012345
0060	36 37	67

Internet Control Message Protocol (icmp), 64 bytes

Packets: 10 · Displayed: 10 (100.0%)

Profile: Default

Another practice if time allows: traceroute to Google



The screenshot shows a terminal window with four tabs. The tabs are labeled Terminal 1, Terminal 3, Terminal 2, and Terminal 4. Terminal 1 contains the output of a traceroute command to Google. Terminal 2 contains the output of a tcpdump session capturing ICMP and UDP traffic on host-nat. Terminal 3 and Terminal 4 are currently empty.

```
root@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  emcliclebi24-oran-ix.uvic.ca (142.104.100.97)  1.333 ms
 9  362-ix-crl-uvcclle.vctr3.bc.net (207.23.244.229)  1.319 ms
10  vctr3rtr2.canarie.ca (199.212.24.98)  1.460 ms
11  stillrtr2.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.le100.net (172.217.1.174)  56.943 ms
root@jovyan:~#
```

```
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 -r -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
20:33:13.798857 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
```

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
```

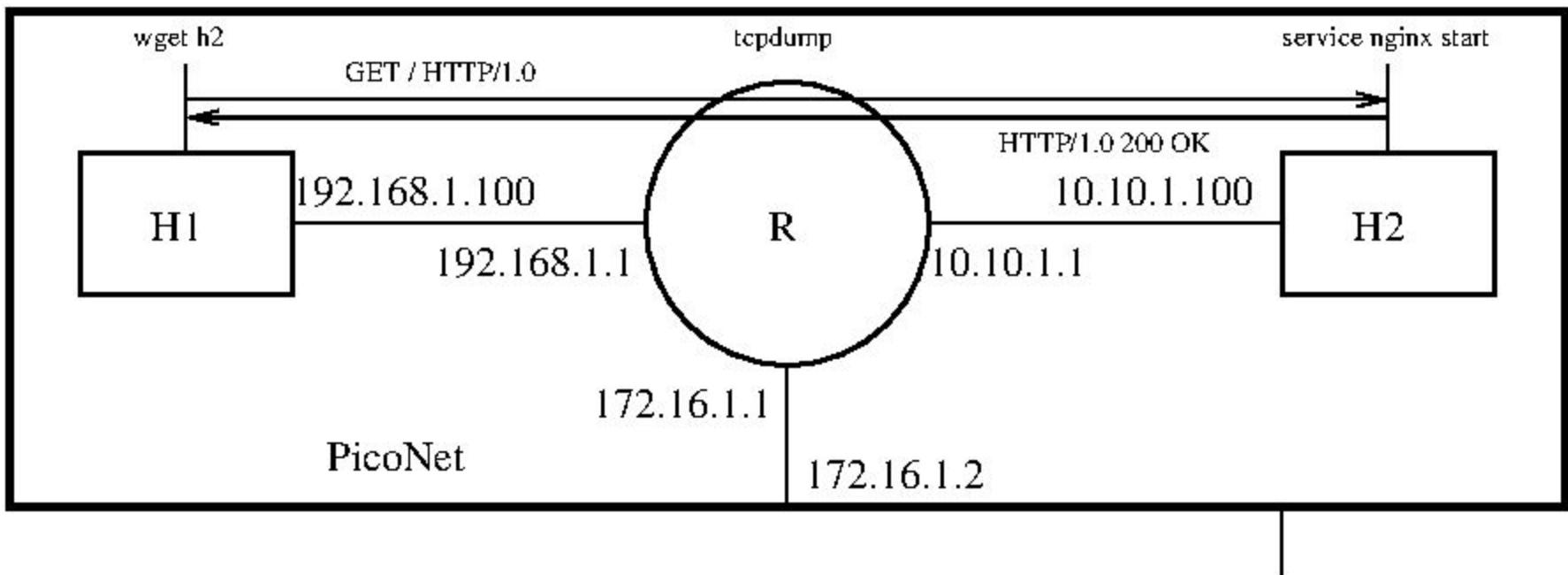
A lot of FUN in PicoNet!

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”

The screenshot shows a terminal window with four tabs:

- Terminal 1:** Shows the command `nc h2 80` being run, which returns an HTTP response with "Connection: keep-alive". The response body contains the HTML content of a page served by Nginx.
- Terminal 2:** Shows the command `service nginx start` being run, followed by `netstat -tunlp`. The output shows active Internet connections, including two entries for Nginx (tcp and tcp6) listening on port 80.
- Terminal 3:** Shows the command `tcpdump -l -n -i r-eth0 tcp port 80` being run. The output displays a series of TCP packets exchanged between the local host (IP 192.168.1.100) and a remote host (IP 10.10.1.100). The connection is persistent, with multiple requests and responses visible.
- Terminal 4:** Shows the command `nc h2 80` being run, which returns an HTTP response with "Connection: keep-alive". The response body contains the HTML content of a page served by Nginx.

Non-persistent HTTP connection

The screenshot shows a terminal window with three tabs open:

- Terminal 1:** Displays the contents of a file, likely an HTML document, showing the start of an HTML body, an XML declaration, and various HTTP headers (Server, Date, Content-Type, Content-Length, Last-Modified, Connection, ETag, Accept-Ranges) followed by the actual HTML content.
- Terminal 3:** Shows the raw TCP traffic for an HTTP request. The sequence starts with a SYN packet (seq 395728913, ack 369831457) and continues through several ACK and SYN-ACK exchanges between the client (IP 10.10.1.100) and the server (IP 192.168.1.100). The server's responses include HTTP headers and body content.
- Terminal 4:** Shows the raw TCP traffic for an HTTP response. It includes multiple ACK packets from the client (IP 192.168.1.100) to the server (IP 10.10.1.100), indicating the non-persistent nature of the connection.

TCPdump: remote raw analysis in PicoNet

```
pan@jupyter:~$ tcpdump -l -n -r http-non-persistent.cap
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], len
gth 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], len
gth 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.tx
t HTTP/1.1
22:54:10.492015 IP 172.16.1.2.80 > 172.16.1.1.57664: Flags [.], ack 179, win 1016, options [nop,nop,TS val 2699675908 ecr 4026006664], length 0
22:54:10.492120 IP 172.16.1.2.80 > 172.16.1.1.57664: Flags [FP.], seq 1:550, ack 179, win 1016, options [nop,nop,TS val 2699675908 ecr 4026006664], length 549: HTTP: HTTP/1.1 200 OK
22:54:10.492176 IP 172.16.1.1.57664 > 172.16.1.2.80: Flags [.], ack 551, win 1002, options [nop,nop,TS val 4026006664 ecr 2699675908], length 0
22:54:10.492495 IP 172.16.1.1.57664 > 172.16.1.2.80: Flags [F.], seq 179, ack 551, win 1002, options [nop,nop,TS val 4026006664 ecr 2699675908], length 0
22:54:10.492512 IP 172.16.1.2.80 > 172.16.1.1.57664: Flags [.], ack 180, win 1016, options [nop,nop,TS val 2699675908 ecr 4026006664], length 0
22:54:10.492669 IP 172.16.1.1.57666 > 172.16.1.2.80: Flags [S], seq 305741031, win 64240, options [mss 1460,sackOK,TS val 4026006665 ecr 0,nop,wscale 6], length 0
22:54:10.492689 IP 172.16.1.2.80 > 172.16.1.1.57666: Flags [S.], seq 2247109765, ack 305741032, win 65160, options [mss 1460,sackOK,TS val 2699675909 ecr 4026006665,nop,wscale 6], leng
th 0
22:54:10.492714 IP 172.16.1.1.57666 > 172.16.1.2.80: Flags [.], ack 1, win 1004, options [nop,nop,TS val 4026006665 ecr 2699675909], length 0
22:54:10.492815 IP 172.16.1.1.57666 > 172.16.1.2.80: Flags [P.], seq 1:179, ack 1, win 1004, options [nop,nop,TS val 4026006665 ecr 2699675909], length 178: HTTP: GET /dns-iterative.tx
t HTTP/1.1
22:54:10.492827 IP 172.16.1.2.80 > 172.16.1.1.57666: Flags [.], ack 179, win 1016, options [nop,nop,TS val 2699675909 ecr 4026006665], length 0
22:54:10.492949 IP 172.16.1.2.80 > 172.16.1.1.57666: Flags [FP.], seq 1:1087, ack 179, win 1016, options [nop,nop,TS val 2699675909 ecr 4026006665], length 1086: HTTP: HTTP/1.1 200 OK
22:54:10.493005 IP 172.16.1.1.57666 > 172.16.1.2.80: Flags [.], ack 1088, win 1000, options [nop,nop,TS val 4026006665 ecr 2699675909], length 0
22:54:10.493338 IP 172.16.1.1.57666 > 172.16.1.2.80: Flags [F.], seq 179, ack 1088, win 1002, options [nop,nop,TS val 4026006665 ecr 2699675909], length 0
22:54:10.493361 IP 172.16.1.2.80 > 172.16.1.1.57666: Flags [.], ack 180, win 1016, options [nop,nop,TS val 2699675909 ecr 4026006665], length 0
pan@jupyter:~$
```

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

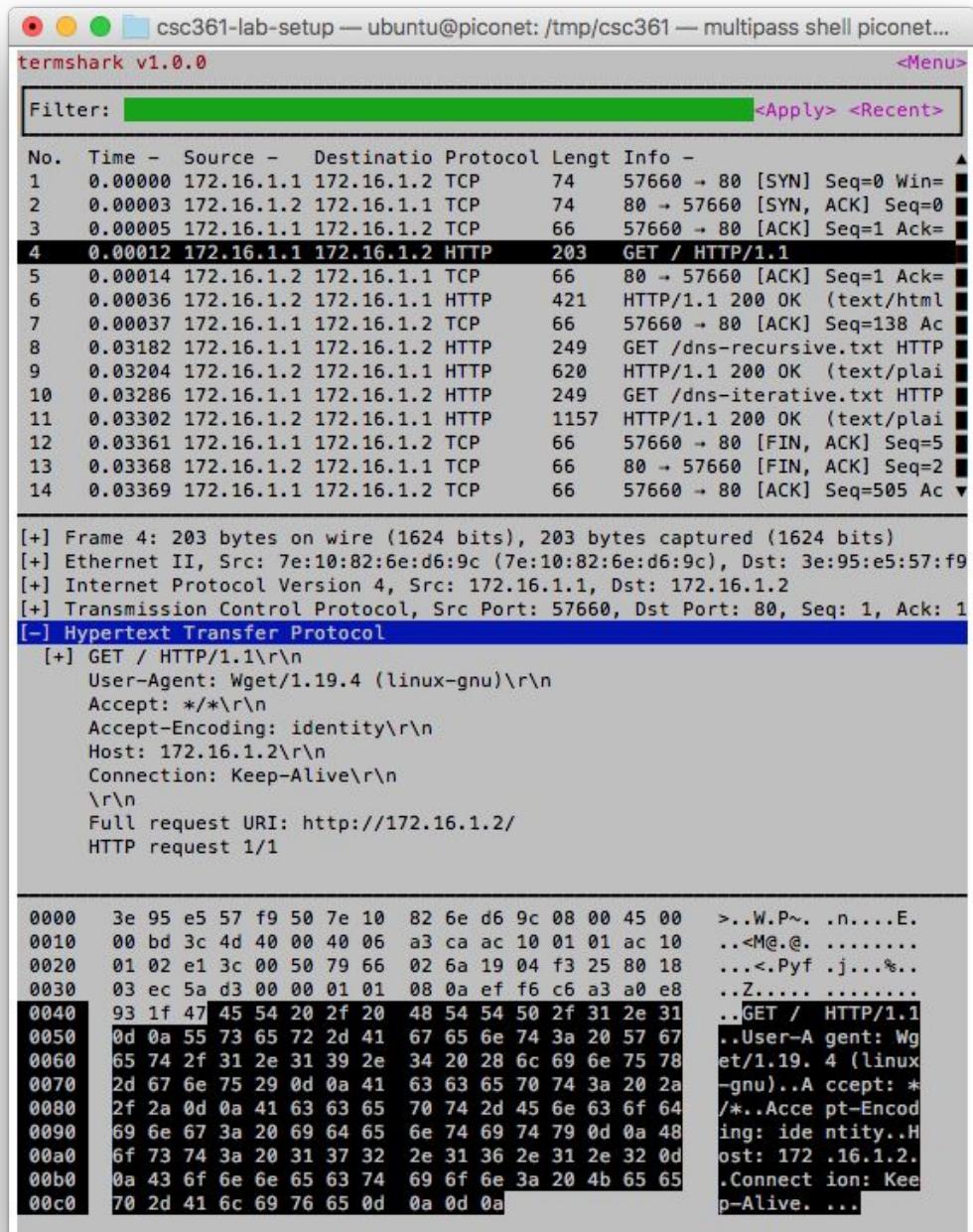
```
root@:~/home/jovyan# tshark -r http-non-persistent.cap
Running as user "root" and group "root". This could be dangerous.
1  0.000000 172.16.1.1 → 172.16.1.2    TCP 74 57662 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=4026006663 TSecr=0 WS=64
2  0.000044 172.16.1.2 → 172.16.1.1    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)
15 0.001473 172.16.1.1 → 172.16.1.2   TCP 66 57664 → 80 [ACK] Seq=179 Ack=551 Win=64128 Len=0 TSval=4026006664 TSecr=2699675908
16 0.001792 172.16.1.1 → 172.16.1.2   TCP 66 57664 → 80 [FIN, ACK] Seq=179 Ack=551 Win=64128 Len=0 TSval=4026006664 TSecr=2699675908
17 0.001809 172.16.1.2 → 172.16.1.1   TCP 66 80 → 57664 [ACK] Seq=551 Ack=180 Win=65024 Len=0 TSval=2699675908 TSecr=4026006664
18 0.001966 172.16.1.1 → 172.16.1.2   TCP 74 57666 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=4026006665 TSecr=0 WS=64
19 0.001986 172.16.1.2 → 172.16.1.1   TCP 74 80 → 57666 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM=1 TSval=2699675909 TSecr=4026006665 WS=64
20 0.002011 172.16.1.1 → 172.16.1.2   TCP 66 57666 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=4026006665 TSecr=2699675909
21 0.002112 172.16.1.1 → 172.16.1.2   HTTP 244 GET /dns-iterative.txt HTTP/1.1
22 0.002124 172.16.1.2 → 172.16.1.1   TCP 66 80 → 57666 [ACK] Seq=1 Ack=179 Win=65024 Len=0 TSval=2699675909 TSecr=4026006665
23 0.002246 172.16.1.2 → 172.16.1.1   HTTP 1152 HTTP/1.1 200 OK (text/plain)
24 0.002302 172.16.1.1 → 172.16.1.2   TCP 66 57666 → 80 [ACK] Seq=179 Ack=1088 Win=64000 Len=0 TSval=4026006665 TSecr=2699675909
25 0.002635 172.16.1.1 → 172.16.1.2   TCP 66 57666 → 80 [FIN, ACK] Seq=179 Ack=1088 Win=64128 Len=0 TSval=4026006665 TSecr=2699675909
26 0.002658 172.16.1.2 → 172.16.1.1   TCP 66 80 → 57666 [ACK] Seq=1088 Ack=180 Win=65024 Len=0 TSval=2699675909 TSecr=4026006665
```

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
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 ...>.Plg .{.....							

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

The screenshot shows the Wireshark interface with a list of captured packets. The first few packets are SYN, ACK, and ACK exchanges between two hosts. Subsequent packets show HTTP GET requests for files like 'dns-recursive.txt' and 'dns-iterative.txt'. The JupyterLab window shows a browser tab at picotest.csc.uvic.ca/user/pan/lab? displaying the same content.

tcp.stream eq 0

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	172.16.1.1	172.16.1.2	TCP	74	57660 - 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TStamp=4025927331 TSval=4025927331 TSerr=0
2	0.000037	172.16.1.2	172.16.1.1	TCP	74	80 - 57660 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM=1 TStamp=26.. TSval=4025927331 TSerr=26..
3	0.000057	172.16.1.1	172.16.1.2	TCP	66	57660 - 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TStamp=4025927331 TSval=4025927331 TSerr=2699596575
4	0.000128	172.16.1.1	172.16.1.2	HTTP	203	GET / HTTP/1.1
5	0.000144	172.16.1.2	172.16.1.1	TCP	66	80 - 57660 [ACK] Seq=1 Ack=138 Win=65024 Len=0 TStamp=2699596575 TSval=4025927331 TSerr=4025927331
6	0.000364	172.16.1.2	172.16.1.1	HTTP	421	HTTP/1.1 200 OK (text/html)
7	0.000377	172.16.1.1	172.16.1.2	TCP	66	57660 - 80 [ACK] Seq=138 Ack=356 Win=64128 Len=0 TStamp=4025927331 TSval=2699596575 TSerr=2699596575
8	0.031822	172.16.1.1	172.16.1.2	HTTP	249	GET /dns-recursive.txt HTTP/1.1
9	0.032041	172.16.1.2	172.16.1.1	HTTP	620	HTTP/1.1 200 OK (text/plain)
10	0.032869	172.16.1.1	172.16.1.2	HTTP	249	GET /dns-iterative.txt HTTP/1.1
11	0.033023	172.16.1.2	172.16.1.1	HTTP	1157	HTTP/1.1 200 OK (text/plain)
12	0.033610	172.16.1.1	172.16.1.2	TCP	66	57660 - 80 [FIN, ACK] Seq=504 Ack=2001 Win=64128 Len=0 TStamp=4025927364 TSval=4025927364 TSerr=2..
13	0.033684	172.16.1.2	172.16.1.1	TCP	66	80 - 57660 [FIN, ACK] Seq=2001 Ack=505 Win=64768 Len=0 TStamp=2699596608 TSval=2699596608 TSerr=4..
14	0.033699	172.16.1.1	172.16.1.2	TCP	66	57660 - 80 [ACK] Seq=505 Ack=2002 Win=64128 Len=0 TStamp=4025927364 TSval=2699596608 TSerr=4..

Frame 4: 203 bytes on wire (1624 bits), 203 bytes captured (1624 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: 57660, Dst Port: 80, Seq: 1, Ack: 1, Len: 137

Hypertext Transfer Protocol
GET / HTTP/1.1\r\nUser-Agent: Wget/1.19.4 (linux-gnu)\r\nAccept: */*\r\nAccept-Encoding: identity\r\nHost: 172.16.1.2\r\nConnection: Keep-Alive\r\n\r\n[Full request URI: http://172.16.1.2/] [HTTP request 1/3] [Response in frame: 6] [Next request in frame: 8]

Name

Name	Last Modified
dns-iterative.cap	seconds ago
dns-recursive.cap	seconds ago
http-non-persistent.cap	seconds ago
http-persistent.cap	seconds ago
nginx-html.zip	go
tcp-connection-	Open
tcp-cubic-after.	Open With
tcp-cubic-before.	+ Open in New Browser Tab
tcp-error-after.c	Rename
tcp-error-before.c	Delete
tcp-reno-mincw	Cut
tcp-reno-mincw	Copy
tcp-reno-mincw	Duplicate
tcp-reno-mincw	Download

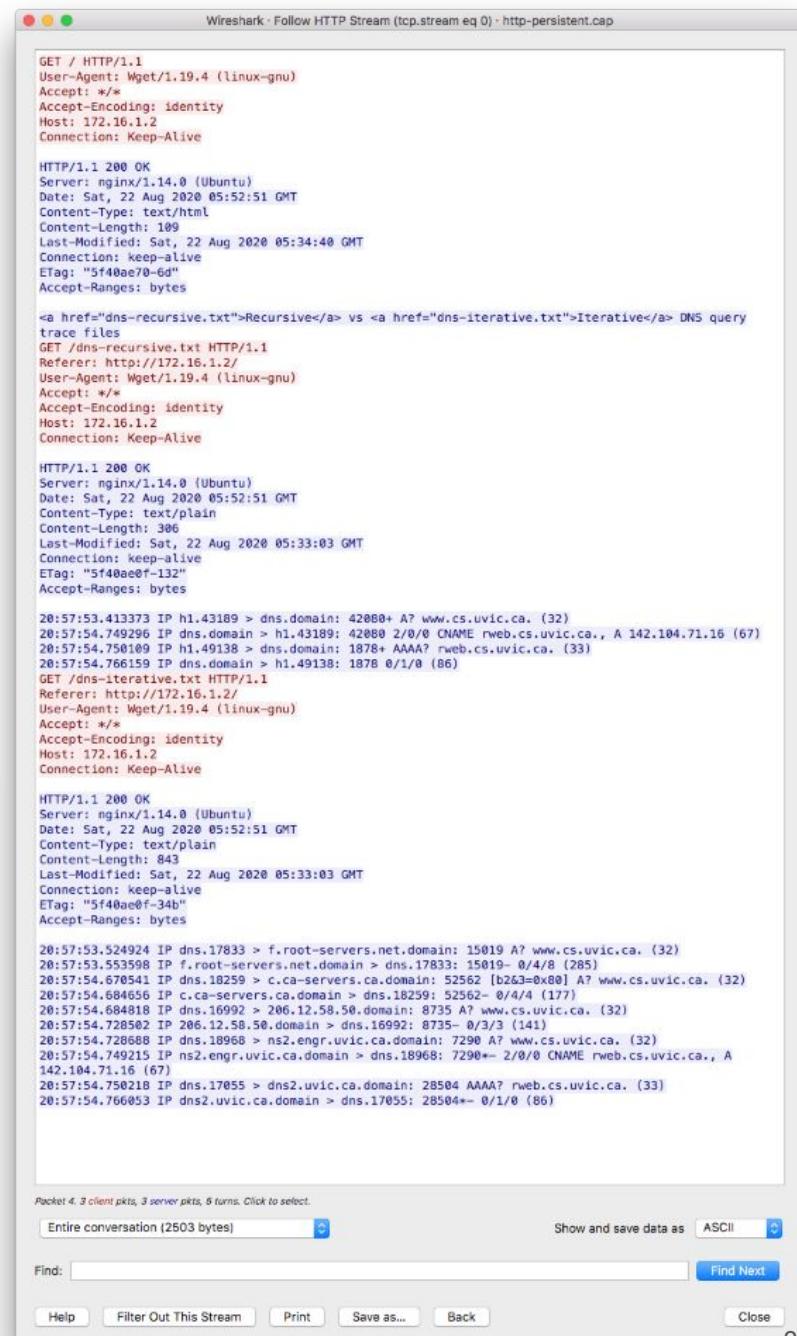
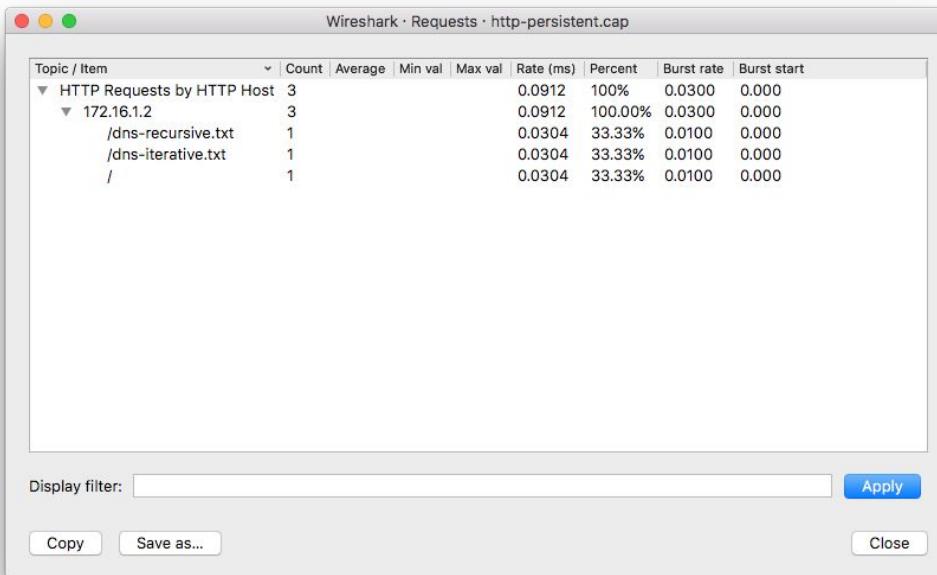
0000 3e 95 e5 57 f9 58 7e 10 82 6e d5 9c 08 00 45 00 > W P~ n E
0010 00 bd 3c 4d 49 00 48 06 a3 c0 ac 10 01 01 ac 10 > MO @
0020 01 02 e1 3c 00 50 79 66 82 6a 19 04 f3 25 80 18 > Pyf j %
0030 03 1f 5a d3 00 00 01 01 08 00 e7 16 c0 a3 a0 e0 Z
0040 93 1f 47 45 54 20 2f 20 48 54 54 50 21 31 2e 31 GET / HTTP/1.1
0050 0d 0f 55 73 65 12 41 67 65 66 74 38 20 57 67 User-Agent: Wg
0060 65 74 2f 31 31 39 2e 30 28 28 6c 69 66 75 78 et/1.19.4 (linux
0070 2d 66 66 5 29 0d 0a 41 63 63 65 78 24 3a 28 0a gnu) A ccept: *0080 2f 28 0d 0a 41 63 63 65 70 74 2d 45 66 63 6f 64 /* Acce p-Encod
0090 69 66 67 3a 20 69 84 65 8e 74 69 74 79 0d 0a 48 ing: Ide ntity H
00a0 6f 73 74 3a 20 31 37 32 2e 31 36 2e 31 2e 32 0d ost: 172.16.1.2
00b0 0a 43 6f 6e 66 65 63 74 69 6f 6e 3a 20 4b 65 65 Connect ion: Kee
00c0 70 2d 41 6c 69 76 65 0d 0a 0d 0a p-Alive: ...

Packets: 14 · Displayed: 14 (100.0%) Profile: Default

38

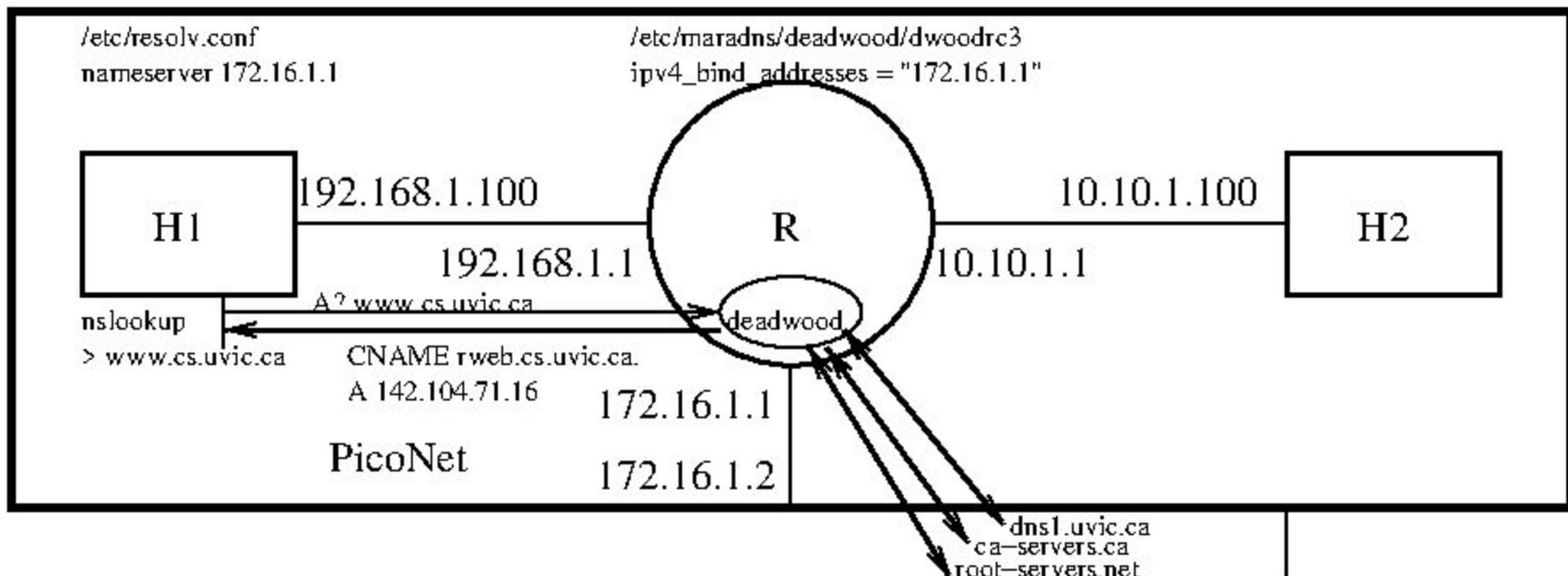
More advanced features

- 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

The screenshot shows a terminal window with four panes. The top-left pane (Terminal 1) displays the output of a nslookup command for 'www.csc.uvic.ca'. It shows the server address (172.16.1.1), the query type (A), and the response from a recursive server (142.104.71.42). The top-right panes (Terminal 3 and Terminal 4) show the output of a tcpdump command capturing DNS traffic on interface r-eth2. The traffic includes multiple queries from various clients (e.g., 172.16.1.1, 199.7.83.42, 206.12.58.52) and responses from a single server (142.104.71.42). The bottom-left pane (Terminal 2) shows the output of an ip netns exec command on interface r-eth0, which captures traffic from a different network namespace.

```
root@piconet:~# nslookup www.csc.uvic.ca
> server 172.16.1.1
Default server: 172.16.1.1
Address: 172.16.1.1#53
> www.csc.uvic.ca
Server:      172.16.1.1
Address:      172.16.1.1#53

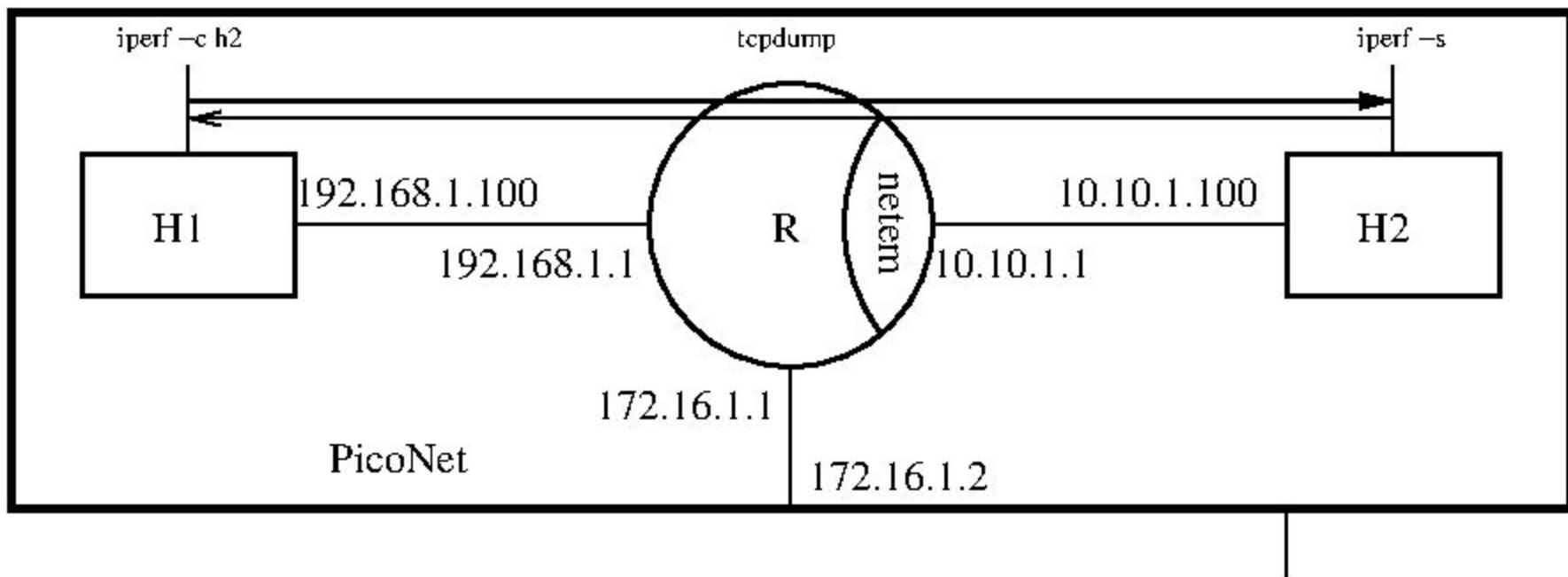
Non-authoritative answer:
Name:   www.csc.uvic.ca
Address: 142.104.71.42
>

root@piconet:~# tcpdump -l -n -i r-eth2 udp port 53
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on r-eth2, link-type EN10MB (Ethernet), capture size 262144 bytes
14:50:32.201042 IP 172.16.1.1.17136 > 128.63.2.53.53: 65506 A? www.csc.uvic.ca. (33)
14:50:33.256981 IP 172.16.1.1.18847 > 199.7.83.42.53: 48225 [b2&3=0x80] A? www.csc.uvic.ca. (33)
14:50:33.260755 IP 199.7.83.42.53 > 172.16.1.1.18847: 48225- 0/4/8 (286)
14:50:34.313777 IP 172.16.1.1.15882 > 199.253.250.68.53: 4154 [b2&3=0x80] A? www.csc.uvic.ca. (33)
)
14:50:34.474257 IP 199.253.250.68.53 > 172.16.1.1.15882: 4154- 0/4/4 (171)
14:50:34.475732 IP 172.16.1.1.18375 > 206.12.58.52.53: 49121 A? www.csc.uvic.ca. (33)
14:50:34.483990 IP 206.12.58.52.53 > 172.16.1.1.18375: 49121 0/3/3 (142)
14:50:34.484433 IP 172.16.1.1.17461 > 142.104.6.1.53: 17329 A? www.csc.uvic.ca. (33)
14:50:34.485544 IP 142.104.6.1.53 > 172.16.1.1.17461: 17329* 1/5/5 A 142.104.71.42 (226)
14:50:34.487015 IP 172.16.1.1.18068 > 142.104.75.2.53: 12756 AAAA? www.csc.uvic.ca. (33)
14:50:34.490865 IP 142.104.75.2.53 > 172.16.1.1.18068: 12756 0/0/0 (33)

root@piconet:~# ip netns exec r tcpdump -l -n -i r-eth0 udp port 53
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on r-eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
14:50:32.200698 IP 192.168.1.100.56825 > 172.16.1.1.53: 3649+ A? www.csc.uvic.ca. (33)
14:50:34.485794 IP 172.16.1.1.53 > 192.168.1.100.56825: 3649 1/0/0 A 142.104.71.42 (49)
14:50:34.486798 IP 192.168.1.100.59732 > 172.16.1.1.53: 61181+ AAAA? www.csc.uvic.ca. (33)
14:50:34.491138 IP 172.16.1.1.53 > 192.168.1.100.59732: 61181 0/1/0 (73)
```

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)

The screenshot shows a terminal window with four tabs open:

- Terminal 1:** Shows the output of an iperf client command on a piconet. It connects to host 2 (TCP port 5001) with a window size of 85.0 KByte (default). The bandwidth is measured as 5.66 KBytes 324 Mbits/sec.
- Terminal 2:** Shows the output of an iperf server command on a piconet. It listens on TCP port 5001 with a window size of 2.83 KByte (WARNING: requested 1.41 KByte). The bandwidth is measured as 5.66 KBytes 115 Kbits/sec.
- Terminal 3:** Shows the output of tc qdisc commands to add and show a netem discipline on interface r-eth1 with a delay of 100ms. It also shows the output of tcpdump capturing traffic on interface r-eth0 for port 5001.
- Terminal 4:** Shows the output of tc qdisc commands to add and show a qdisc netem discipline on interface r-eth1 with a delay of 100.0ms. It also shows the output of tcpdump capturing traffic on interface r-eth0 for port 5001.

TCP connection release (4-way handshake)

The screenshot shows a terminal window with four tabs open:

- Terminal 1:** Shows the command `iperf -c h2 -l 724 -n 5792` being run on a host named `h1`. It connects to host `h2` on port 5001 and reports a bandwidth of 5.66 KBytes (324 Mbits/sec).
- Terminal 2:** Shows the command `iperf -s -w 1448 -l 724` being run on a host named `h2`. It warns that the TCP window size is set to 1448 bytes, which may lead to poor performance. It then listens on port 5001.
- Terminal 3:** Shows the initial connection setup between `h1` and `h2`. It includes several TCP segments with various flags (e.g., SYN, ACK, FIN) and sequence numbers.
- Terminal 4:** Shows the final release of the connection. It displays the termination of the connection with segments containing the FIN flag and sequence numbers corresponding to the ones in Terminal 3.

TCP flow control

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

tcp-connection-flow.cap

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.1.100	10.10.1.100	TCP	74	36286 - 5001 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=3366068716 ..
2	0.107762	10.10.1.100	192.168.1.100	TCP	74	5001 - 36286 [SYN, ACK] Seq=0 Ack=1 Win=1448 Len=0 MSS=1460 SACK_PERM=1 TSval=1..
3	0.107797	192.168.1.100	10.10.1.100	TCP	66	36286 - 5001 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=3366068824 TSecr=10..
4	0.107954	192.168.1.100	10.10.1.100	TCP	790	36286 - 5001 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len=724 TSval=3366068824 TSecr=10..
5	0.107967	192.168.1.100	10.10.1.100	TCP	790	[TCP Window Full] 36286 - 5001 [PSH, ACK] Seq=725 Ack=1 Win=64256 Len=724 TSva..
6	0.213507	10.10.1.100	192.168.1.100	TCP	66	5001 - 36286 [ACK] Seq=1 Ack=725 Win=724 Len=0 TSval=102209553 TSecr=3366068824 ..
7	0.213728	10.10.1.100	192.168.1.100	TCP	66	5001 - 36286 [ACK] Seq=1 Ack=1449 Win=1448 Len=0 TSval=102209554 TSecr=3366068824 ..
8	0.213735	192.168.1.100	10.10.1.100	TCP	790	36286 - 5001 [PSH, ACK] Seq=1449 Ack=1 Win=64256 Len=724 TSval=3366068930 TSecr=..
9	0.213736	192.168.1.100	10.10.1.100	TCP	790	[TCP Window Full] 36286 - 5001 [PSH, ACK] Seq=2173 Ack=1 Win=64256 Len=724 TSva..
10	0.213848	10.10.1.100	192.168.1.100	TCP	66	[TCP ZeroWindow] 5001 - 36286 [ACK] Seq=1 Ack=2897 Win=0 Len=0 TSval=102209559 ..
11	0.318529	10.10.1.100	192.168.1.100	TCP	66	[TCP Window Update] 5001 - 36286 [ACK] Seq=1 Ack=2897 Win=1448 Len=0 TSval=102209559 ..
12	0.318532	192.168.1.100	10.10.1.100	TCP	790	36286 - 5001 [PSH, ACK] Seq=2897 Ack=1 Win=64256 Len=724 TSval=3366069035 TSecr=..
13	0.318533	192.168.1.100	10.10.1.100	TCP	790	[TCP Window Full] 36286 - 5001 [PSH, ACK] Seq=3621 Ack=1 Win=64256 Len=724 TSva..
14	0.427871	10.10.1.100	192.168.1.100	TCP	66	[TCP ZeroWindow] 5001 - 36286 [ACK] Seq=1 Ack=4345 Win=0 Len=0 TSval=102209768 ..
15	0.427888	10.10.1.100	192.168.1.100	TCP	66	[TCP Window Update] 5001 - 36286 [ACK] Seq=1 Ack=4345 Win=1448 Len=0 TSval=102209768 ..
16	0.427891	192.168.1.100	10.10.1.100	TCP	790	36286 - 5001 [PSH, ACK] Seq=4345 Ack=1 Win=64256 Len=724 TSval=3366069144 TSecr=..
17	0.530058	10.10.1.100	192.168.1.100	TCP	66	5001 - 36286 [ACK] Seq=1 Ack=5069 Win=1448 Len=0 TSval=102209870 TSecr=3366069144 ..
18	0.530071	192.168.1.100	10.10.1.100	TCP	790	36286 - 5001 [FIN, PSH, ACK] Seq=5069 Ack=1 Win=64256 Len=724 TSval=3366069246 ..
19	0.641399	10.10.1.100	192.168.1.100	TCP	66	5001 - 36286 [ACK] Seq=1 Ack=5794 Win=1448 Len=0 TSval=102209982 TSecr=3366069246 ..
20	0.653045	10.10.1.100	192.168.1.100	TCP	66	5001 - 36286 [FIN, ACK] Seq=1 Ack=5794 Win=1448 Len=0 TSval=102209993 TSecr=3366069369 ..
21	0.653061	192.168.1.100	10.10.1.100	TCP	66	36286 - 5001 [ACK] Seq=5794 Ack=2 Win=0 Len=0 TSval=3366069369 TSecr=102209..

```

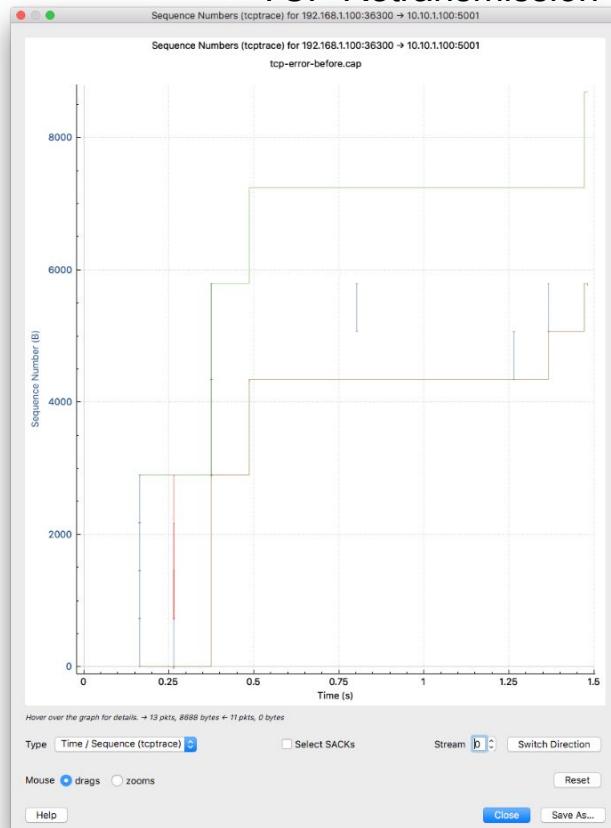
> Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits)
> Ethernet II, Src: d6:81:68:d4:44:bc (d6:81:68:d4:44:bc), Dst: 42:19:1e:01:2c:f8 (42:19:1e:01:2c:f8)
> Internet Protocol Version 4, Src: 192.168.1.100, Dst: 10.10.1.100
▼ Transmission Control Protocol, Src Port: 36286, Dst Port: 5001, Seq: 0, Len: 0
    Source Port: 36286
    Destination Port: 5001
    [Stream index: 0]
    [TCP Segment Len: 0]
    Sequence number: 0 (relative sequence number)
    Sequence number (raw): 160082804
    [Next sequence number: 1 (relative sequence number)]
    Acknowledgment number: 0
    Acknowledgment number (raw): 0
    1010 .... = Header Length: 40 bytes (10)
    ▼ Flags: 0x0002 (SYN)
        000.... .... = Reserved: Not set
        ...0.... .... = Nonce: Not set
        ....0.... .... = Congestion Window Reduced (CWR): Not set
        ....0.... .... = ECN-Echo: Not set
        ....0.... .... = Urgent: Not set
        ....0.... .... = Acknowledgment: Not set
        ....0.... .... = Push: Not set
        ....0.... .... = Reset: Not set
        ....0.... ..1. = Sync: Set
            ▶ [Expert Info [Chat/Sequence]: Connection establish request (SYN): server port 5001]
            ....0....0 = Fin: Not set
            [TCP Flags: .....S.1]
            Window size value: 64240
            [Calculated window size: 64240]
            Checksum: 0xcdca8 [unverified]
            [Checksum Status: Unverified]
            Urgent pointer: 0
            Options: (20 bytes), Maximum segment size, SACK permitted, Timestamps, No-Operation (NOP), Window scale
                ▶ TCP Option - Maximum segment size: 1460 bytes
                ▶ TCP Option - SACK permitted
                ▶ TCP Option - Timestamps: TSval 3366068716, TSecr 0
                    Kind: Time Stamp Option (8)
                    Length: 10
                    Timestamp value: 3366068716
                    Timestamp echo reply: 0
                ▶ TCP Option - No-Operation (NOP)
                ▶ TCP Option - Window scale: 6 (multiply by 64)
            ▶ [Timestamps]
        0000 42 19 1e 01 2c f8 d6 81 68 d4 44 bc 08 00 45 00 B..., ..h-D--E-
        0010 00 3c 7c 15 40 00 06 f1 2c c8 a8 01 64 0a 0a <| @@ , ,d-
        0020 01 64 8d be 13 89 09 8a b4 00 00 00 a8 02 d.....t...
        0030 fa f0 cd a8 00 00 02 04 05 b4 04 02 08 0a c8 a2 !.....
        0040 21 ec 00 00 00 01 03 03 06 !

```

Packets: 21 - Displayed: 21 (100.0%) Profile: Default

TCP error control

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



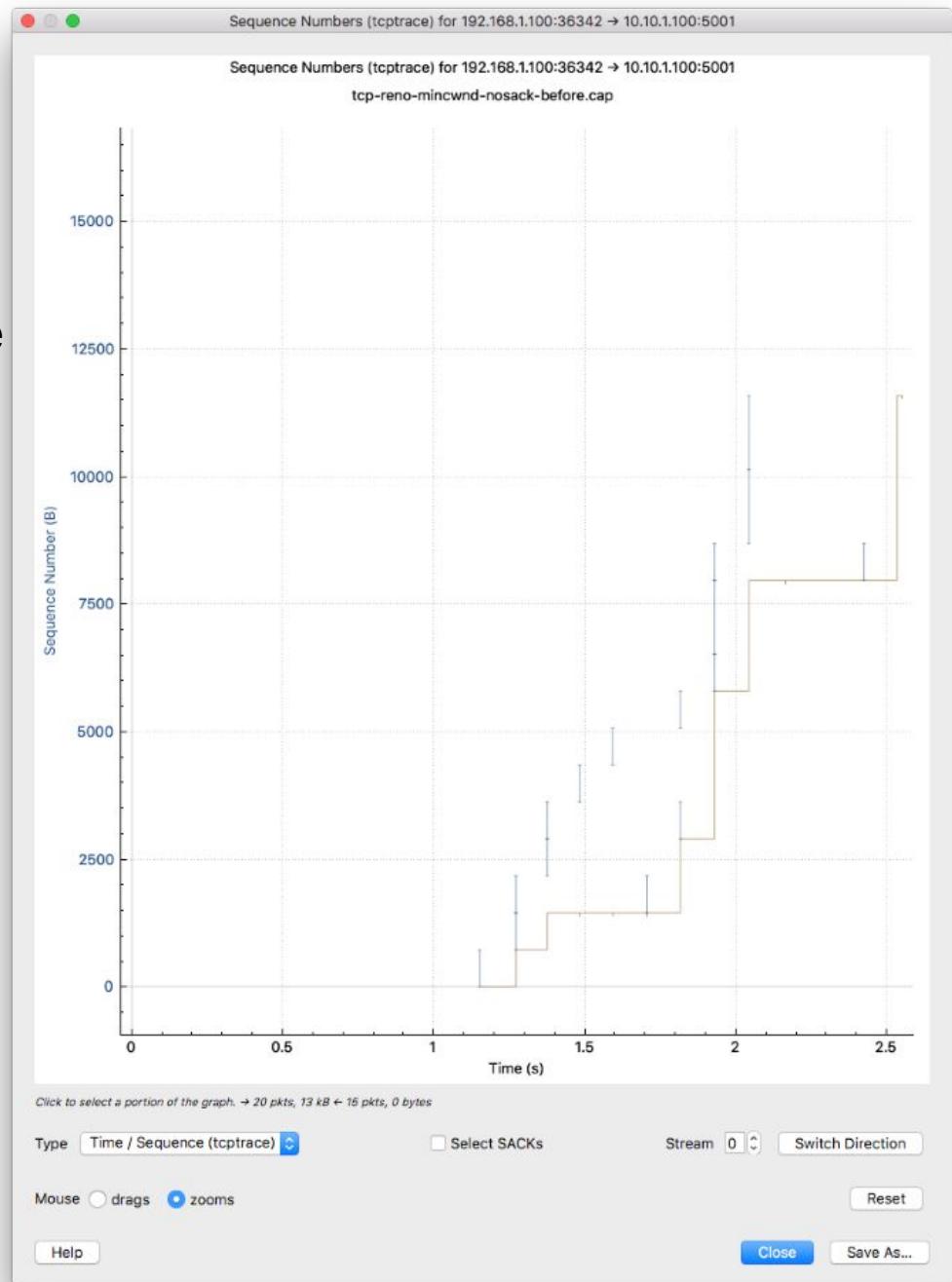
tcp-error-before.cap						
Apply a display filter ... <None>						
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.1.100	10.10.1.100	TCP	74	36300 - 5001 [SYN] Seq=0 Win=65504 Len=0 MSS=730 SACK_PERM=1 TSval=3368478335 TSecr=104619224...
2	0.163968	10.10.1.100	192.168.1.100	TCP	74	5001 - 36300 [SYN, ACK] Seq=0 Ack=1 Win=2896 Len=0 MSS=1468 SACK_PERM=1 TSval=1...
3	0.163986	192.168.1.100	10.10.1.100	TCP	66	36300 - 5001 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=3368478456 TSecr=104619123...
4	0.164144	192.168.1.100	10.10.1.100	TCP	798	36300 - 5001 [PSH, ACK] Seq=1 Ack=1 Win=65563 Len=724 TSval=3368478499 TSecr=10...
5	0.164160	192.168.1.100	10.10.1.100	TCP	798	36300 - 5001 [PSH, ACK] Seq=725 Ack=1 Win=65538 Len=724 TSval=3368478499 TSecr=10...
6	0.164167	192.168.1.100	10.10.1.100	TCP	798	36300 - 5001 [PSH, ACK] Seq=1449 Ack=1 Win=65538 Len=724 TSval=3368478499 TSecr=10...
7	0.164172	192.168.1.100	10.10.1.100	TCP	790	[TCP Window Full] 36300 - 5001 [PSH, ACK] Seq=2173 Ack=1 Win=65536 Len=724 TSva...
8	0.264893	10.10.1.100	192.168.1.100	TCP	78	[TCP Dup ACK 2#1] 5001 - 36300 [ACK] Seq=1 Ack=1 Win=2896 Len=0 TSval=104619224...
9	0.264896	10.10.1.100	192.168.1.100	TCP	78	[TCP Dup ACK 2#2] 5001 - 36300 [ACK] Seq=1 Ack=1 Win=2896 Len=0 TSval=104619224...
10	0.264897	10.10.1.100	192.168.1.100	TCP	78	[TCP Dup ACK 2#3] 5001 - 36300 [ACK] Seq=1 Ack=1 Win=2896 Len=0 TSval=104619224...
11	0.264912	192.168.1.100	10.10.1.100	TCP	798	[TCP Fast Retransmission] 36300 - 5001 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=724...
12	0.374813	10.10.1.100	192.168.1.100	TCP	66	[TCP ZeroWindow] 5001 - 36300 [ACK] Seq=1 Ack=2897 Win=0 Len=0 TSval=104619334...
13	0.375807	10.10.1.100	192.168.1.100	TCP	66	[TCP Window Update] 5001 - 36300 [ACK] Seq=1 Ack=2897 Win=0 TSval=1046...
14	0.375812	192.168.1.100	10.10.1.100	TCP	1514	36300 - 5001 [PSH, ACK] Seq=2897 Ack=1 Win=65538 Len=1448 TSval=3368478710 TSecr=104619224...
15	0.375814	192.168.1.100	10.10.1.100	TCP	1514	36300 - 5001 [FIN, PSH, ACK] Seq=3435 Ack=1 Win=65536 Len=1448 TSval=3368478710...
16	0.4865173	10.10.1.100	192.168.1.100	TCP	66	5001 - 36300 [ACK] Seq=1 Ack=4345 Win=1448 Len=0 TSval=1046194466 TSecr=33684787...
17	0.4865209	10.10.1.100	192.168.1.100	TCP	66	[TCP Window Update] 5001 - 36300 [ACK] Seq=1 Ack=4345 Win=2896 Len=0 TSval=1046...
18	0.802706	192.168.1.100	10.10.1.100	TCP	790	[TCP Retransmission] 36300 - 5001 [FIN, PSH, ACK] Seq=5069 Ack=1 Win=65536 Len=...
19	1.264934	192.168.1.100	10.10.1.100	TCP	790	[TCP Retransmission] 36300 - 5001 [ACK] Seq=4345 Ack=1 Win=65536 Len=724 TSval=...
20	1.366420	10.10.1.100	192.168.1.100	TCP	66	5001 - 36300 [ACK] Seq=1 Ack=5069 Win=2172 Len=0 TSval=104628322 TSecr=33684796...
21	1.366443	192.168.1.100	10.10.1.100	TCP	798	[TCP Retransmission] 36300 - 5001 [FIN, PSH, ACK] Seq=5069 Ack=1 Win=65536 Len=...
22	1.471844	10.10.1.100	192.168.1.100	TCP	66	5001 - 36300 [ACK] Seq=1 Ack=5794 Win=2896 Len=0 TSval=104628432 TSecr=33684797...
23	1.482188	10.10.1.100	192.168.1.100	TCP	66	5001 - 36300 [FIN, ACK] Seq=1 Ack=5794 Win=2896 Len=0 TSval=104620442 TSecr=336...
24	1.482201	192.168.1.100	10.10.1.100	TCP	66	36300 - 5001 [ACK] Seq=5794 Ack=2 Win=65536 Len=0 TSval=3368478918 TSecr=104620...

```
▶ Frame 11: 790 bytes on wire (6320 bits), 790 bytes captured (6320 bits)
▶ Ethernet II, Src: D6:81:08:d4:44:bc (d6:81:08:d4:44:bc), Dst: 42:19:1e:01:2c:f8 (42:19:1e:01:2c:f8)
▶ Internet Protocol Version 4, Src: 192.168.1.100, Dst: 10.10.1.100
▼ Transmission Control Protocol, Src Port: 36300, Dst Port: 5001, Seq: 1, Ack: 1, Len: 724
    Source Port: 36300
    Destination Port: 5001
    [Stream index: 0]
    [TCP Segment Len: 724]
    Sequence number: 1      (relative sequence number)
    Sequence number (raw): 1727186960
    [Next sequence number: 725      (relative sequence number)]
    Acknowledgment number: 1      (relative ack number)
    Acknowledgment number (raw): 3164957388
    1000 .... = Header Length: 32 bytes (8)
    ▶ Flags: 0x0108 (PSH, ACK)
    Window size value: 1024
    [Calculated window size: 65536]
    [Window size scaling factor: 64]
    Checksum: 0xd074 [unverified]
    [Checksum Status: Unverified]
    Urgent pointer: 0
▶ Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
▼ [SEQ/ACK analysis]
    [iRTT: 0.163986000 seconds]
    [Bytes in flight: 2896]
    [Bytes sent since last PSH flag: 724]
▼ [TCP Analysis Flags]
    ▶ [Expert Info (Note/Sequence): This frame is a (suspected) fast retransmission]
    ▶ [Expert Info (Note/Sequence): This frame is a (suspected) retransmission]
▶ [Timestamps]
    TCP payload (724 bytes)
    Data (724 bytes)
```

```
0020 01 64 8d cc 13 89 66 f2 c8 10 bc a5 6a cc 80 18 d...f...j...`  
0030 04 00 0d 74 00 00 01 01 08 0a c8 c6 77 88 03 ..c...`  
0040 5c d8 00 00 00 34 35 37 38 39 30 31 32 33 32 ..\`..45 67890123  
0050 34 35 36 37 38 39 30 31 32 33 34 35 36 37 38 39 456789012345678901  
0060 30 31 32 33 34 35 36 37 38 39 30 31 32 33 34 35 0123456789012345
```

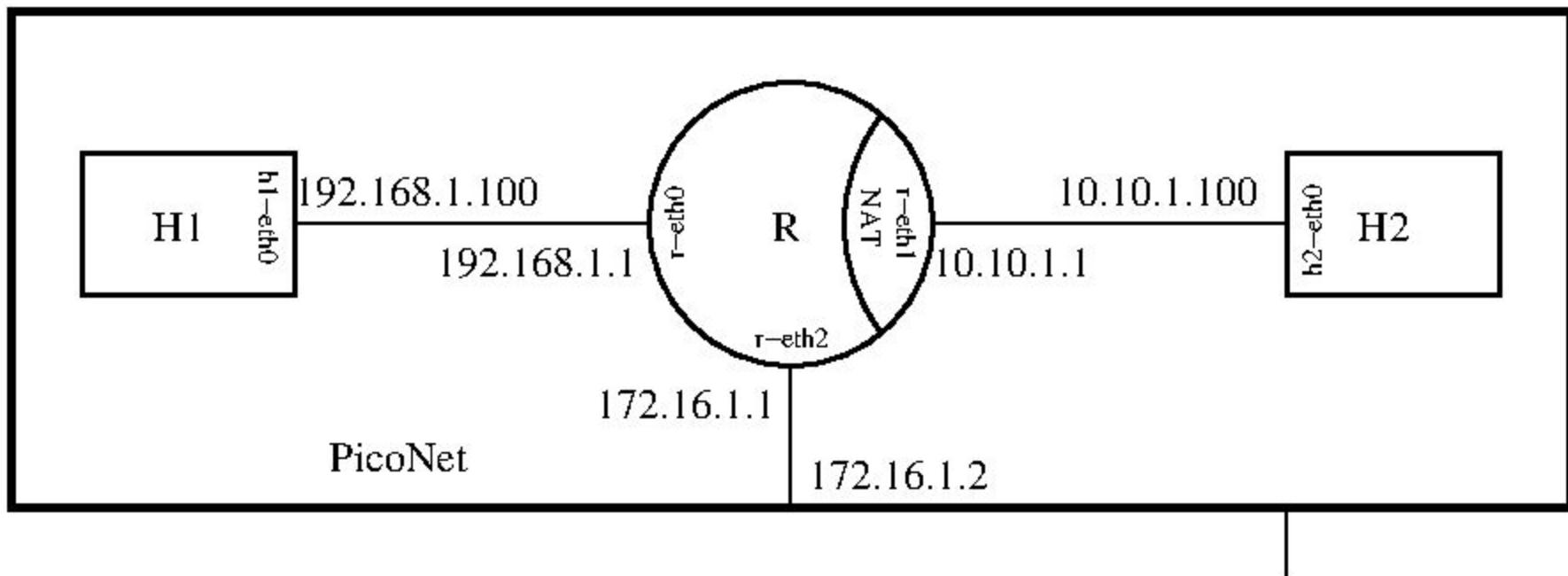
TCP congestion control

- 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:~# 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:~#
```

Terminal 3

```
root@h1:~# 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:~# 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:~$
```

After NAT

The screenshot shows a terminal window with four panes, each displaying command-line output from a host machine named "jupyter-pan".

- Terminal 1:** Shows the results of a ping test between two hosts, h1 and h2. Both hosts show 0% packet loss.
- Terminal 3:** Shows the results of a ping test between host h2 and host h1. Host h2 shows 0% packet loss.
- Terminal 2:** Shows the results of a ping test between host h2 and host h1. Host h2 shows 0% packet loss.
- Terminal 4:** Shows the configuration of iptables on the host to enable IP masquerading for traffic from host h2.

```
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> []

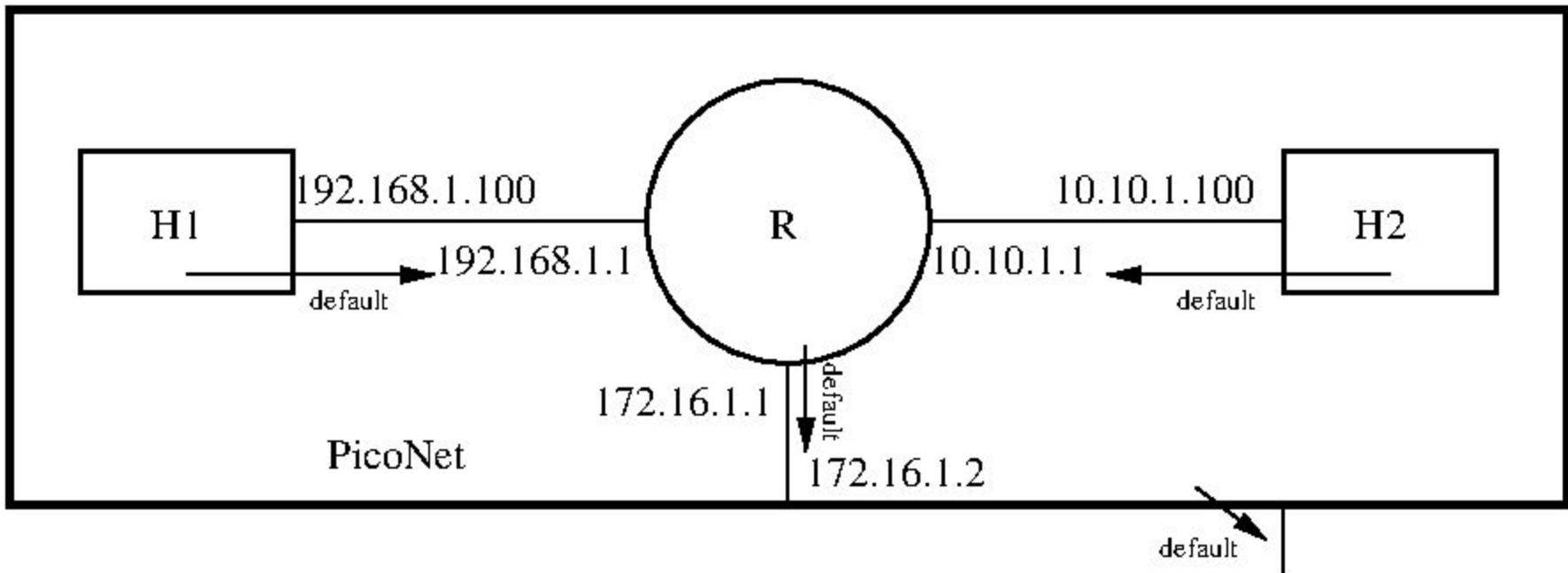
Terminal 3 ×
root@h1:~# piconet> tcpdump -l -n -i r-eth0 icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on r-eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
00:22:41.805907 IP 192.168.1.100 > 10.10.1.100: ICMP echo request, id 412, seq 1, length 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 ×
joyvan@jupyter-pan:~$ sudo ip netns exec r iptables -t nat -A POSTROUTING -o r-eth1 -j MASQ
UERADE
joyvan@jupyter-pan:~$
```

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

The screenshot shows a terminal window with four tabs, each displaying the output of the 'ip r' command. The tabs are labeled Terminal 1, Terminal 2, Terminal 3, and Terminal 4.

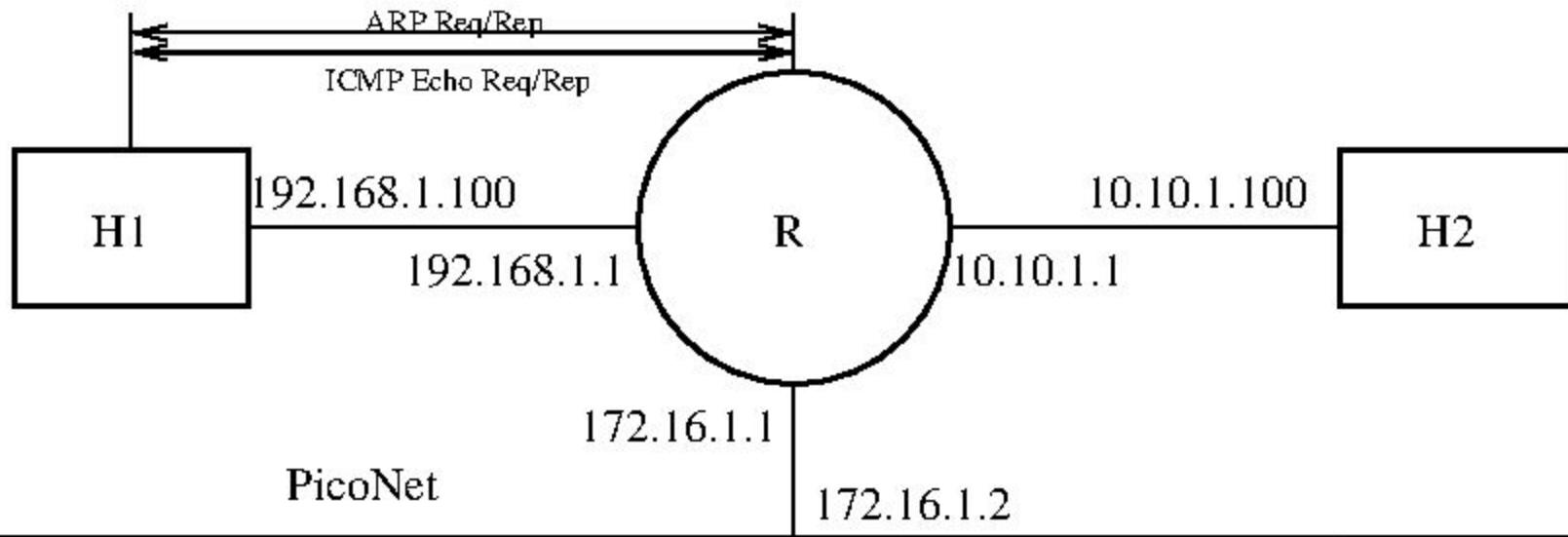
- Terminal 1:** Shows a default route via interface h1-eth0 with IP 192.168.1.1. The command is: `root@h1:~# ip r`.
Output:

```
root@h1:~# ip r
default via 192.168.1.1 dev h1-eth0
192.168.1.0/24 dev h1-eth0 proto kernel scope link src 192.168.1.100
root@h1:~# piconet> []
```
- Terminal 2:** Shows a default route via interface h2-eth0 with IP 10.10.1.1. The command is: `root@h2:~# ip r`.
Output:

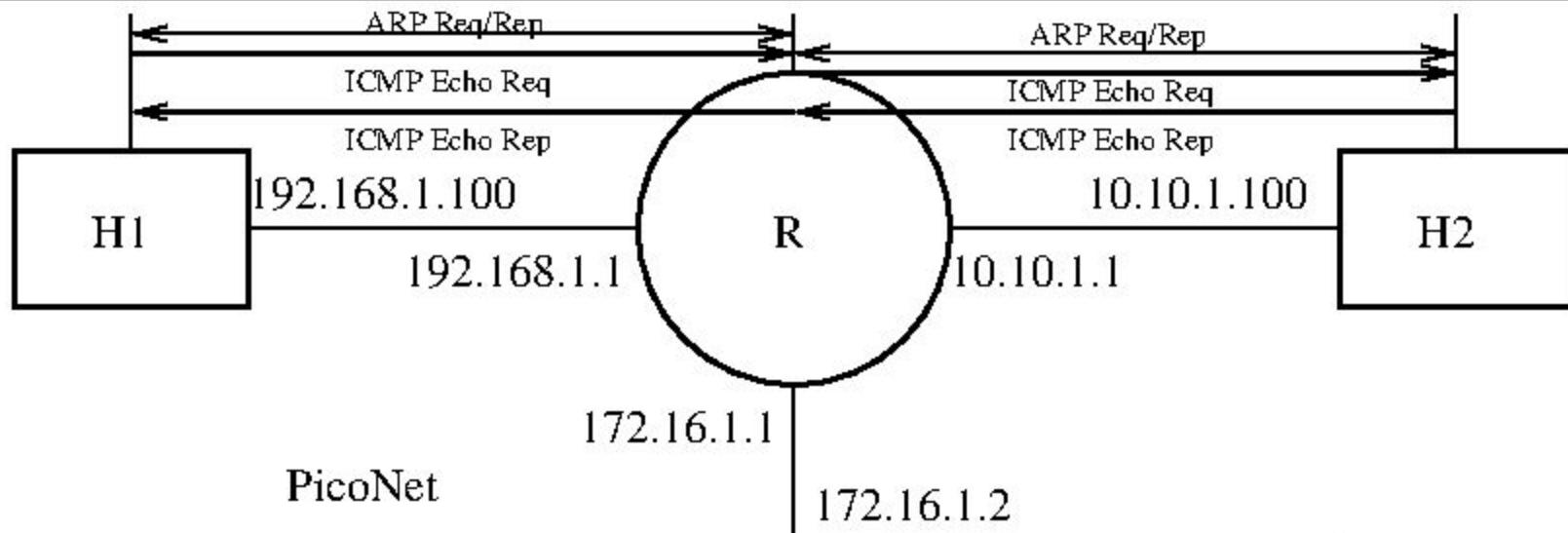
```
root@h2:~# ip r
default via 10.10.1.1 dev h2-eth0
10.10.1.0/24 dev h2-eth0 proto kernel scope link src 10.10.1.100
root@h2:~# piconet> []
```
- Terminal 3:** Shows a default route via interface r-eth2 with IP 172.16.1.2. The command is: `root@r:~# ip r`.
Output:

```
root@r:~# ip r
default via 172.16.1.2 dev r-eth2
10.10.1.0/24 dev r-eth1 proto kernel scope link src 10.10.1.1
172.16.1.0/24 dev r-eth2 proto kernel scope link src 172.16.1.1
192.168.1.0/24 dev r-eth0 proto kernel scope link src 192.168.1.1
root@r:~# piconet> []
```
- Terminal 4:** Shows a default route via interface eth0 with IP 172.17.0.1. The command is: `jovyan@jupyter-pan:~$ ip r`.
Output:

```
jovyan@jupyter-pan:~$ ip r
default via 172.17.0.1 dev eth0
172.16.1.0/24 via 172.16.1.2 dev host-nat
172.17.0.0/16 dev eth0 proto kernel scope link src 172.17.0.10
jovyan@jupyter-pan:~$
```



ARP/Interworking Lab



Within the same subnet (routing not involved)

File Edit View Run Kernel Git Tabs Settings Help

Terminal 1

```
root@h1 piconet> ping -c 1 192.168.1.1
PING 192.168.1.1 (192.168.1.1) 56(84) bytes of data.
64 bytes from 192.168.1.1: icmp_seq=1 ttl=64 time=0.099 ms

--- 192.168.1.1 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.099/0.099/0.099/0.000 ms
root@h1 piconet> arp
Address      HWtype  HWaddress        Flags Mask     Iface
r-net1       ether    a6:76:62:fc:02:fd  C          h1-eth0
root@h1 piconet>
```

Terminal 2

```
root@h2 piconet> 
```

Terminal 3

```
root@h2 piconet> tcpdump -l -n -i r-eth0 -e
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on r-eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
13:52:39.287252 4e:0b:73:25:7e:05 > ff:ff:ff:ff:ff:ff, ethertype ARP (0x0806), length 42:
Request who-has 192.168.1.1 tell 192.168.1.100, length 28
13:52:39.287290 a6:76:62:fc:02:fd > 4e:0b:73:25:7e:05, ethertype ARP (0x0806), length 42:
Reply 192.168.1.1 is-at a6:76:62:fc:02:fd, length 28
13:52:39.287299 4e:0b:73:25:7e:05 > a6:76:62:fc:02:fd, ethertype IPv4 (0x0800), length 98
: 192.168.1.100 > 192.168.1.1: ICMP echo request, id 301, seq 1, length 64
13:52:39.287319 a6:76:62:fc:02:fd > 4e:0b:73:25:7e:05, ethertype IPv4 (0x0800), length 98
: 192.168.1.1 > 192.168.1.100: ICMP echo reply, id 301, seq 1, length 64
13:52:44.398638 a6:76:62:fc:02:fd > 4e:0b:73:25:7e:05, ethertype ARP (0x0806), length 42:
Request who-has 192.168.1.100 tell 192.168.1.1, length 28
13:52:44.398741 4e:0b:73:25:7e:05 > a6:76:62:fc:02:fd, ethertype ARP (0x0806), length 42:
Reply 192.168.1.100 is-at 4e:0b:73:25:7e:05, length 28
```

Terminal 4

```
jovyan@jupyter-pan:~$ sudo ip netns exec r tcpdump -l -n -i r-eth1 -e
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on r-eth1, link-type EN10MB (Ethernet), capture size 262144 bytes
[ ]
```

Between different subnets (routing involved too)

The screenshot shows four terminal windows in a terminal application interface:

- Terminal 1:** Shows ping statistics from 192.168.1.1 to 192.168.1.100 and 10.10.1.100.
- Terminal 2:** Shows the root prompt on a host named piconet.
- Terminal 3:** Shows a continuous stream of TCPdump output capturing traffic on interface r-eth0, showing ARP and ICMP requests/replies between 192.168.1.1 and 10.10.1.100.
- Terminal 4:** Shows a continuous stream of TCPdump output capturing traffic on interface r-eth1, showing ARP and ICMP requests/replies between 10.10.1.100 and 192.168.1.100.

```
File Edit View Run Kernel Git Tabs Settings Help
S Terminal 1 ×
--- 192.168.1.1 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.099/0.099/0.099/0.000 ms
root@piconet:~# arp
Address      HWtype  HWaddress      Flags Mask     Iface
r-net1        ether   a6:76:62:fc:02:fd  C          h1-eth0
root@piconet:~# arp -d r-net1
root@piconet:~# arp
root@piconet:~# ping -c 1 10.10.1.100
PING 10.10.1.100 (10.10.1.100) 56(84) bytes of data.
64 bytes from 10.10.1.100: icmp_seq=1 ttl=63 time=0.278 ms

--- 10.10.1.100 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.278/0.278/0.278/0.000 ms
root@piconet:~# ping -c 1 192.168.1.100
PING 192.168.1.100 (192.168.1.100) 56(84) bytes of data.
64 bytes from 192.168.1.100: icmp_seq=1 ttl=63 time=0.278 ms

--- 192.168.1.100 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.278/0.278/0.278/0.000 ms
root@piconet:~# ping -c 1 192.168.1.100
PING 192.168.1.100 (192.168.1.100) 56(84) bytes of data.
64 bytes from 192.168.1.100: icmp_seq=1 ttl=63 time=0.278 ms

S Terminal 2 ×
root@piconet:~# ping -c 1 10.10.1.100
PING 10.10.1.100 (10.10.1.100) 56(84) bytes of data.
64 bytes from 10.10.1.100: icmp_seq=1 ttl=63 time=0.278 ms

S Terminal 3 ×
root@piconet:~# tcpdump -l -n -i r-eth0 -e
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on r-eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
13:53:43.723615 4:e:0:b:73:25:7e:05 > ff:ff:ff:ff:ff:ff, ethertype ARP (0x0806), length 42:
Request who-has 192.168.1.1 tell 192.168.1.100, length 28
13:53:43.723658 a6:76:62:fc:02:fd > 4:e:0:b:73:25:7e:05, ethertype ARP (0x0806), length 42:
Reply 192.168.1.1 is-at a6:76:62:fc:02:fd, length 28
13:53:43.723666 4:e:0:b:73:25:7e:05 > a6:76:62:fc:02:fd, ethertype IPv4 (0x0800), length 98
: 192.168.1.100 > 10.10.1.100: ICMP echo request, id 309, seq 1, length 64
13:53:43.723857 a6:76:62:fc:02:fd > 4:e:0:b:73:25:7e:05, ethertype IPv4 (0x0800), length 98
: 10.10.1.100 > 192.168.1.100: ICMP echo reply, id 309, seq 1, length 64
13:53:48.910827 a6:76:62:fc:02:fd > 4:e:0:b:73:25:7e:05, ethertype ARP (0x0806), length 42:
Request who-has 192.168.1.100 tell 192.168.1.1, length 28
13:53:48.910941 4:e:0:b:73:25:7e:05 > a6:76:62:fc:02:fd, ethertype ARP (0x0806), length 42:
Reply 192.168.1.100 is-at 4:e:0:b:73:25:7e:05, length 28
S Terminal 4 ×
jovyan@jupyter-pan:~$ sudo ip netns exec r tcpdump -l -n -i r-eth1 -e
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on r-eth1, link-type EN10MB (Ethernet), capture size 262144 bytes
13:53:43.723683 92:bd:67:5e:7c:9d > ff:ff:ff:ff:ff:ff, ethertype ARP (0x0806), length 42:
Request who-has 10.10.1.100 tell 10.10.1.1, length 28
13:53:43.723711 c6:f0:f7:7e:f6:b7 > 92:bd:67:5e:7c:9d, ethertype ARP (0x0806), length 42:
Reply 10.10.1.100 is-at c6:f0:f7:7e:f6:b7, length 28
13:53:43.723801 92:bd:67:5e:7c:9d > c6:f0:f7:7e:f6:b7, ethertype IPv4 (0x0800), length 98
: 192.168.1.100 > 10.10.1.100: ICMP echo request, id 309, seq 1, length 64
13:53:43.723842 c6:f0:f7:7e:f6:b7 > 92:bd:67:5e:7c:9d, ethertype IPv4 (0x0800), length 98
: 10.10.1.100 > 192.168.1.100: ICMP echo reply, id 309, seq 1, length 64
13:53:48.910873 c6:f0:f7:7e:f6:b7 > 92:bd:67:5e:7c:9d, ethertype ARP (0x0806), length 42:
Request who-has 10.10.1.1 tell 10.10.1.100, length 28
13:53:48.910967 92:bd:67:5e:7c:9d > c6:f0:f7:7e:f6:b7, ethertype ARP (0x0806), length 42:
Reply 10.10.1.1 is-at 92:bd:67:5e:7c:9d, length 28
```