

# 13 – Networking and Package Management

CS 2043: Unix Tools and Scripting, Spring 2019 [1]

---

Matthew Milano

February 20, 2019

Cornell University

# Table of Contents

---

## 1. THE INTERNET

As always: Everybody! ssh to wash.cs.cornell.edu

- Quiz time! Everybody! run **quiz-02-20-19**
- You can just explain a concept from last class, doesn't have to be a command this time.

# THE INTERNET

---

# How do computers communicate?

- Send data back and forth
- Data takes the form of *packets*



Figure 1: Paper airplane

## Throwing paper airplanes blind???

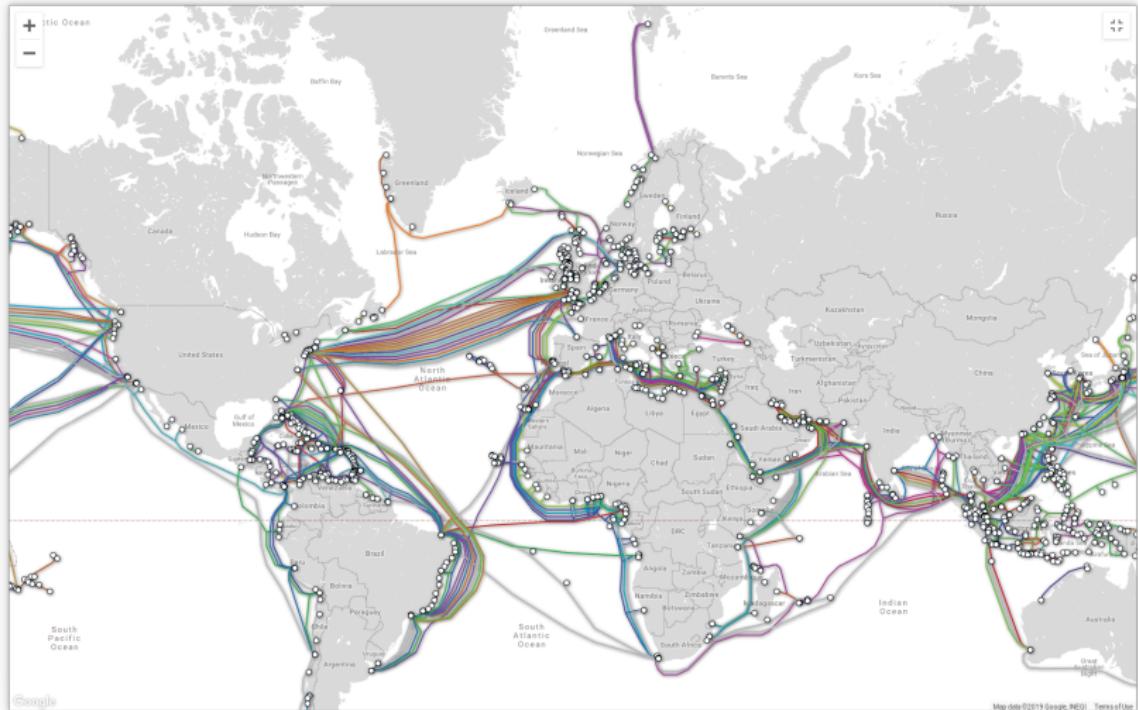
- Surprisingly accurate metaphor.
- You might miss! Need to acknowledge receipt
- You might throw at different speeds! Need sequence numbers
- these things (and more!) handled by the TCP protocol

## Ok, but what's really going on?

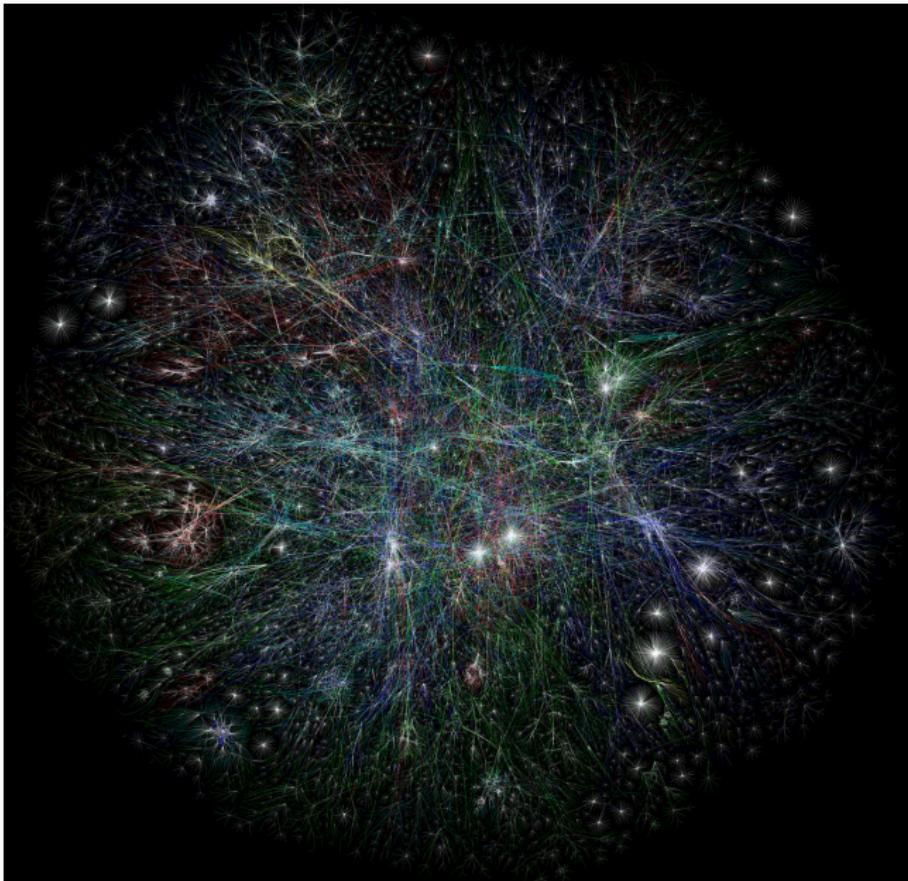
- Network cards convert local communication to network communication
- Limited memory and processing power
- Packet is small, “safe” size
  - will always fit in network hardware memory
  - is very fast to transfer; can’t easily “split” a packet
- once a packet arrives, networking hardware moves it as “data” to your application
- gives the illusion of a stream of packets
  - or an unending torrent of paper airplanes...

# WHAT IS... THE INTERNET

- interconnected networks of computers



# WHAT IS... THE INTERNET



Ok, smaller scale

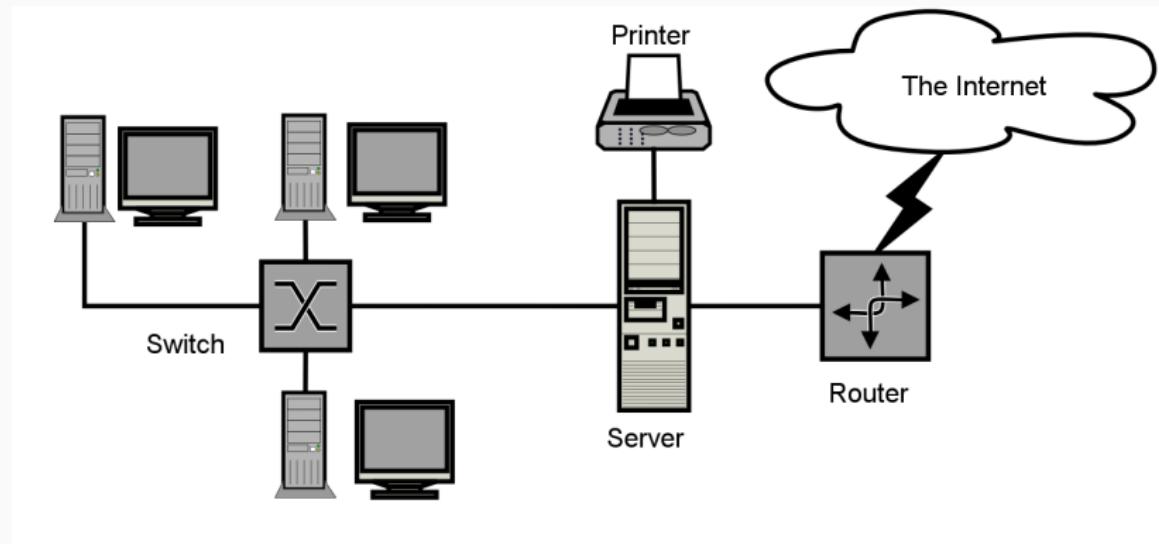


Figure 4: Small network

# Finding your target

- You are *directly* connected to a switch
  - you send all your traffic there
  - the *switch* needs to know where to forward it
- Need an *address*: a destination the switch can understand
- We use IP addresses
  - just a sequence of numbers
- Can also *broadcast*
  - everyone gets it
  - usually blocked by firewalls outside of your local network

## IPv4 Addresses:

- Usually displayed as **XXX.XXX.XXX.XXX**
  - example: **192.168.1.101**
  - example: **128.84.216.194**
- GLOBALLY routable: (most) IP addresses globally unique!
  - a few small exceptions: **192.168.\***, **10.\***
  - You can use these locally, but nobody can reach them from the wider internet.
  - think of them like private roads in the address system.
- How do we know they're globally unique?
  - Convention! The ICANN tells you what your number is, gets angry when you use the wrong one

## Getting an IP address

- IP addresses change based on your network
- Each internet connection has its own “slice” of the IP space
  - computers attached to that network use its “slice”
- Can just give yourself an IP address
  - things break if you go outside the slice
- Can ask for help via DHCP
  - some server will give you an address
  - it probably knows which addresses are valid
  - it can tell when they’re used.
- ever see a **169.254.XX.YY** address? That’s your computer wildly guessing!

## MAC Addresses:

- based on your actual exact hardware
- does not change based on network
- only used for *local* forwarding
  - within the same route
  - we'll talk about that later
- How switches actually connect to your device
- use the ARP protocol to map IP to MAC
  - Literally broadcast “who has 192.168.1.1? Tell 192.168.1.5”

# Command break: ifconfig

configure your internet devices

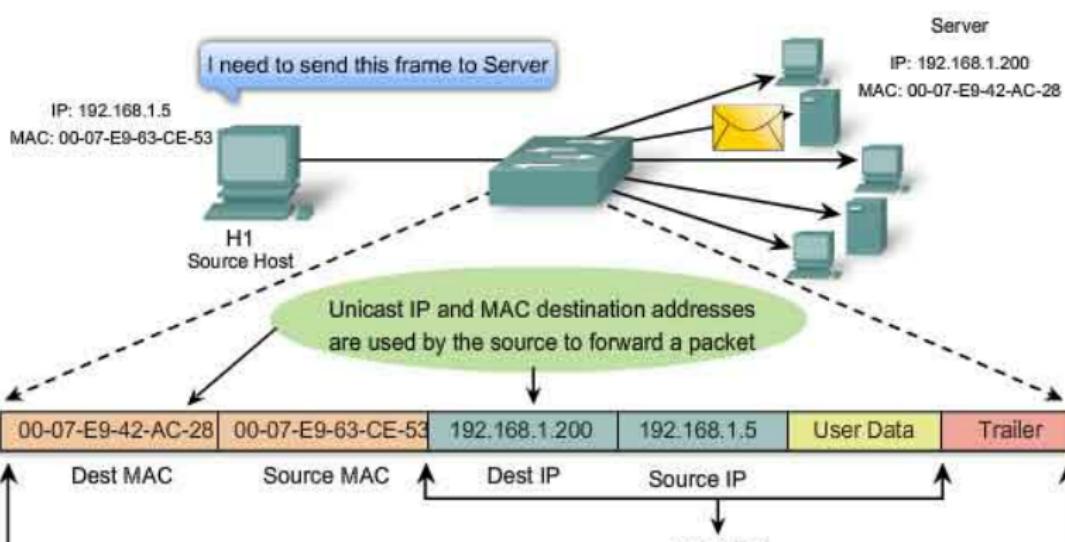
ifconfig [Options...]

```
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
      inet 127.0.0.1 netmask 255.0.0.0
      inet6 ::1 prefixlen 128 scopeid 0x10<host>
        loop txqueuelen 1000 (Local Loopback)
        RX packets 40 bytes 2357 (2.3 KiB)
        RX errors 0 dropped 0 overruns 0 frame 0
        TX packets 40 bytes 2357 (2.3 KiB)
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlp3s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
      inet 10.132.6.227 netmask 255.255.128.0 broadcast 10.132.127.255
      inet6 fe80::4141:aa4f:3c83:5a88 prefixlen 64 scopeid 0x20<link>
        ether 00:23:15:f0:91:41 txqueuelen 1000 (Ethernet)
        RX packets 97595 bytes 34824177 (33.2 MiB)
        RX errors 0 dropped 0 overruns 0 frame 0
        TX packets 69065 bytes 34033683 (32.4 MiB)
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

# Making packets

- Ethernet “frames” and IP “packets” are not exactly the same thing
- Like putting smaller envelopes in bigger ones
- “Wrap” Ethernet information around IP information



# Routing packets

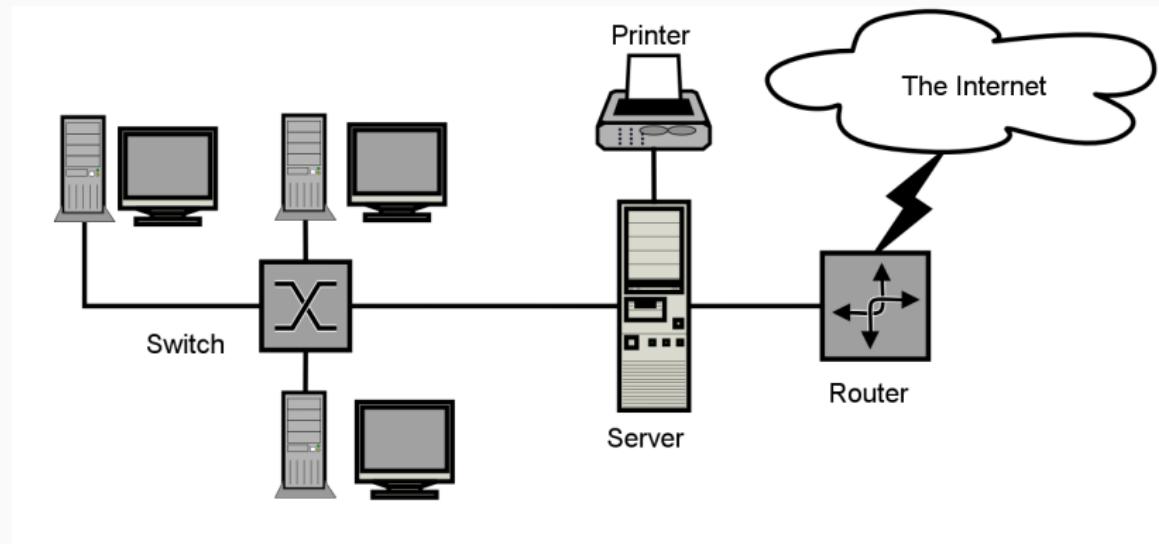


Figure 6: Small network

## Routing packets

- Routers jump across network boundaries
- Limit of MAC protocol; MAC routes locally, IP routes globally
- Your *netmask* defines local IP addresses
  - Example: My IP is **192.168.1.100** and my netmask is **255.255.255.0**
  - Everything that starts with **192.168.1** is *local*; everything else needs *routing*
- Your *gateway* handles routing
  - Example: My IP is **192.168.1.100** and my gateway is **192.168.1.1**
  - Gateways have lots of IP addresses; they're connected to lots of networks!
  - Like this one Island in Canada...

# Routing packets

- All IP addresses outside of local network use Router as initial destination
- Router re-packages packet for new network, acts as sender in that network
  - it will eventually hit a new network, and do that again

trace the **route** your packet takes

**traceroute <dest\_addr>**

- lists each routing hop between you and destination
- displays *DNS name* or IP address

# Traceroute example

Example:

```
$ traceroute google.com
traceroute to google.com (172.217.12.174), 30 hops max, 60 byte packets
1 rhodes1-ex-vl1245.net.cornell.edu (128.84.216.1) 1.372 ms 1.334 ms 1.355 ms
2 core1-mx-xe-11-0-4.net.cornell.edu (128.253.222.161) 0.720 ms 0.706 ms core2-mx-xe-11-0
3 wan2-mx-et-0-0-0.net.cornell.edu (128.253.222.58) 0.570 ms 0.568 ms 0.564 ms
4 199.109.109.25 (199.109.109.25) 3.089 ms 3.019 ms 3.021 ms
5 199.109.107.162 (199.109.107.162) 8.076 ms 8.042 ms 8.064 ms
6 72.14.202.166 (72.14.202.166) 8.007 ms 8.083 ms 8.054 ms
7 108.170.248.1 (108.170.248.1) 10.017 ms 10.052 ms 9.915 ms
8 108.170.226.201 (108.170.226.201) 9.055 ms 108.170.226.199 (108.170.226.199) 9.017 ms 1
9 lga25s62-in-f14.le100.net (172.217.12.174) 8.996 ms 8.992 ms 8.961 ms
```

## DNS names

- A big dictionary in the sky
- translates english *domain* names (like **google.com**) into IP addresses
- Also managed by ICANN

### domain information proper

```
dig [@DNS server] [domain name]
```

- gives you a lot of information behind the domain name
- includes IP address, owner, owner's e-mail address, and more

# dig example

```
$ dig @1.1.1.1 google.com

; <>>> DiG 9.12.2-P2 <>>> @1.1.1.1 google.com
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 22816
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1452
;; QUESTION SECTION:
;google.com.           IN      A

;; ANSWER SECTION:
google.com.        215     IN      A      216.58.219.206

;; Query time: 10 msec
;; SERVER: 1.1.1.1#53(1.1.1.1)
;; WHEN: Wed Feb 20 10:51:56 EST 2019
;; MSG SIZE  rcvd: 55
```

# References

---

- [1] Stephen McDowell, Bruno Abraha, Hussam Abu-Libdeh, Nicolas Savva, David Slater, and others over the years.  
“Previous Cornell CS 2043 Course Slides”.