Administivia

- **IP Project**: out later today
  - Partner form: due TONIGHT by 11:59pm
  - You will get an email confirming your team tomorrow morning

- **IP gearup**: tonight 5-7pm, CIT368

- **IP Milestone**: meet with me/a TA on/before next Friday (October 6) to discuss your design
  - (No working code yet, just some serious plans/sketches)

- **HW1 (short)**: Due next Thursday
Today

Continuing network layer

• IP forwarding mechanics
• About the IP project
The story so far:

=> Can “easily” communicate with nodes on the same network, but what about other networks?

=> Routers know about multiple networks, forward packets between them.
What does it mean to be on the same network?
All systems with an IP address have a configuration like this:

```
Addr: 138.16.161.209
Mask: 255.255.255.0
```

```
10001010 00010000 10100001 11010001
& 11111111 11111111 11111111 00000000

00001010 00010000 10100001 00000000
```

24 bits 138.16.161.0

256 possible hosts
All systems with an IP address have a configuration like this

```
138.16.161.209
10001010 00010000 10100001 11010001
138.16.161.209
255.255.255.0
11111111 11111111 11111111 00000000
```

Can also write as 138.16.161.209/24

“Prefix notation” or “CIDR notation”
All systems with an IP address have a configuration like this

Addr: 138.16.161.209

Mask: 255.255.255.0
All systems with an IP address have a configuration like this

138.16.161.209

Addr: 138.16.161.209
Mask: 255.255.255.0

10001010 00010000 10100001 11010001
11111111 11111111 11111111 00000000

=> Bitmask used to “filter out” which part is for hosts on the same network
Assigned prefix for this network

138.16.161.0/24

138.16.161.204

1.2.3.4

\[ \begin{aligned}
10001010 & 00010000 & 10100001 & \ldots \\
10001010 & 00010000 & 10100001 & 10100001 \\
00000001 & 00000010 & 00000011 & 00000100 \\
\end{aligned} \]

\( N \Rightarrow \) First \( N \) bits are network part.
The mask can be any size 0-32
Not just checking the first three digits!
Common prefixes

1.2.0.0/16 00000001 00000010 xxxxxxxx xxxxxxxx

8.0.0.0/8 00001000 xxxxxxxx xxxxxxxx xxxxxxxx

123.10.1.0/24 01111011 00001010 00000001 xxxxxxxx

201.112.10.200/30 11001001 01110000 00001010 110010xx
Oldest Allocations

This chart shows the IPv6 address space on a plane using a fractal mapping which preserves grouping. Any consecutive string of IPv6 will translate to a single compact, contiguous region on the map. Each of the 256 numbered blocks represents one sub-net (containing all IPv6s that start with that number). The upper left section shows the blocks sold directly to corporations and governments in the 1990s before the RIRs took over allocation.
How many addresses are in the network 1.1.0.0/20?

Is 1.1.16.1 in this prefix?

NO.
**CIDR** is a notation for describing blocks of IP addresses and is used heavily in various networking configurations. IP addresses contain 4 octets, each consisting of 8 bits giving values between 0 and 255. The decimal value that comes after the slash is the number of bits consisting of the routing prefix. This in turn can be translated into a netmask, and also designates how many available addresses are in the block.

CIDR Base IP: 1.1.0.0
Broadcast IP: 1.1.15.255
Count: 4,096

* For routing mask values <= 30, first and last IPs are base and broadcast addresses and are unusable.

Created by [Yuval Adam](https://github.com). Source available on [Github](https://github.com).
How do we move packets between networks?
Forwarding IP packets

To more networks (i.e., Internet)

Src: 1.2.1.3
Dst: 1.2.2.100

1.2.1.3
1.2.1.1
1.2.1.200
1.2.2.1
1.2.2.100
1.2.2.105

/1.2.1.0/24
/1.2.2.0/24
IP forwarding

Decide where to send packets based on forwarding table

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Interface/Next hop</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Key Type:** An IP prefix (1.2.1.0/24)

Value type: Multiple forms

- **Interface (IF0):** “This is my neighbor (on local net), link-layer can figure it out”
  => “Local delivery”

- **Next hop IP** (eg. 1.1.1.1): send packet to this IP instead
  => Need to search for next hop in table!
Forwarding IP packets

Src: 1.2.1.3
Dst: 1.2.2.100

To more networks (ie, Internet)
Forwarding IP packets

Prefix | Interface/Next hop
--- | ---
1.2.1.0/24 | IF1
1.2.2.0/24 | IF2
Forwarding IP packets

To more networks (i.e., Internet)

Prefix | Interface
--- | ---
1.2.1.0/24 | IF1
1.2.2.0/24 | IF2
... | ...

1.2.1.2
1.2.1.3
1.2.1.200
01:A5:27::...

Src: 1.2.1.3
Dst: 1.2.2.100

01:A5:27::...
What about the rest?

How to reach networks that aren’t directly connected?

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Interface/Next hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.2.1.0/24</td>
<td>IF1</td>
</tr>
<tr>
<td>1.2.2.0/24</td>
<td>IF2</td>
</tr>
<tr>
<td>&lt;everything else&gt;</td>
<td></td>
</tr>
</tbody>
</table>

To more networks (ie, Internet)
Wi-Fi

Configure IPv4: Using DHCP
IPv4 Address: 138.16.161.209
Subnet Mask: 255.255.255.0
Router: 138.16.161.1

DEFAULT GATEWAY,
Default gateway: where to send to reach anything not in the table

=> Use "Next hop IP" in table

Gateway: device at the "edge" of a network (eg. Brown <-> Internet)

<table>
<thead>
<tr>
<th>Prefix</th>
<th>IF/Next hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.2.1.0/24</td>
<td>IF1</td>
</tr>
<tr>
<td>1.2.2.0/24</td>
<td>IF2</td>
</tr>
<tr>
<td>8.0.0.0/30</td>
<td>IF0</td>
</tr>
<tr>
<td>0.0.0.0/0</td>
<td>8.0.0.2</td>
</tr>
</tbody>
</table>

⇒ 0.0.0.0 matches on everything! Problem?

Can have multiple matches in table => use the most specific (longest) prefix (more on this later)
Does it scale?

<table>
<thead>
<tr>
<th>Prefix</th>
<th>IF/Next hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.2.1.0/24</td>
<td>IF1</td>
</tr>
<tr>
<td>1.2.2.0/24</td>
<td>IF2</td>
</tr>
<tr>
<td>8.0.0.0/30</td>
<td>IF0</td>
</tr>
<tr>
<td>0.0.0.0/0</td>
<td>8.0.0.2</td>
</tr>
</tbody>
</table>

Yes! (At least enough to make the Internet as we know it…)

- Forward packets based on IP prefixes
- Don’t need to keep track of every single host
- Routers at the “edges” of the network don’t need to know about every route

- Larger, highly-connected routers (“core routers”) do need very large tables, specialized hardware, optimization tricks…
Map of the Internet, 2021 (via BGP)
OPTE project
A forwarding table (my laptop)

deemer@ceres ~ % ip route
default via 10.3.128.1 dev wlp2s0
10.3.128.0/18 dev wlp2s0 proto dhcp scope link src 10.3.135.44 metric 3003
172.18.0.0/16 dev docker0 proto kernel scope link src 172.18.0.1
192.168.1.0/24 dev enp0s31f6 proto kernel scope link src 192.168.1.1
A routing table

R6#sh ip ro
Gateway of last resort is 108.34.215.1 to network 0.0.0.0

S* 0.0.0.0/0 [1/0] via 108.34.215.1
   10.0.0.0/8 is variably subnetted, 7 subnets, 3 masks
C  10.1.0.0/24 is directly connected, wlan-ap0
L  10.1.0.2/32 is directly connected, wlan-ap0
O IA 10.1.44.1/32 [110/1001] via 10.20.30.33, 3w4d, Tunnel0
C  10.1.48.0/24 is directly connected, Loopback0
L  10.1.48.1/32 is directly connected, Loopback0
C  10.20.30.32/31 is directly connected, Tunnel0
L  10.20.30.32/32 is directly connected, Tunnel0

108.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C  108.34.215.0/24 is directly connected, GigabitEthernet0/0
L  108.34.215.208/32 is directly connected, GigabitEthernet0/0

172.16.0.0/16 is variably subnetted, 2 subnets, 2 masks
C  172.16.98.0/24 is directly connected, Vlan98
L  172.16.98.1/32 is directly connected, Vlan98

172.17.0.0/16 is variably subnetted, 6 subnets, 3 masks
O IA 172.17.44.0/24 [110/1001] via 10.20.30.33, 3w4d, Tunnel0
C  172.17.48.0/24 is directly connected, Vlan20
A routing table

```
R6#sh ip ro
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
      D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
      N1 - OSPF NSSA external type 1, NZ - OSPF NSSA external type 2
      E1 - OSPF external type 1, E2 - OSPF external type 2
      i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
      ia - IS-IS inter area, * - candidate default, U - per-user static route
      o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP
      + - replicated route, % - next hop override

Gateway of last resort is 108.34.215.1 to network 0.0.0.0

S*  0.0.0.0/0 [1/0] via 108.34.215.1
    10.0.0.0/8 is variably subnetted, 7 subnets, 3 masks
C   10.1.0.0/24 is directly connected, wlan-ap0
L   10.1.0.2/32 is directly connected, wlan-ap0
O IA 10.1.44.1/32 [110/1001] via 10.20.30.33, 3w4d, Tunnel0
C   10.1.48.0/24 is directly connected, Loopback0
L   10.1.48.1/32 is directly connected, Loopback0
```
show route table inet.0 active-path

inet.0: 866991 destinations, **13870153** routes (866991 active, 0 holddown, 0 hidden)

+ = Active Route, - = Last Active, * = Both

<table>
<thead>
<tr>
<th>IP Address</th>
<th>Route Type</th>
<th>Age</th>
<th>Local Prefs</th>
<th>Source</th>
<th>AS Path</th>
<th>Validation State</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0.0.0/0</td>
<td>Static/5</td>
<td>5w0d</td>
<td>19:43:09</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>to 12.0.1.1 via em0.0</td>
</tr>
<tr>
<td>1.0.0.0/24</td>
<td>BGP/170</td>
<td>1d 10:24:47</td>
<td>100</td>
<td>12.122.83.238</td>
<td>7018 3356 13335 I</td>
<td>valid</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>to 12.0.1.1 via em0.0</td>
</tr>
<tr>
<td>1.0.4.0/22</td>
<td>BGP/170</td>
<td>1d 10:24:47</td>
<td>100</td>
<td>12.122.83.238</td>
<td>7018 3356 4826 38803 I</td>
<td>valid</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>to 12.0.1.1 via em0.0</td>
</tr>
<tr>
<td>1.0.4.0/24</td>
<td>BGP/170</td>
<td>1d 10:24:47</td>
<td>100</td>
<td>12.122.83.238</td>
<td>7018 3356 4826 38803 I</td>
<td>valid</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>to 12.0.1.1 via em0.0</td>
</tr>
<tr>
<td>1.0.5.0/24</td>
<td>BGP/170</td>
<td>1d 10:24:47</td>
<td>100</td>
<td>12.122.83.238</td>
<td>7018 3356 4826 38803 I</td>
<td>valid</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>to 12.0.1.1 via em0.0</td>
</tr>
<tr>
<td>1.0.6.0/24</td>
<td>BGP/170</td>
<td>1d 10:24:47</td>
<td>100</td>
<td>12.122.83.238</td>
<td>7018 3356 4826 38803 I</td>
<td>valid</td>
</tr>
</tbody>
</table>
deemer@ceres ~ % ip route
default via 10.3.128.1 dev wlp2s0
10.3.128.0/18 dev wlp2s0 proto dhcp scope link src 10.3.135.44 metric 3003
172.18.0.0/16 dev docker0 proto kernel scope link src 172.18.0.1
192.168.1.0/24 dev enp0s31f6 proto kernel scope link src 192.168.1.1

Some router in
Brown.

Let's break this down further...
Example: Different networks on my laptop

Each interface is connected to a different network and has its own IP address.

Laptop decides how to forward packets to interfaces.

192.168.0.1
172.16.0.0/16
172.168.0.0/24
172.16.0.0/16

IP forwarding table:

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Interface/Net/Next</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.3.12.0/16</td>
<td>WiFi</td>
</tr>
<tr>
<td>172.16.0.0/24</td>
<td>ETHERNET</td>
</tr>
<tr>
<td>172.168.0.0/16</td>
<td>DOCKER</td>
</tr>
<tr>
<td>*</td>
<td>10.3.12.1/16</td>
</tr>
</tbody>
</table>

Examples:

IP: 192.168.0.2 => ETHERNET

IP: 10.3.12.1/16 = Default, look up 10.3.12.1 => WIFI.
How does forwarding actually work?
The IPv4 Header

```
<table>
<thead>
<tr>
<th>0</th>
<th>4</th>
<th>8</th>
<th>16</th>
<th>31</th>
<th>20  bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>IHL</td>
<td>TOS</td>
<td>Total length</td>
<td>Identification</td>
<td>Flags</td>
</tr>
<tr>
<td>TTL</td>
<td>Protocol</td>
<td>Header checksum</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Source address</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Destination address</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Options</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

On every IP packet
### The IPv4 Header

Defined by RFC 791

RFC (Request for Comment): defines network standard
Most Important fields

- **Version**: 4 for IPv4 packets, 6 for IPv6
- **Source address**: where the packet came from
- **Destination address**: where the packet is going

(continued…)

WHAT ROUTER MUST LOOK AT.
More important fields

- **TTL (time-to-live):** decremented each hop
  - Can prevent forwarding loops (and do other stuff…)

- **Checksum:** computed over header (very weak!)

- **Protocol identifier:** describes what’s in the packet
  - 6: TCP, 17: UDP, 1: ICMP, …
  - Defines the type of the payload
Less important fields

• Header length: in 32-bit units
  – >5 implies use of IP options
  – Almost all routers ignore IP options

• Fragmentation
  – Network can fragment a packet if next link requires a small frame
  – Most routers don’t fragment (or reassemble fragments)

• We won’t talk about…
  – Type of Service (TOS): basic traffic classification
  – Identifier: might have special meaning on some networks
Forwarding steps

What does a device do when it receives a packet?
Device: host, router, …
A “networking stack”
THINKING ABOUT A NETWORKING STACK.

OS

KERNEL

USER SPACE

TCP, SOCKET

IP

IP LAYER
NETWORK LAYER

IP: DO FORWARDING Lookup

ROUTER

IP.....

SNOWCAST... BROWSER ...

PSO SOCKET
A “networking stack”

A networking stack consists of several layers:

- **User space** and **Kernel space**
- **Applications**: High-level software that interacts with the user.
- **Sockets, TCP, etc.**: Middleware for communication between applications.
- **Network layer**: Responsible for data transmission across networks.
- **IP**: Internet Protocol, handles packet routing at the network layer.
- **Interfaces**: Physical connections to the network, like Ethernet or Wi-Fi.

On a router: packet might be for a different destination

=> send out another interface

If it’s for one for an IP assigned to this device, send to higher layer
Forwarding mechanics

When an IP packet arrives at a host/router:
IP Forwarding: Steps

What do you do when you get a packet?

1. Is the packet valid?
   - Is check sum valid? => IF NO, DROP
   - Is TP < 0? => DROP

2. Is it for me? => IF DEST IP == (any local IP)
   => SEND IT TO 'OS
      - Check if dest addr is for this host.
      - IF YES, send "upstream" to OS. => OS considers protocol
        field: TCP, UDP...

3. Is it for a local network? => Consider dest IP
   - Does packet match any networks in forwarding table?
   - IF YES, send on that interface.

4. Do I have a next hop? => IF SO, lookup interface
   FOR NEXT hop IP.
Looking at it in a different way.

1. Is it valid?
   - No: Drop
   - Yes: Check the forwarding table

2. My IP?
   - Yes: Send to OS
   - No: Send to that interface

3. Local interface?
   - Yes: Send to that interface

4. Next hop?

To send:
- Decrement TTL
- Recompute checksum
- Send on interface
Forwarding mechanics

When an IP packet arrives at a host/router:
• Is it valid? Verify checksum over header
How to avoid loops?

<table>
<thead>
<tr>
<th>Version</th>
<th>IHL</th>
<th>TOS</th>
<th>Total length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identification</td>
<td>Flags</td>
<td>Fragment offset</td>
<td></td>
</tr>
<tr>
<td>TTL</td>
<td>Protocol</td>
<td>Header checksum</td>
<td></td>
</tr>
</tbody>
</table>

**TTL (Time to Live):** Decrement by 1 at each hop, send back error at 0

**traceroute:** tool to send packets with increasing TTLs

=> can learn about network paths!
Traceroute

• When TTL reaches 0, router may send back an error
  – ICMP TTL exceeded

• If it does, we can identify a path used by a packet!
Traceroute example

[deemer@Warsprite ~]$ traceroute -q 1 google.com
traceroute to google.com (142.251.40.174), 30 hops max, 60 byte packets
1 router1-nac.linode.com (207.99.1.13) 0.621 ms
2 if-0-1-0-0-0.gw1.cjj1.us.linode.com (173.255.239.26) 0.499 ms
3 72.14.222.136 (72.14.222.136) 0.949 ms
4 72.14.222.136 (72.14.222.136) 0.919 ms
5 108.170.248.65 (108.170.248.65) 1.842 ms
6 lga25s81-in-f14.1e100.net (142.251.40.174) 1.812 ms
Traceroute example

[deemer@Warsprite ~]$ traceroute -q 1 amazon.co.uk
traceroute to amazon.co.uk (178.236.7.220), 30 hops max, 60 byte packets
  1 router2-nac.linode.com (207.99.1.14)    0.577 ms
  2 if-11-1-0-1-0.gw2.cjj1.us.linode.com (173.255.239.16)  0.461 ms
  3 ix-et-2-0-2-0.tcore3.njy-newark.as6453.net (66.198.70.104) 1.025 ms
  4 be3294.ccr41.jfk02.atlas.cogentco.com (154.54.47.217) 2.938 ms
  5 be2317.ccr41.lon13.atlas.cogentco.com (154.54.30.186) 69.725 ms
  6 be2350.rcr21.b023101-0.lon13.atlas.cogentco.com (130.117.51.138) 69.947 ms
  7 a100-row.demarc.cogentco.com (149.11.173.122) 71.639 ms
  8 150.222.15.28 (150.222.15.28)   78.217 ms
  9 150.222.15.21 (150.222.15.21)   84.383 ms
 10  *
 11 150.222.15.4 (150.222.15.4)   74.529 ms
     ...                              