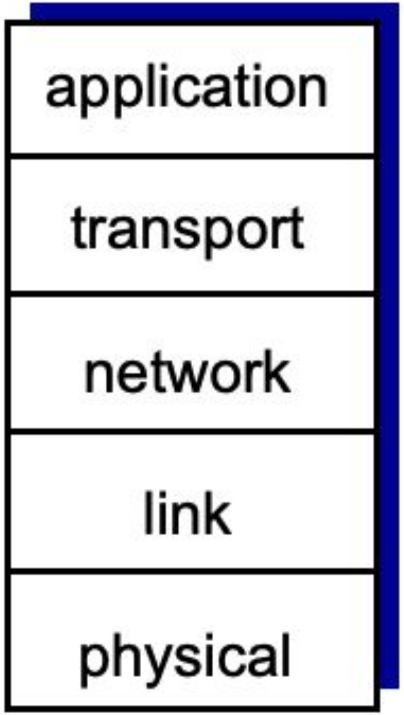
Network Layer

CS5700 Fall 2019

Where we are?

- Do you remember the responsibility of each layer?



Agenda

- Overview
- IP (v4, v6)
- Routing protocols
- ICMP

Overview

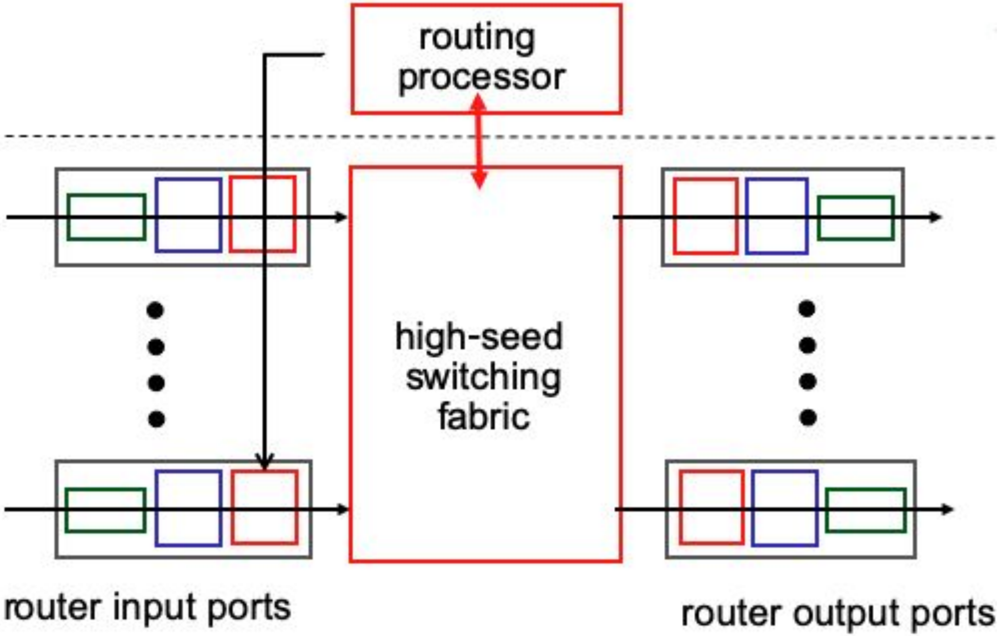
Two key network layer functions

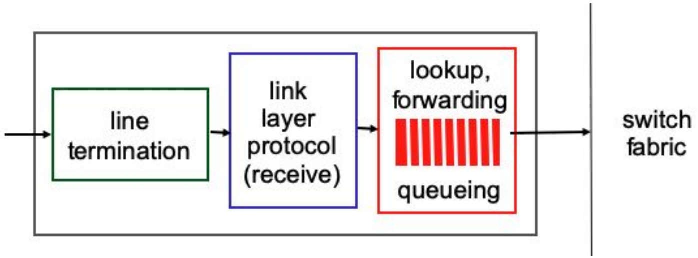
- ***Forwarding***: move packets from router's input port to appropriate output port
  - Local, per router function
- ***Routing***: determine route taken by packets from source to destination
  - Network wide logic
  - Determine how datagram is routed among routers along end-to-end path from source to destination

Network service model

- Reliability?
- Order?
- Delay?
- Throughput?

Inside a router





Destination-based forwarding table

forwarding table

Destination Address Range	Link Interface
11001000 00010111 00010000 00000000 through 11001000 00010111 00010111 11111111	0
11001000 00010111 00011000 00000000 through 11001000 00010111 00011000 11111111	1
11001000 00010111 00011001 00000000 through 11001000 00010111 00011111 11111111	2
otherwise	3

Destination-based forwarding table

Destination Address Range	Link interface
11001000 00010111 00010*** *********	0
11001000 00010111 00011000 *********	1
11001000 00010111 00011*** *********	2
otherwise	3

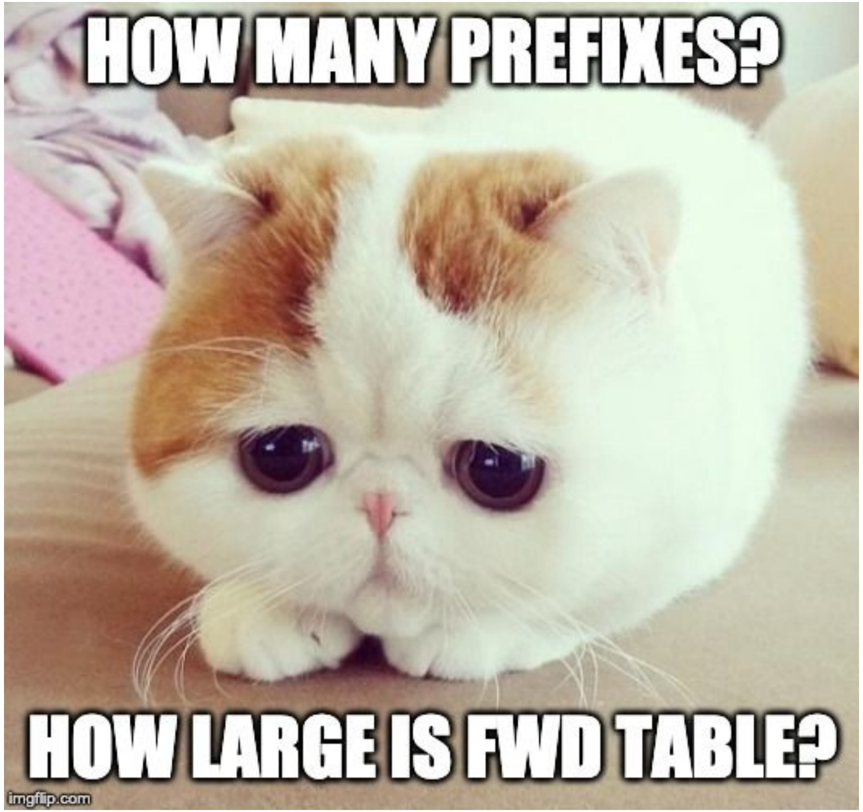
DA: 11001000 00010111 00010110 10100001 which interface?

DA: 11001000 00010111 00011000 10101010 which interface?

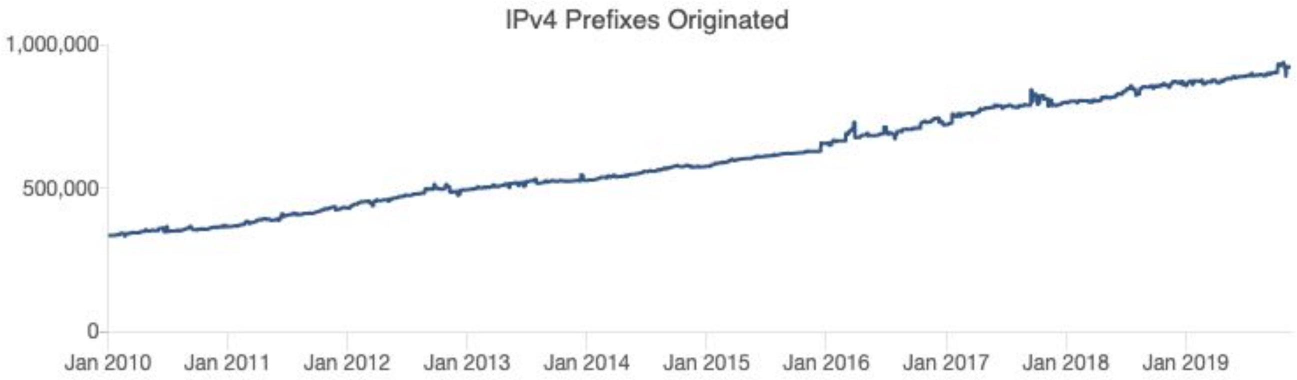
Longest prefix matching

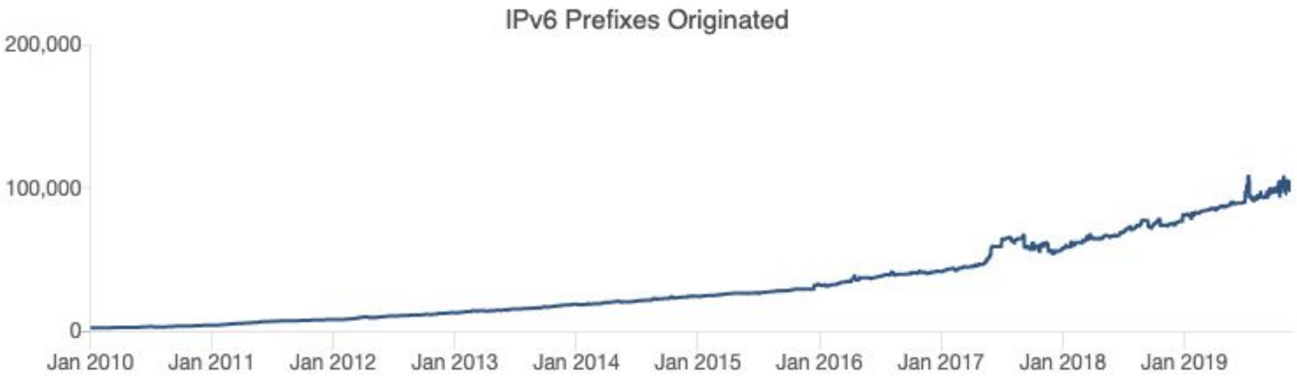
- When looking for forwarding table entry for a given destination address, use the longest address prefix that matches destination address.

Question!



How many prefixes are there?





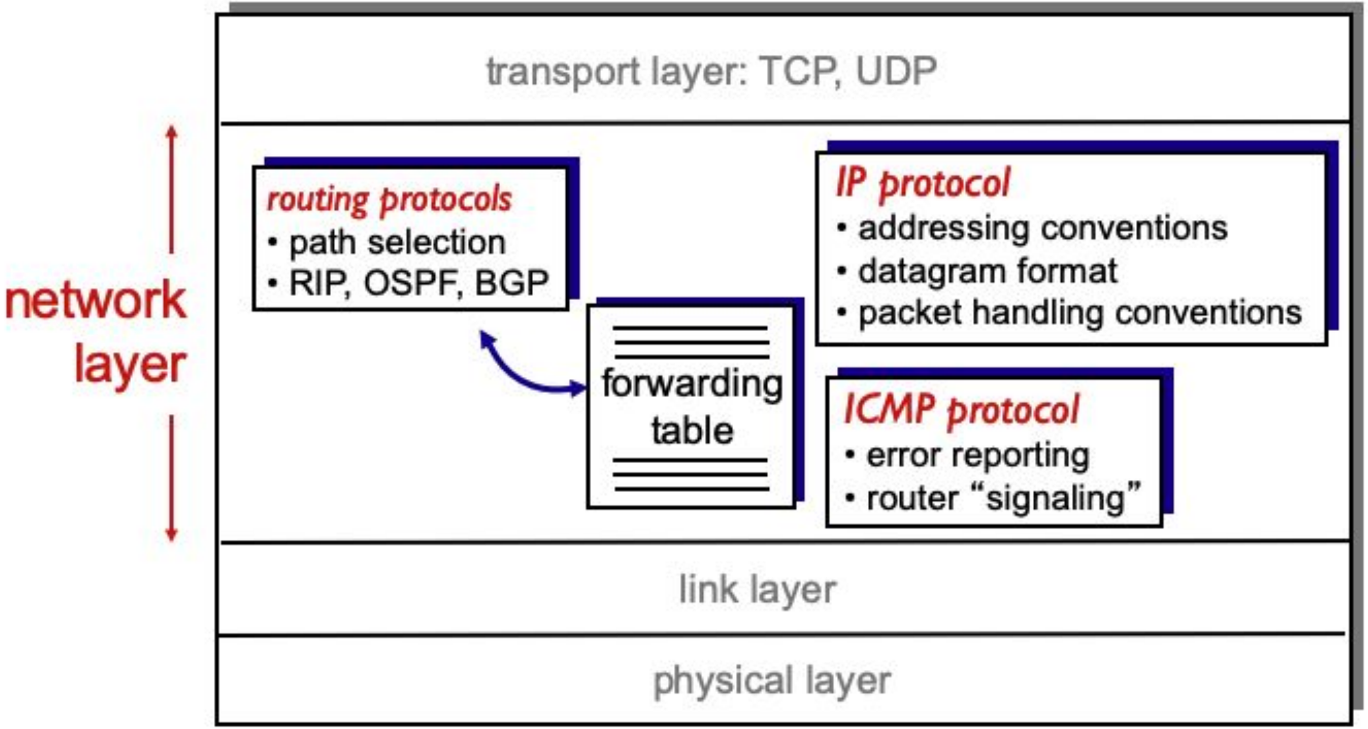
How large is the forwarding table?

```
rviews@route-server.ip.att.net> show route summary
Autonomous system number: 65000
Router ID: 12.0.1.28

inet.0: 763336 destinations, 12211712 routes (763336 active, 0 holddown, 0 hidden)
              Direct:        1 routes,         1 active
               Local:        1 routes,         1 active
                 BGP: 12211603 routes, 763227 active
              Static:      107 routes,       107 active

inet6.0: 73069 destinations, 1168998 routes (73069 active, 0 holddown, 0 hidden)
              Direct:        1 routes,         1 active
               Local:        2 routes,         2 active
                 BGP: 1168992 routes,   73063 active
              Static:        2 routes,         2 active
               INET6:        1 routes,         1 active
```

Forwarding table example

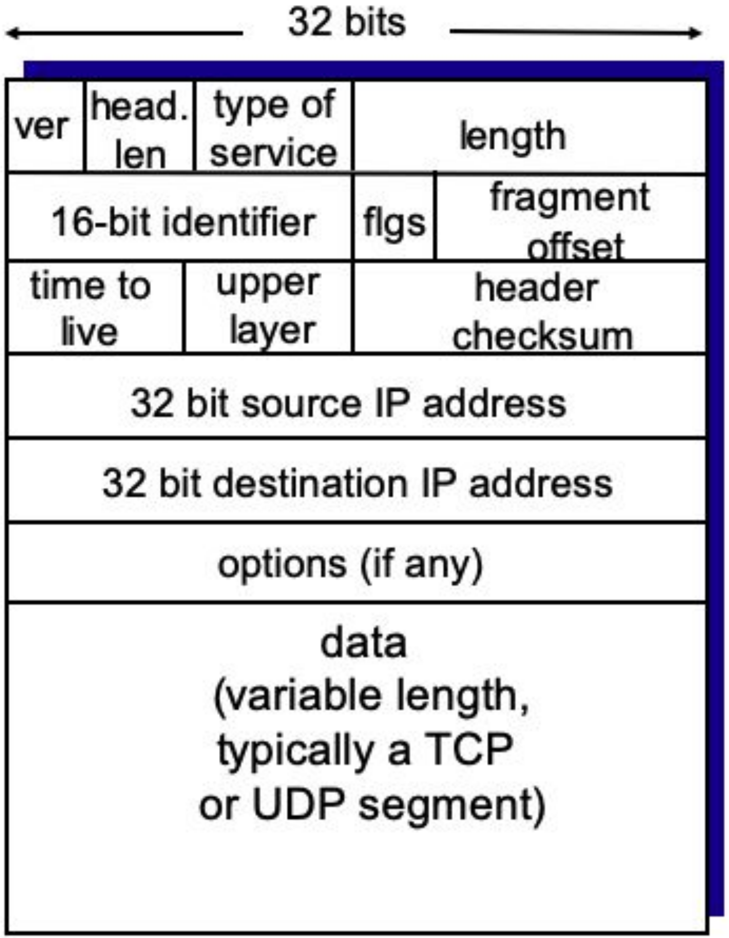
```
rviews@route-server.ip.att.net> show route forwarding-table
Routing table: default.inet
Internet:
Enabled protocols: Bridging,
Destination        Type RtRef Next hop           Type Index    NhRef Netif
default            user     7 0:0:5e:0:1:1       ucst      578    32 em0.0
default            perm     0                    rjct       36     1
0.0.0.0/32         perm     0                    dscd       34   106
1.93.23.118/32     user     0                    dscd       34   106
12.0.0.0/8         user     0                    indr  1048574   300
                              0:0:5e:0:1:1       ucst      578    32 em0.0
12.0.0.0/9         user     0                    indr  1048574   300
                              0:0:5e:0:1:1       ucst      578    32 em0.0
12.0.1.0/24        intf     0                    rslv      565     1 em0.0
12.0.1.0/32        dest     0 12.0.1.0           recv      563     1 em0.0
12.0.1.1/32        dest     0 0:0:5e:0:1:1       ucst      578    32 em0.0
12.0.1.25/32       dest     1 0:6:5b:f0:d6:44    ucst      584     2 em0.0
12.0.1.28/32       intf     0 12.0.1.28          locl      564     2
12.0.1.28/32       dest     0 12.0.1.28          locl      564     2
12.0.1.62/32       dest     0 0:13:80:d1:64:40   ucst      593     1 em0.0
12.0.1.71/32       dest     1 0:21:5e:c8:a4:78   ucst      580     2 em0.0
12.0.1.139/32      dest     1 0:5:85:ce:1d:f4    ucst      579     2 em0.0
12.0.1.148/32      dest     1 52:54:0:42:2c:ed   ucst      585     2 em0.0
12.0.1.160/32      dest     1 52:54:0:1:8e:4     ucst      586     2 em0.0
```

Network layer



IP (v4 and v6)

IPv4 datagram format

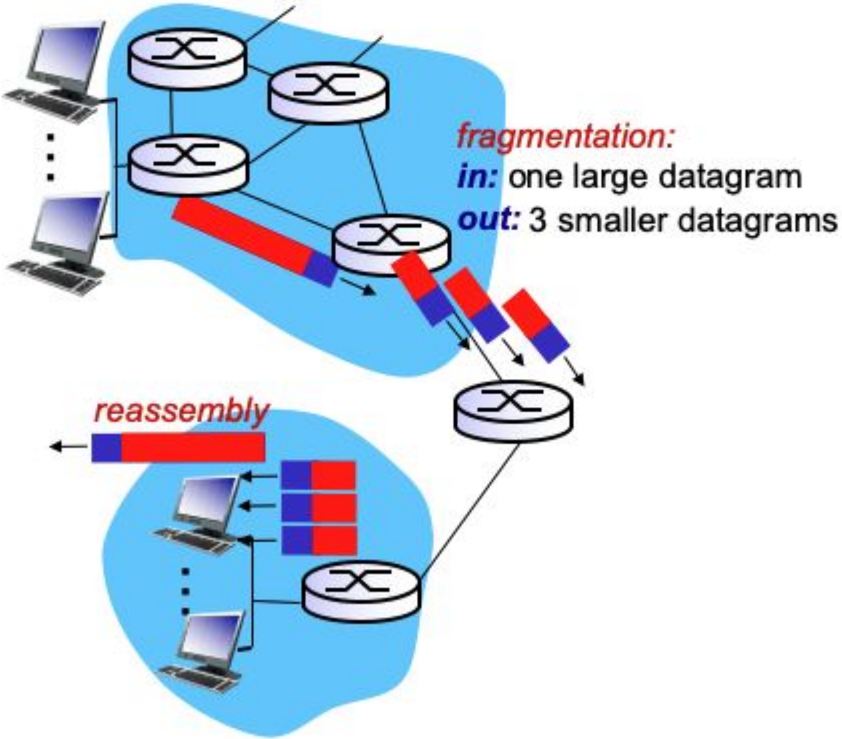


how much overhead?

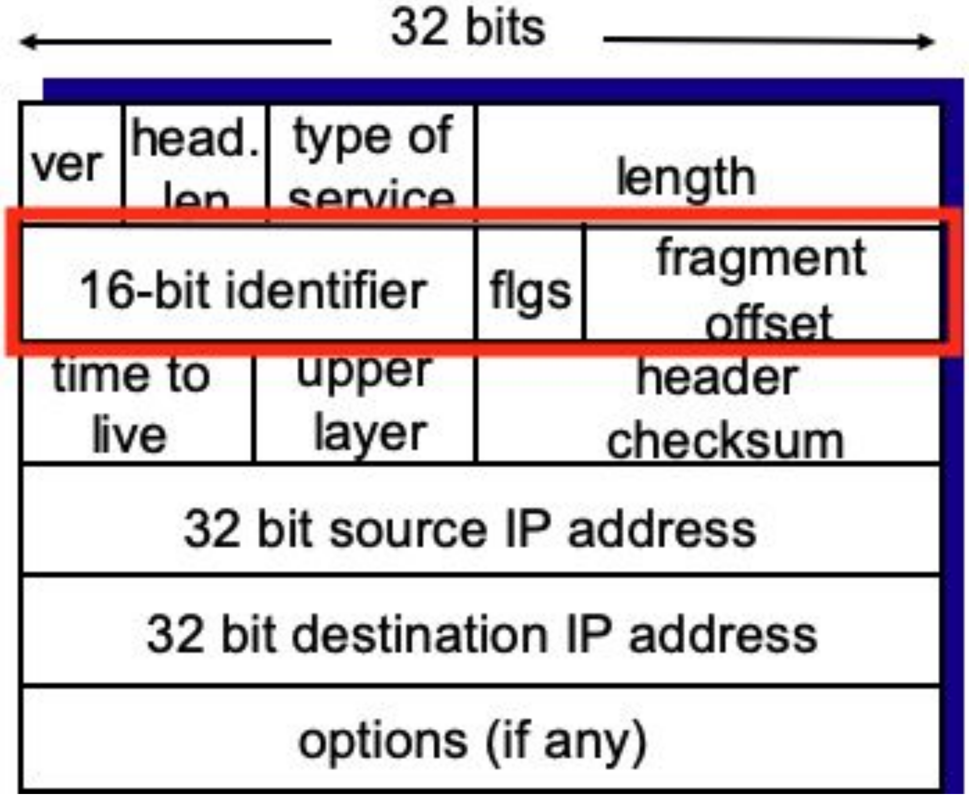
- 20 bytes of TCP
- 20 bytes of IP
- = 40 bytes + app layer overhead

IP fragmentation, reassembly

- Network links have MTU (max transfer size), aka largest possible link level frame
- Large IP datagram will be “fragmented”
- IP header bits used to identify, order related fragments



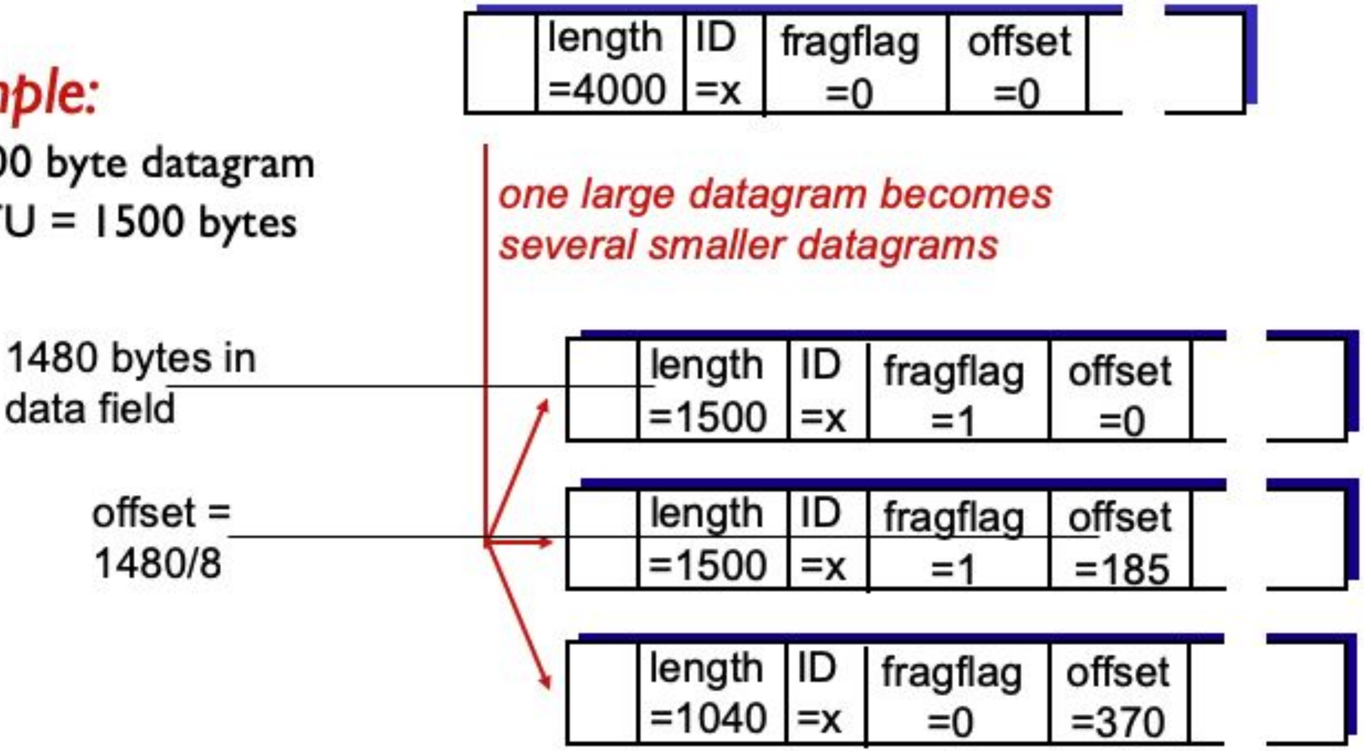
IP fragmentation, assembly



IP fragmentation, assembly

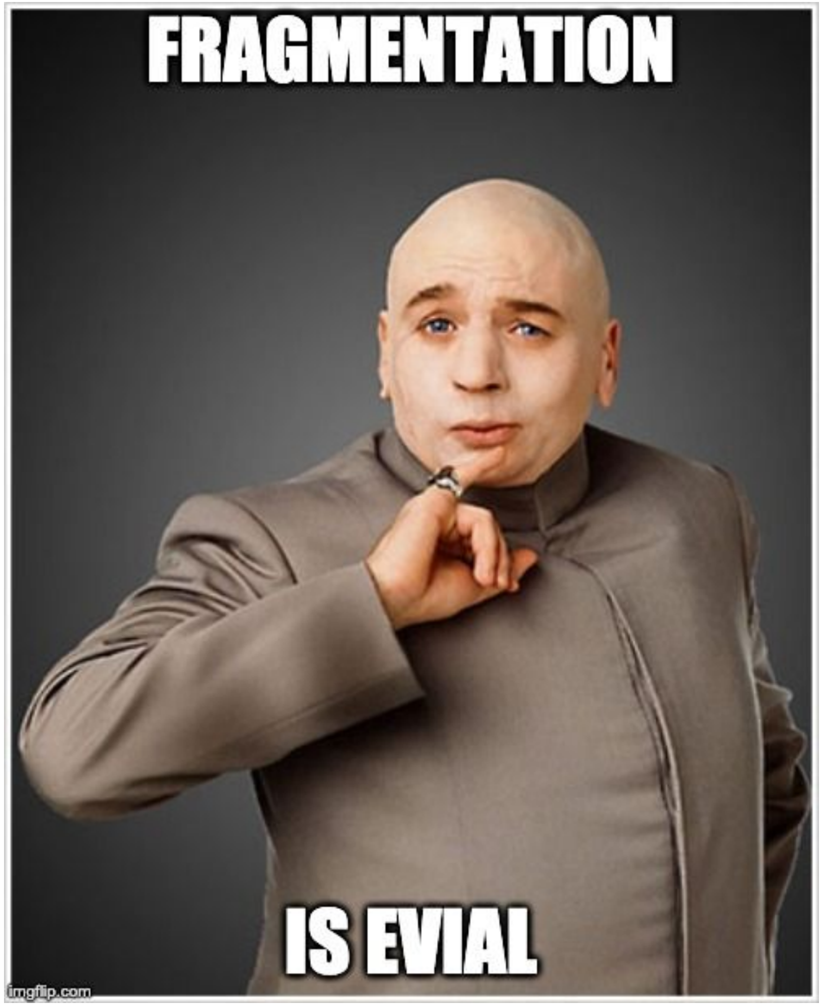
example:

- 4000 byte datagram
- MTU = 1500 bytes

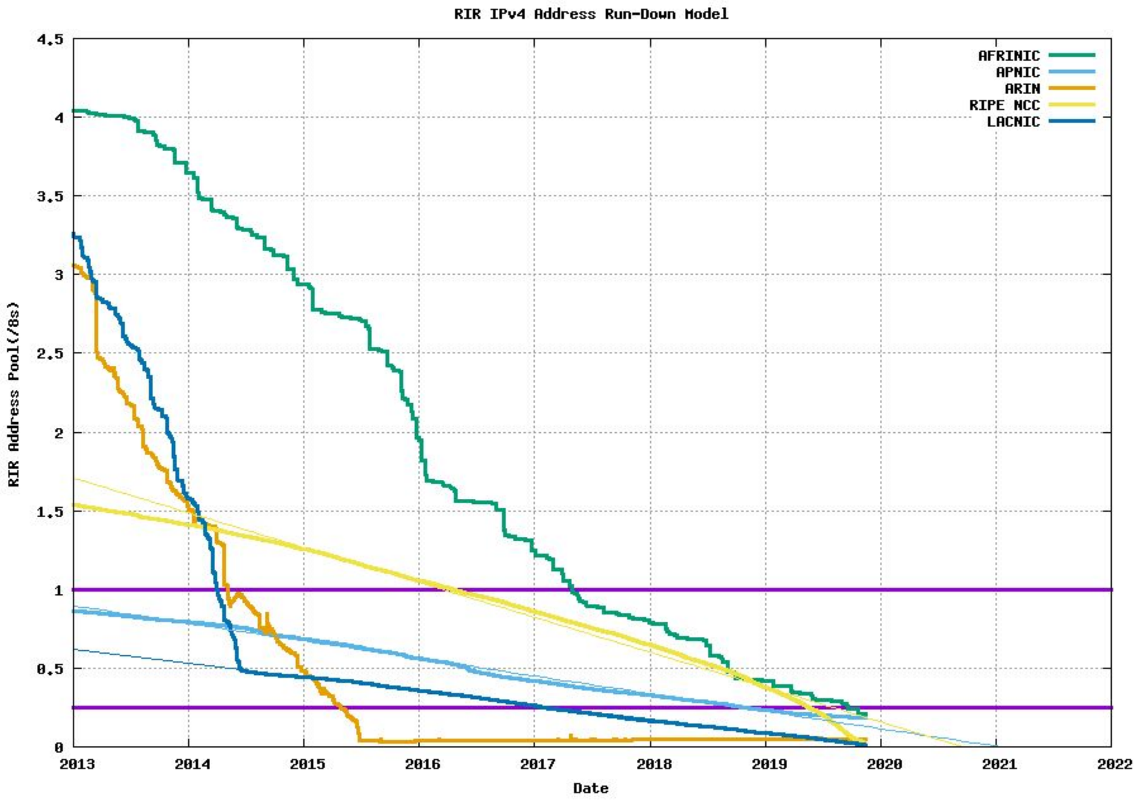


Why?

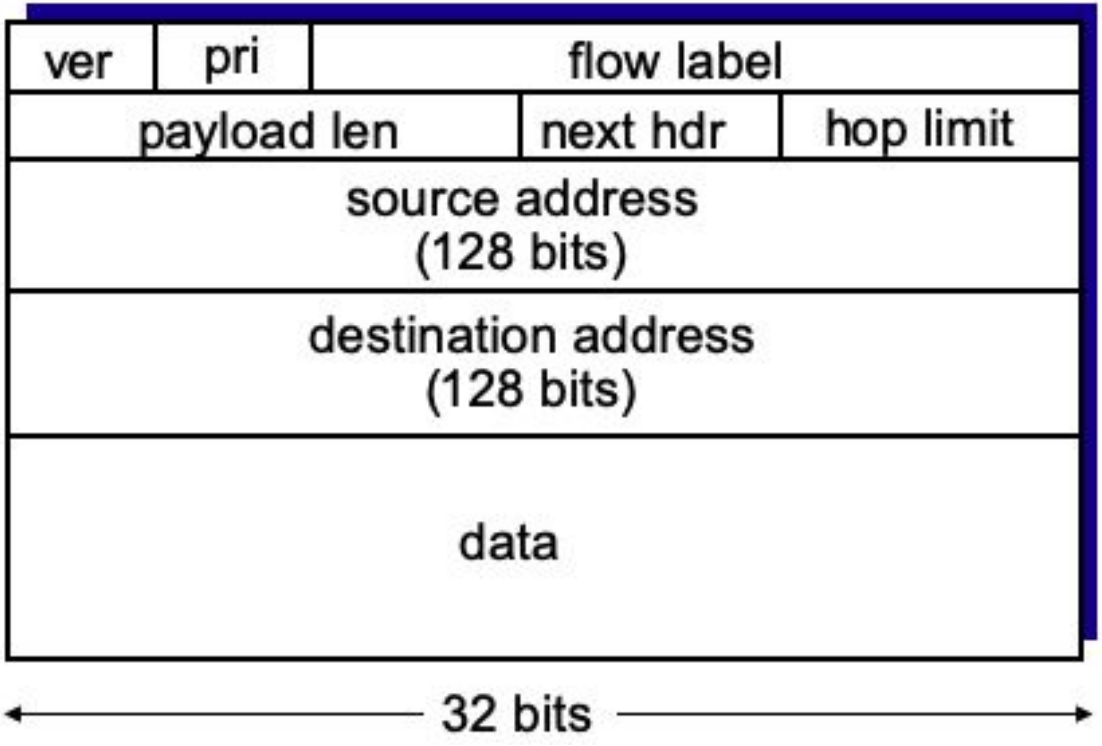
- Send packet from A to B
- Probability of a success delivery is p
- What happens in case of fragmentation?



IPv6 motivation



IPv6 datagram format



IPv6 header

- Fixed length 40 bytes
  - 32 bytes are used for source and destination IP addresses
- No checksum
  - Lower layer protocols have CRC to detect errors
- Hot limit is the same as TTL in v4

How many addresses?

- Land 148,940,000 km^2
- Water 361,132,000 km^2
- Total 510,072,000 km^2
- Number of IPv6 addresses 2^{128}
- 66,712,614,478,140,039,732,307 IP addresses per ft^2

IPv6 address notation

- 128 bits noted as eight 16-bit fields
- Separated by colons, not dots
- Each integer is represented by 4 hexadecimal digits
- E.g. 2001:0DB8:0000:0000:0000:0000:3257:9652

IPv6 shorthand

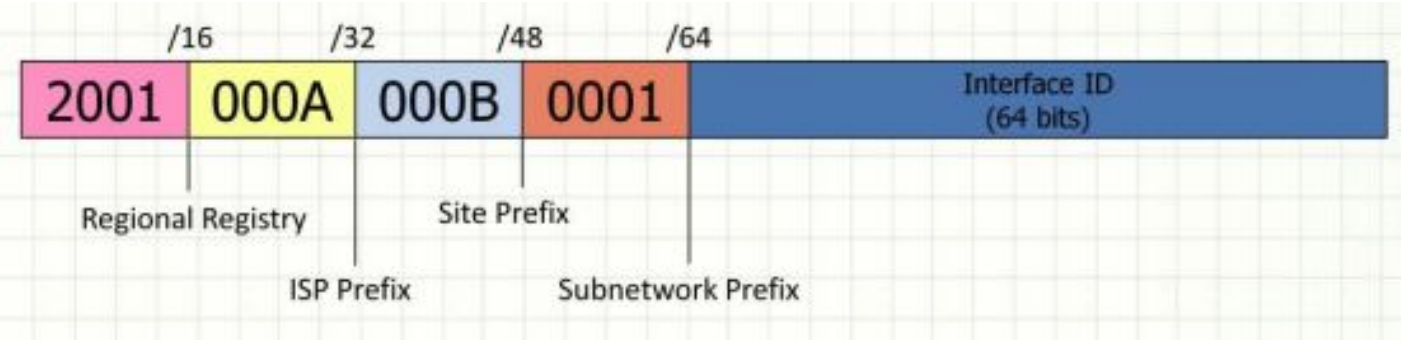
- It is likely that at first there may be many many zero's in the address.
- FF01:0000:0000:0000:0000:0000:0000:0001
- FF01:0:0:0:0:0:0:1
- FF01:0:0::1
- FF01::1
- 0:0:0:0:0:0:0:1 = ::1
- 0:0:0:0:0:0:0:0 = ::

IPv6 network notation

- IPv6 network address are denoted by CIDR notation
- The initial bits of IPv6 address form the network prefix
- E.g. 2001:CDBA:9ABC:5678::/64
  - 2001:CDBA:9ABC:5678:: to 2001:CDBA:9ABC:5678::FFFF:FFFF:FFFF:FFFF

IPv6 network notation

- Network prefix 64 bits, host identifier 64 bits
- ISP allocates you /48
- 2^{16} gives 65536 subnets
- You allocate a /64 to each interface



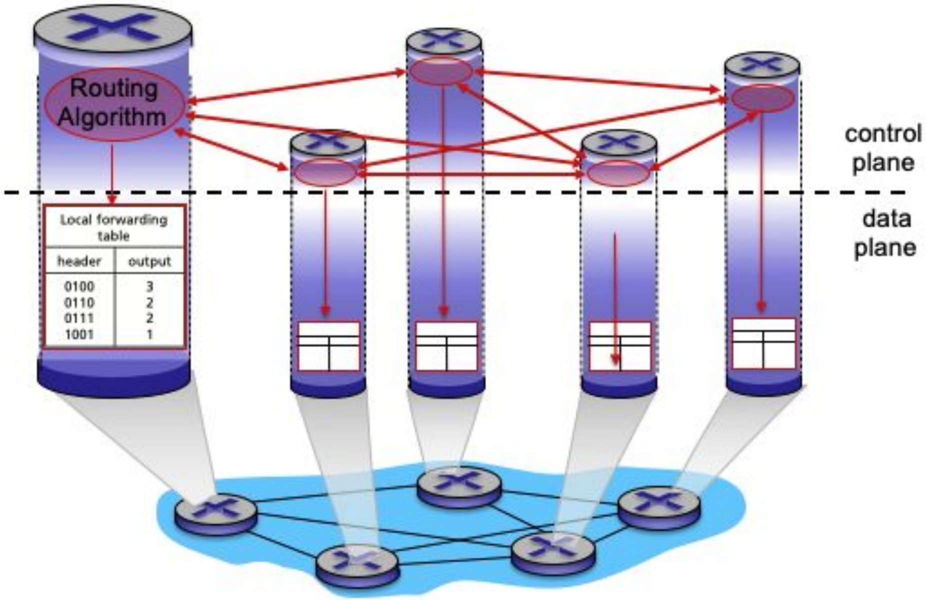
Routing protocols

Routing

- Determine route taken by packets from source to destination
- Two approaches
  - per-router control (traditional)
  - Logically centralized control (software defined networking, aka SDN)

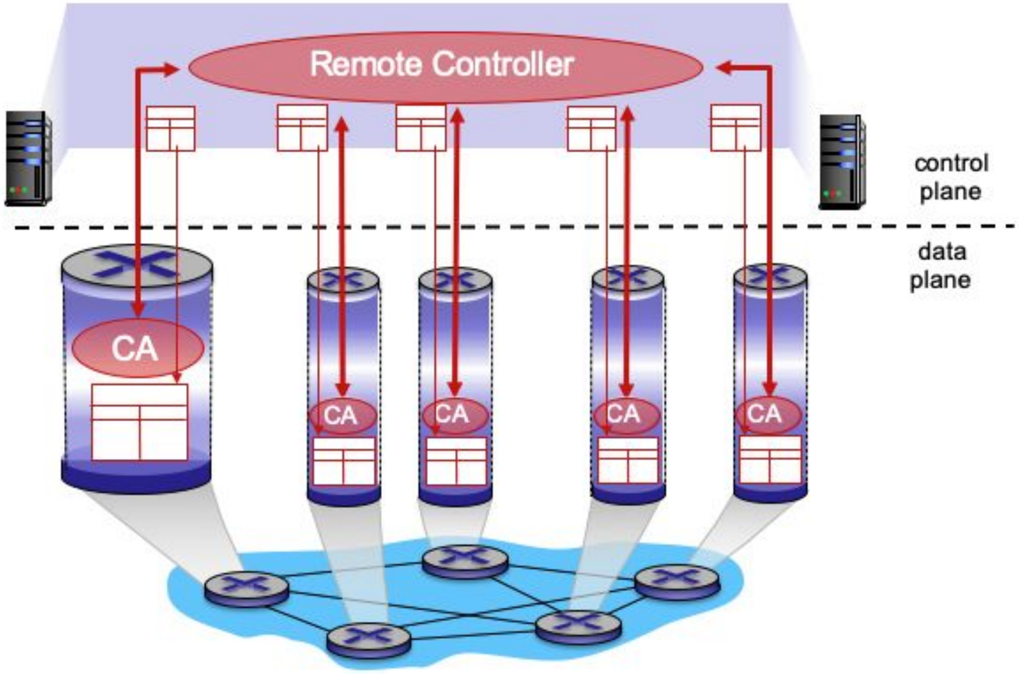
Per-router control

Individual routing algorithm components in each and every router, interact with each other to compute forwarding tables.



Logically centralized control

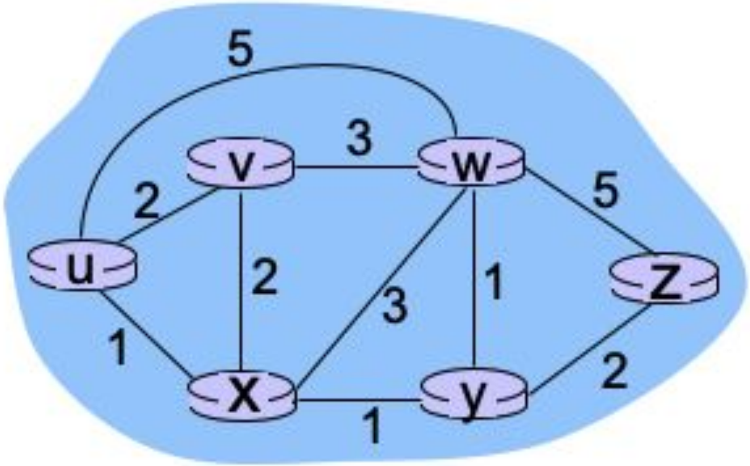
A distinct remote controller interacts with local control agents (CAs) in routers to compute forwarding tables



Goal of routing protocols

- Determine “good” paths from source to destination
- Path is a sequence of routers packets will traverse

Graph abstraction of the network

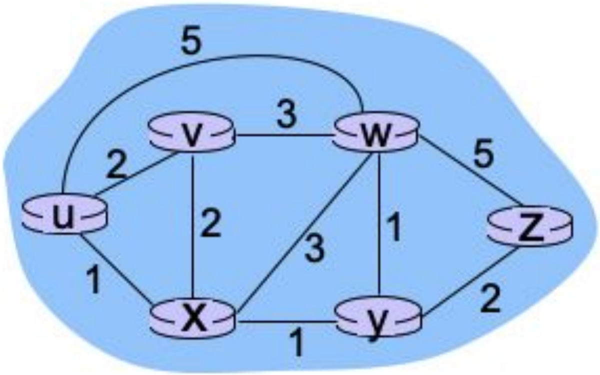


graph: G = (N,E)

N = set of routers = { u, v, w, x, y, z }

E = set of links ={ (u,v), (u,x), (v,x), (v,w), (x,w), (x,y), (w,y), (w,z), (y,z) }

Graph abstraction of the network



$c(x,x') =$ cost of link (x,x')
e.g., $c(w,z) = 5$

cost could always be 1, or inversely related to bandwidth, or inversely related to congestion

cost of path $(x_1, x_2, x_3, \ldots, x_p) = c(x_1,x_2) + c(x_2,x_3) + \ldots + c(x_{p-1},x_p)$

key question: what is the least-cost path between u and z ?
routing algorithm: algorithm that finds that least cost path

Routing algorithm classification

- Global
    - All routers have complete topology, link cost info
    - **“Link state” algorithm**
- Decentralized
    - Router only knows physically connected neighbors, link costs to neighbors
    - Iterative process of computation and exchange info
    - **“Distance vector” algorithm**

Link-state routing algorithm

- Use Dijkstra's algorithm
- Network topology, link cost known to all nodes
  - via link-state broadcast, all nodes have same info
- Compute least cost paths from one to all other nodes
  - produce forwarding table for that node

Distance vector algorithm

- Distributed version of Bellman-Ford

let

$d_x(y) :=$ cost of least-cost path from x to y

then

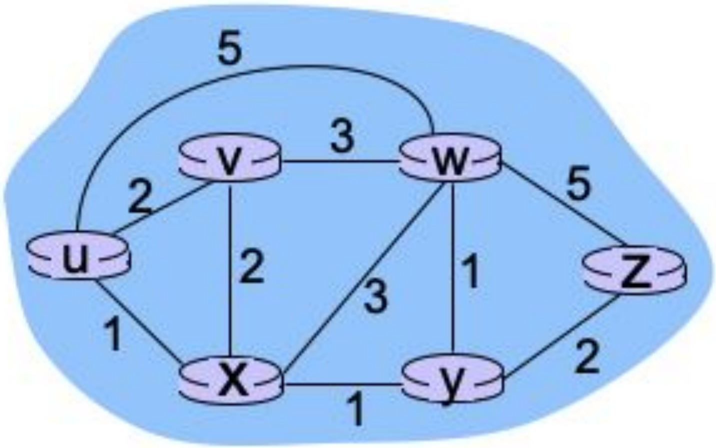
$$d_x(y) = \min_v \{c(x,v) + d_v(y)\}$$

cost from neighbor v to destination y

cost to neighbor v

min taken over all neighbors v of x

Distance vector algorithm



clearly, $d_v(z) = 5$, $d_x(z) = 3$, $d_w(z) = 3$

$$
\begin{aligned}
d_u(z) = \min \{ \; & c(u,v) + d_v(z), \\
& c(u,x) + d_x(z), \\
& c(u,w) + d_w(z) \; \} \\
= \min \{ & 2 + 5, \\
& 1 + 3, \\
& 5 + 3 \} \; = 4
\end{aligned}
$$

Distance vector algorithm

- $D_x(y)$ = estimate of least cost from x to y
- x maintains distance vector $D_x = [D_x(y) \text{ for } y \text{ in } N]$
- Node x:
  - knows cost to each neighbor v which is $c(x,v)$
  - maintains its neighbors distance vectors. For each neighbor v, x maintains $D_v = [D_v(y) \text{ for all } y \text{ in } N]$

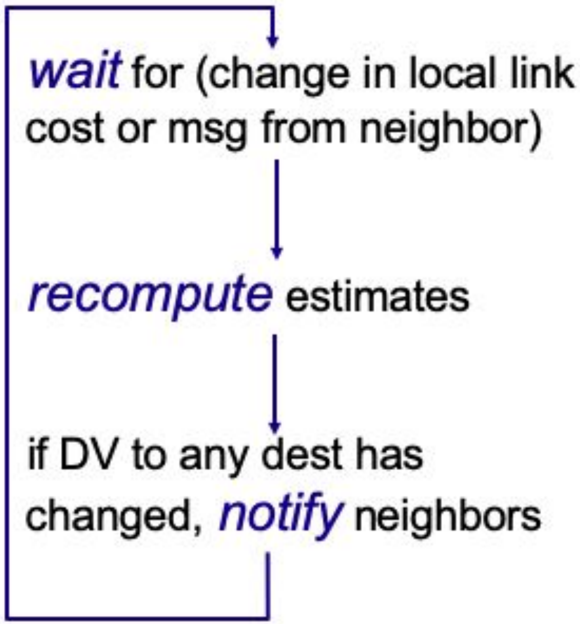
Distance vector algorithm

- Key idea
  - from time to time, each node sends its own distance vector estimate to neighbors
  - when x receives new distance vector from neighbor, it updates its own distance vector

$$D_x(y) \leftarrow \min_v\{c(x,v) + D_v(y)\} \quad \text{for each node } y \in N$$

Distance vector algorithm

- Iterative and asynchronous. Each iteration is caused by
  - local link cost change
  - distance vector update from neighbors
- Distributed
  - only notify its own neighbors



node x table

from \ cost to	x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

from \ cost to	x	y	z
x	0		
y	2	0	1
z	7	1	0

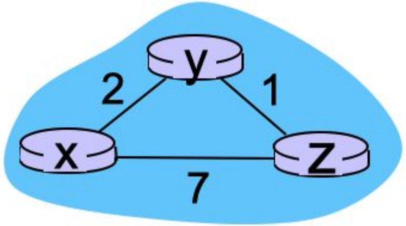
node y table

from \ cost to	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

node z table

from \ cost to	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0

time



node x table

from \ cost to	x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

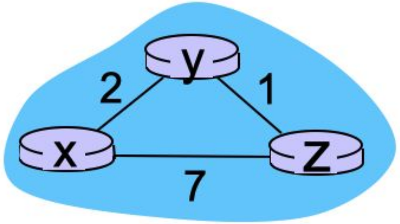
from \ cost to	x	y	z
x	0	2	
y	2	0	1
z	7	1	0

node y table

from \ cost to	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

node z table

from \ cost to	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0



time

node x table

from \ cost to	x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

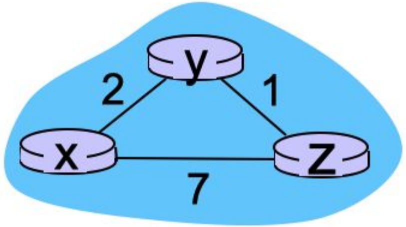
from \ cost to	x	y	z
x	0	2	3
y	2	0	1
z	7	1	0

node y table

from \ cost to	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

node z table

from \ cost to	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0



time

node x table

from \ cost to	x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

from \ cost to	x	y	z
x	0	2	3
y	2	0	1
z	7	1	0

from \ cost to	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

node y table

from \ cost to	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

from \ cost to	x	y	z
x	0	2	7
y	2	0	1
z	7	1	0

from \ cost to	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

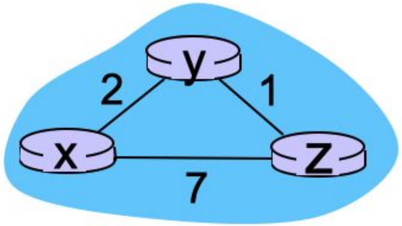
node z table

from \ cost to	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0

from \ cost to	x	y	z
x	0	2	7
y	2	0	1
z	3	1	0

from \ cost to	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

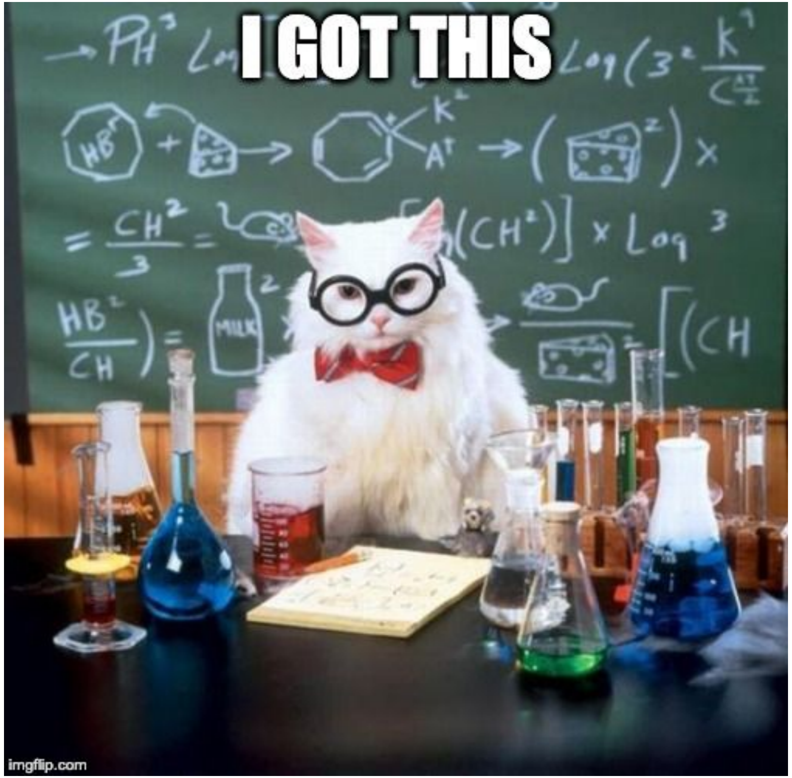
time

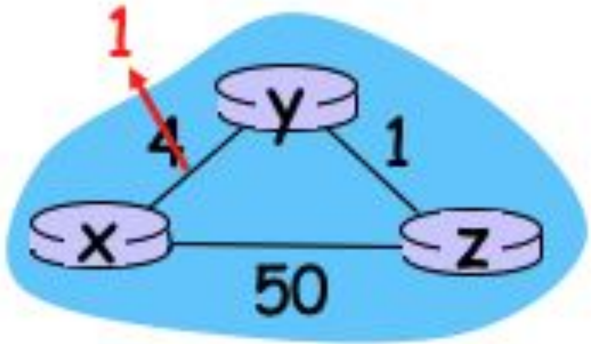


Link cost change

- Node detects local link cost change
- Update routing info, recalculates distance vector
- Notify neighbors if distance vector changes

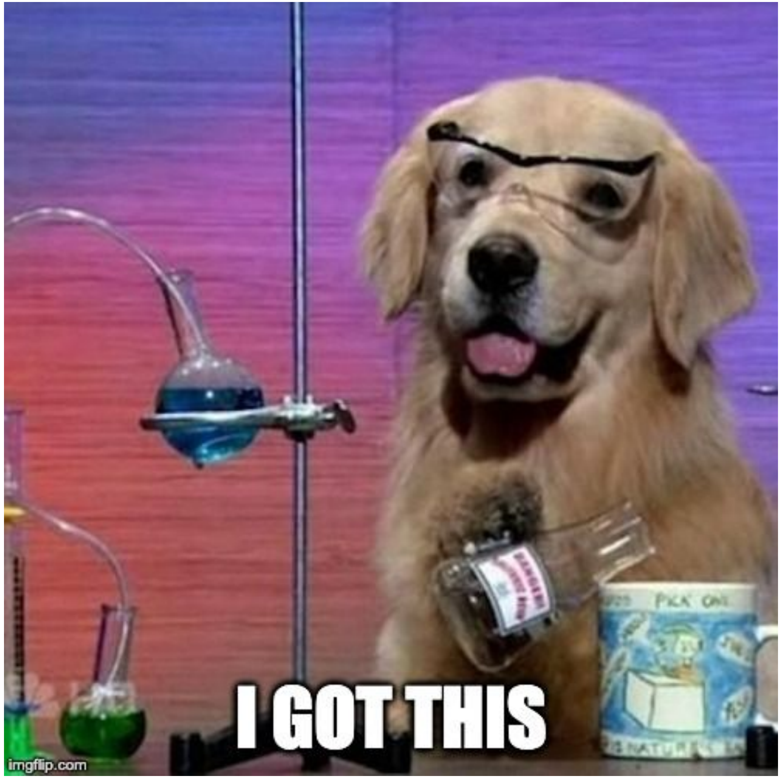
What happens when cost reduces?

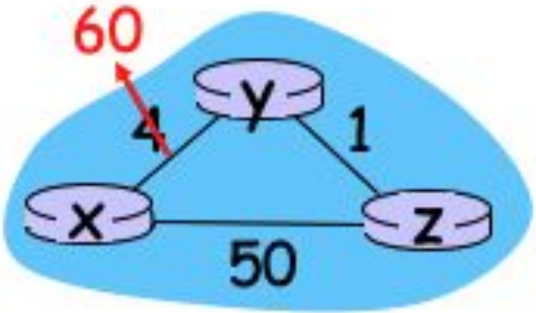




Distance vector algorithm

What happens when cost increases





Distance vector

Poisoned reverse

- If z routes through y to get to x
  - z tells y its distance to x is infinite
  - so y won't route to x via z

Comparison between LS and DV

- Message complexity
  - LS: with n nodes and E links, $O(nE)$ msg sent
  - DV: it varies
- Speed of convergence
  - LS: with n nodes and E links, $O(n^2)$
  - DV: it varies


PICK ONE ALGORITHM
APPLY TO THE WHOLE INTERNET
imgflip.com

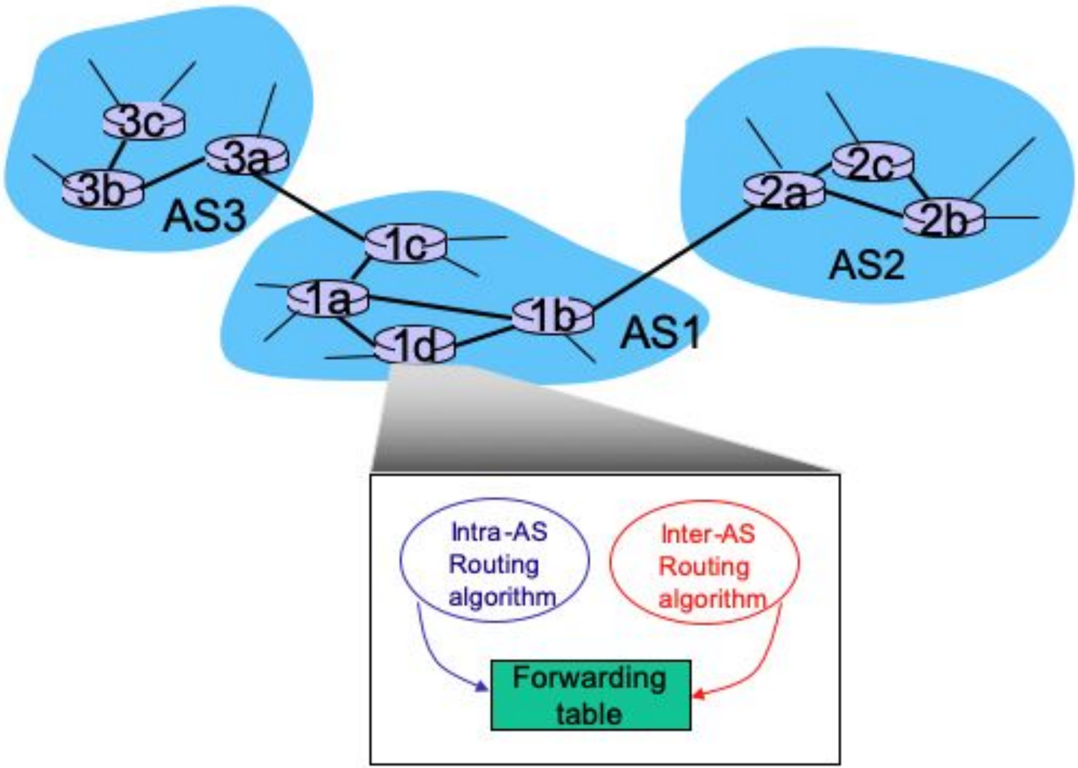
No!!!!!

- Scalability
  - neither can scale to handle the entire Internet
- Administrative autonomy
  - Internet = network of networks
  - each network admin may want to control routing in its own network

The Internet approach

- Aggregate routers into regions known as “autonomous systems” (aka AS)
- intra-AS routing
  - routing among hosts/routers in the same AS
  - routers in same AS run the same protocol
  - routers in different AS can run different protocols
- inter-AS routing
  - routing among AS’es

The Internet approach



Intra-AS routing

- Also known as Interior Gateway Protocols (IGP)
- Most common intra-AS routing protocols
  - RIP: Routing Information Protocol
    - based on distance vector
  - OSPF: Open Shortest Path First
    - based on link state
  - EIGRP: Enhanced Interior Gateway Routing Protocol
    - Proprietary protocol by Cisco

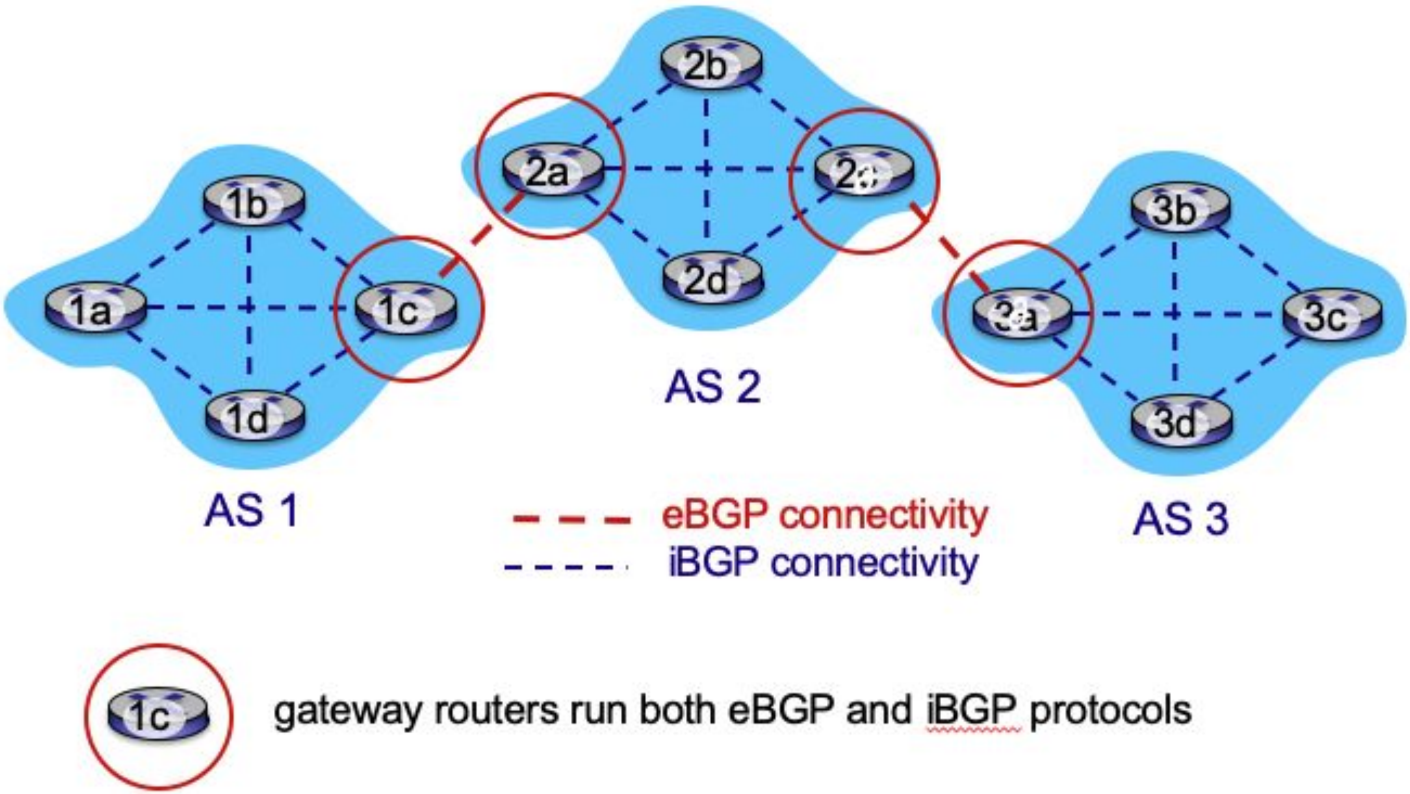
Inter-AS routing

- BGP: Border Gateway Protocol
  - the de facto inter-AS routing protocol
- No other inter-AS routing protocols!

BGP

- BGP provides each AS a means to
  - eBGP: obtain subnet reachability information from neighboring AS'es
  - iBGP: propagate reachability information to all AS-internal routers
  - Determine "good" routes to other networks based on reachability information and policy

BGP



2b
2a
2c
2d
1b
1a
1c
1d
3b
3a
3c
3d
AS 1
AS 2
AS 3
eBGP connectivity
iBGP connectivity
1c
gateway routers run both eBGP and iBGP protocols

Questions?