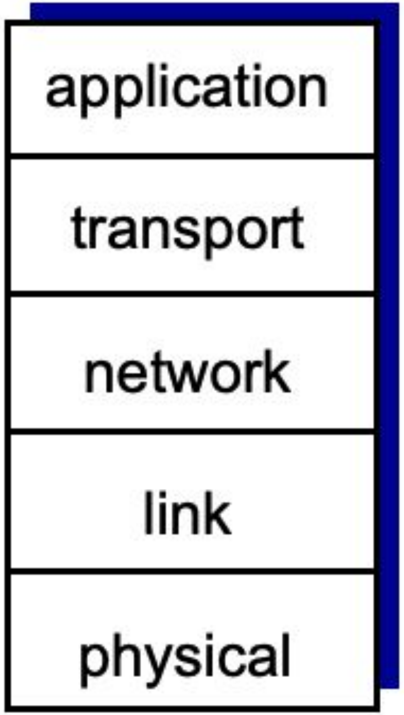
Network Layer

CS5700 Fall 2019

Where we are?

- Do you remember the responsibility of each layer?



Agenda

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

Overview

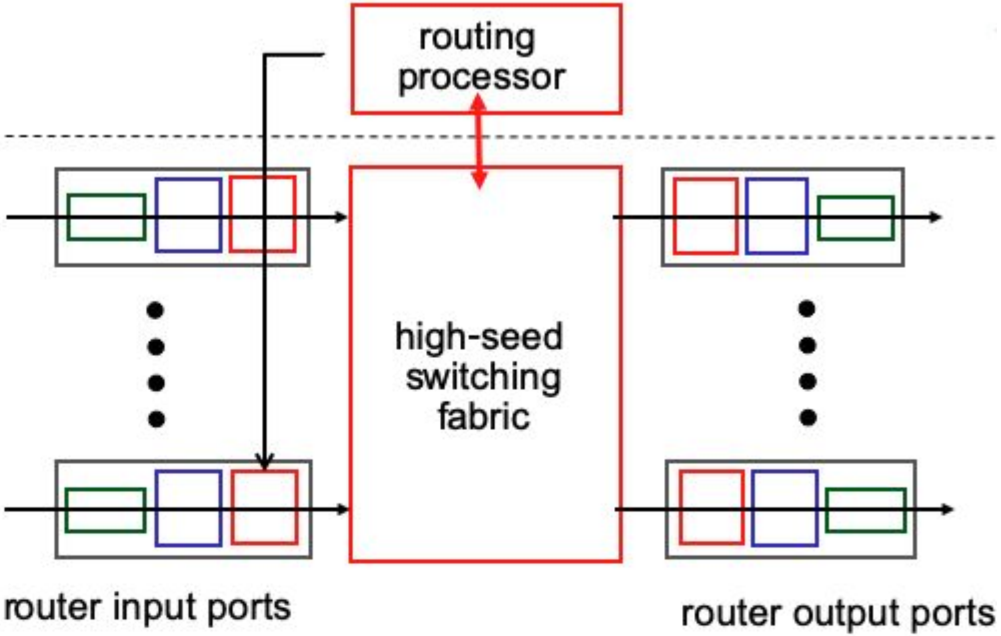
Two key network layer functions

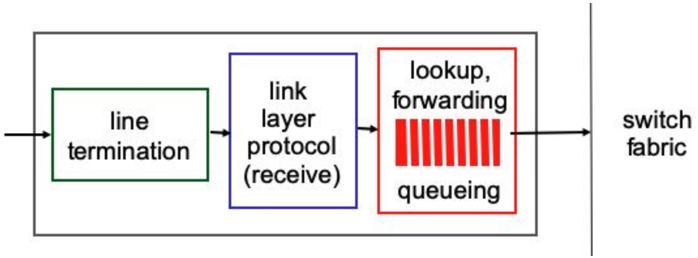
- ***Forwarding***: move packets from router's input port to appropriate output port
  - Local, per router function
- ***Routing***: determine route taken by packets from source to destination
  - Network wide logic
  - Determine how datagram is routed among routers along end-to-end path from source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

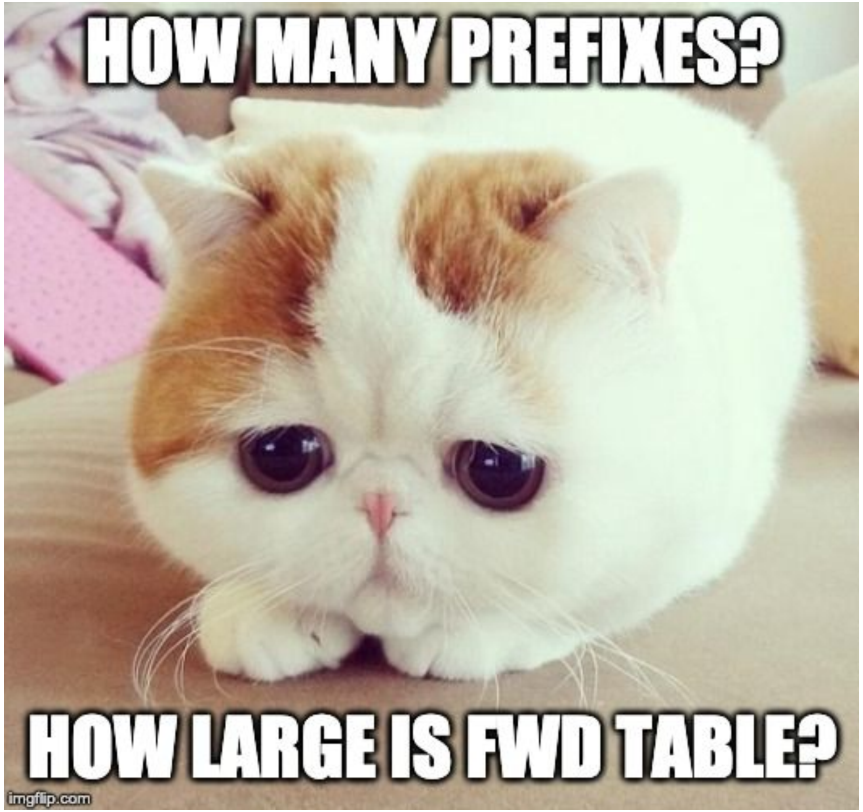
DA: 11001000 00010111 00010110 10100001 which interface?

DA: 11001000 00010111 00011000 10101010 which interface?

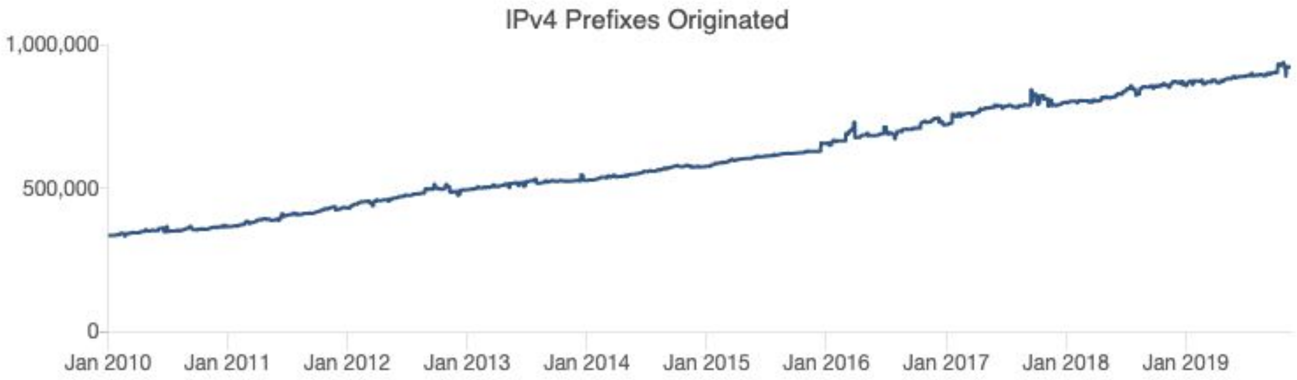
Longest prefix matching

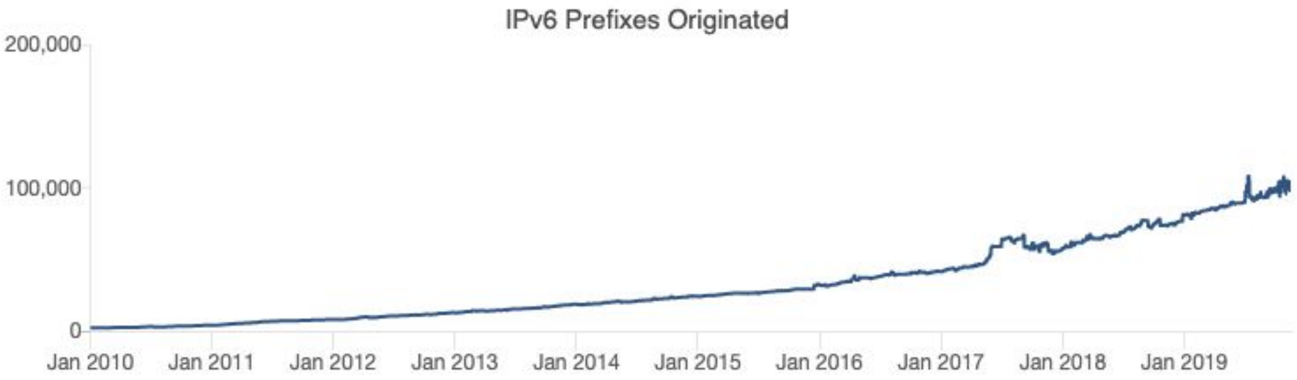
- When looking for forwarding table entry for a given destination address, use the longest address prefix that matches destination address.

Question!



How many prefixes are there?





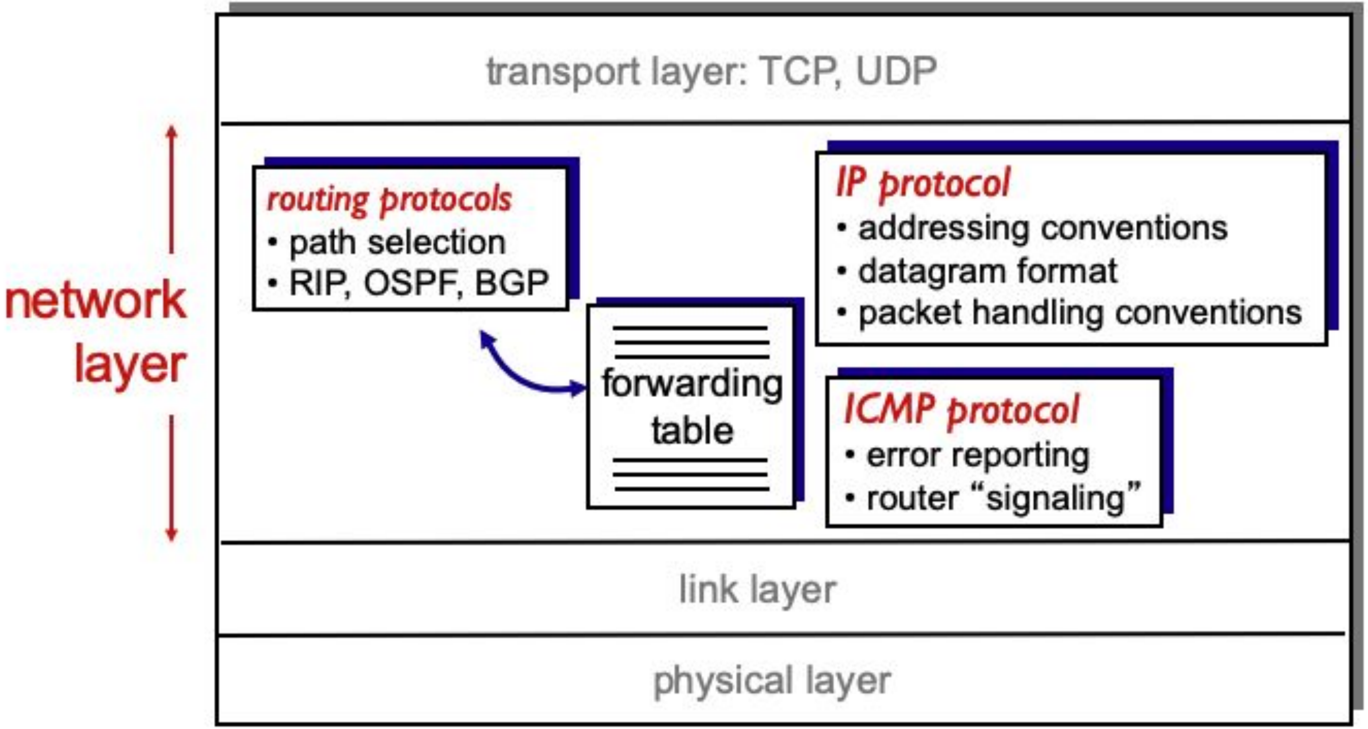
How large is the forwarding table?

```
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

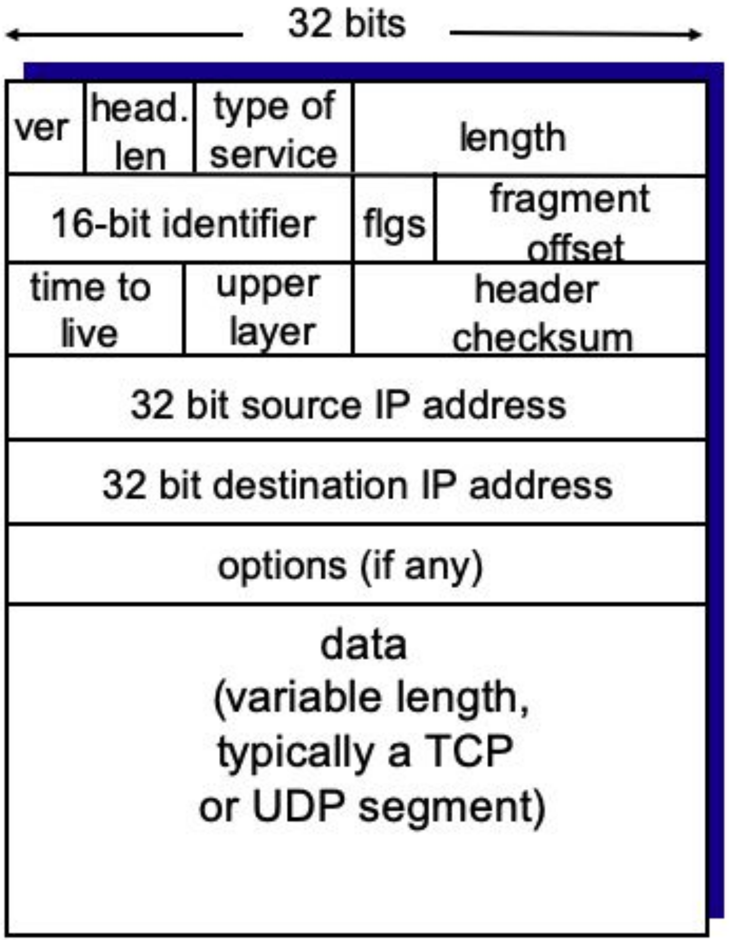
```
rviews@route-server.ip.att.net> show route forwarding-table
Routing table: default.inet
Internet:
Enabled protocols: Bridging,
Destination        Type RtRef Next hop           Type Index    NhRef Netif
default            user     7 0:0:5e:0:1:1       ucst      578    32 em0.0
default            perm     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

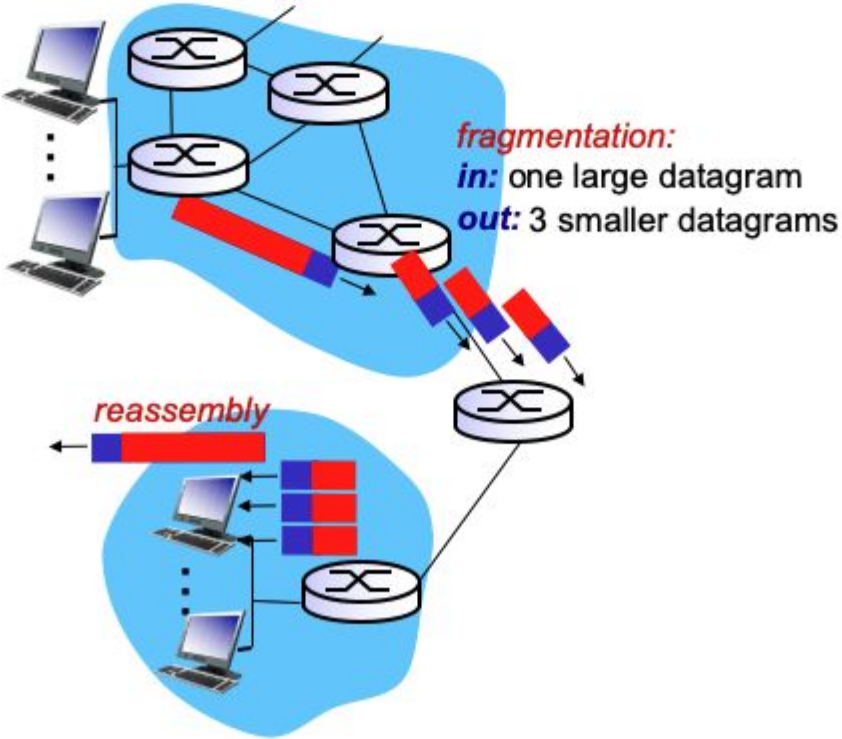


how much overhead?

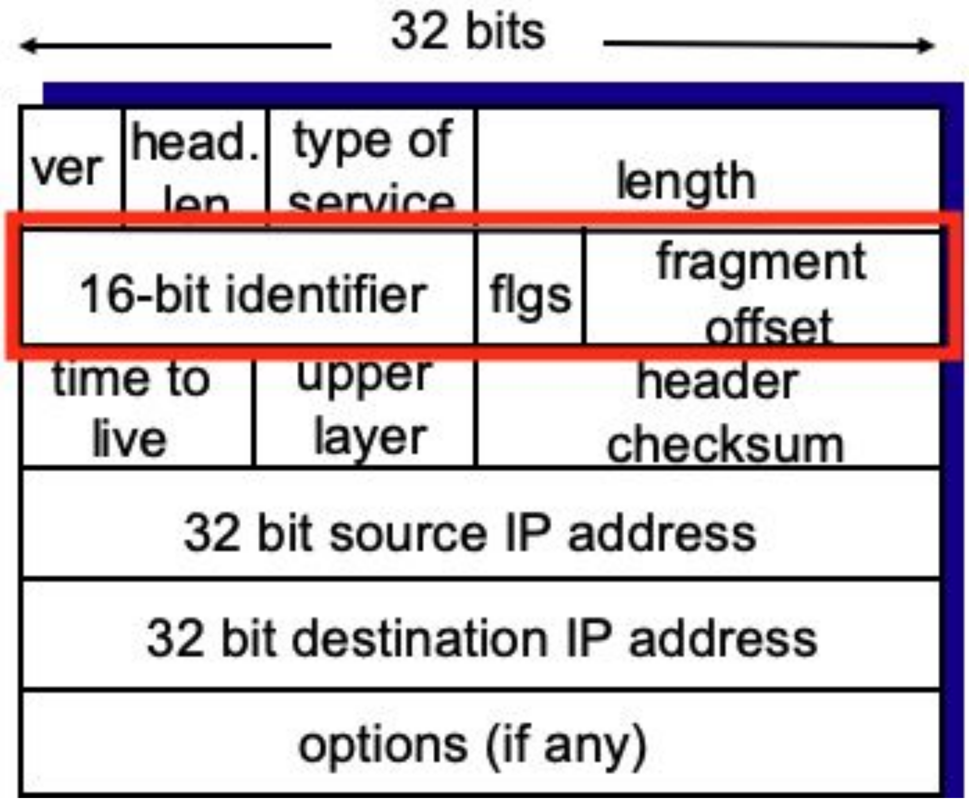
- 20 bytes of TCP
- 20 bytes of IP
- = 40 bytes + app layer overhead

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

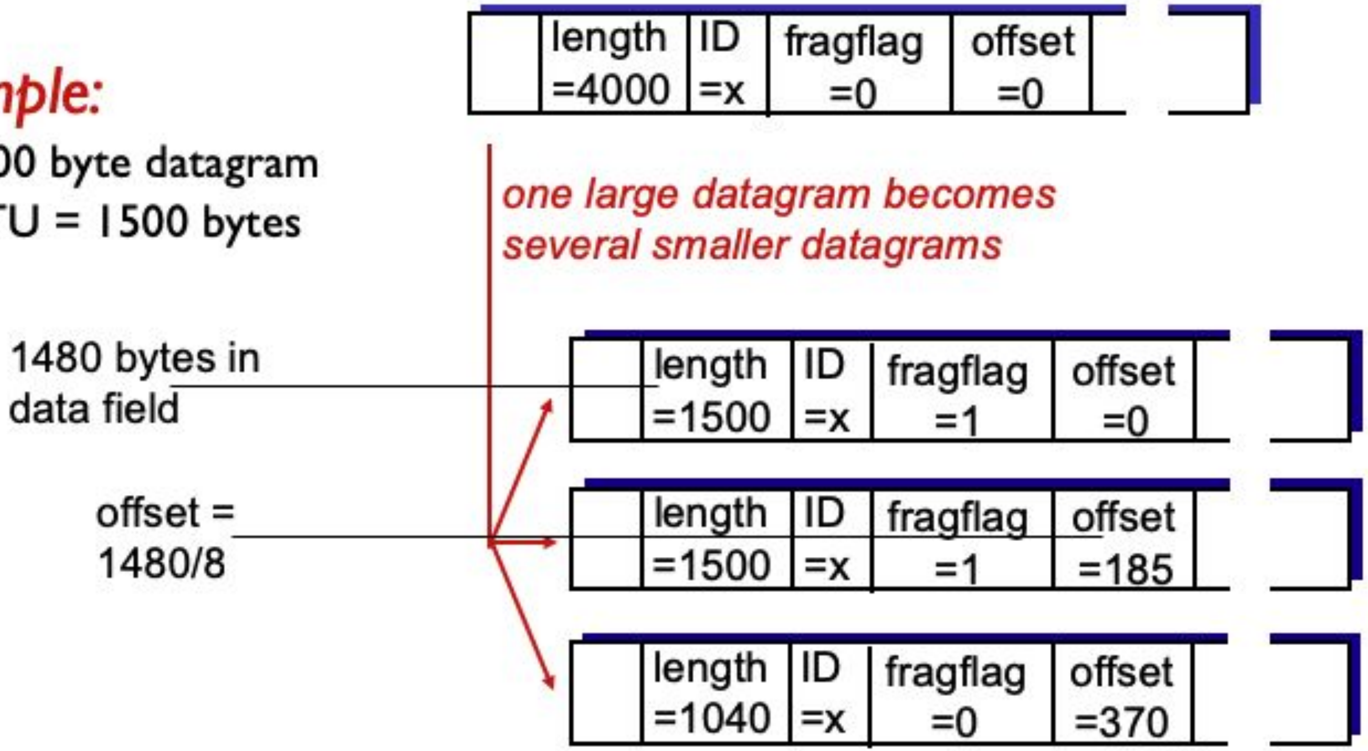


IP fragmentation, assembly



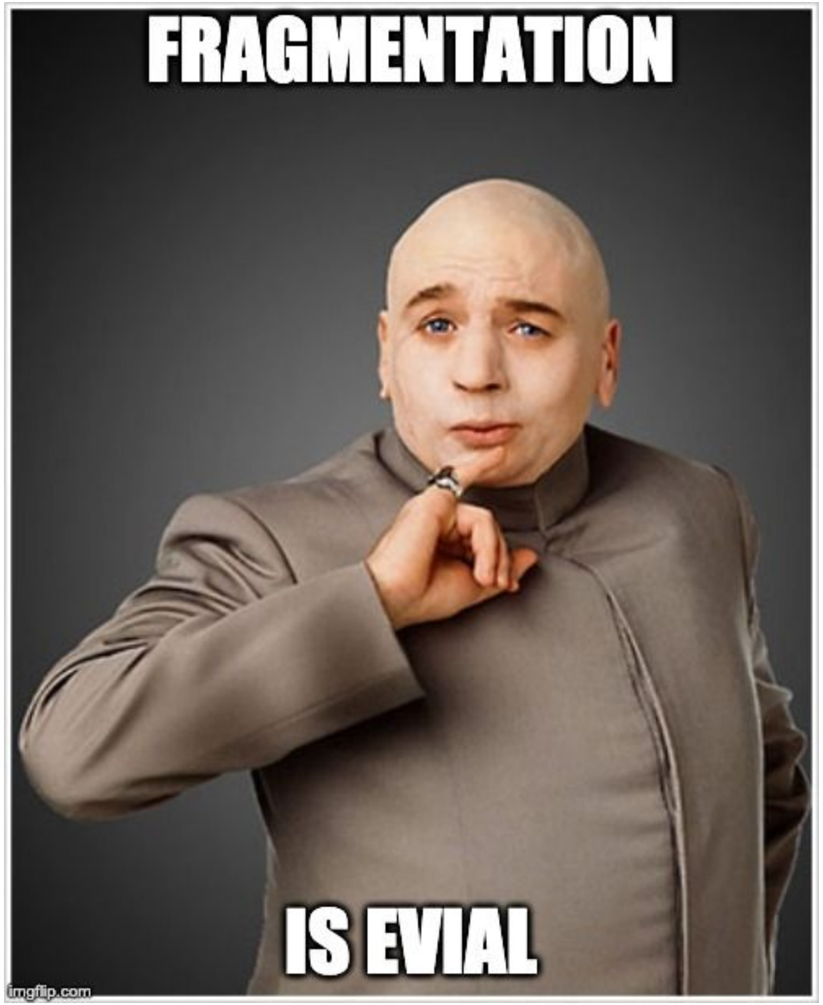
IP fragmentation, assembly

example:
- 4000 byte datagram
- MTU = 1500 bytes

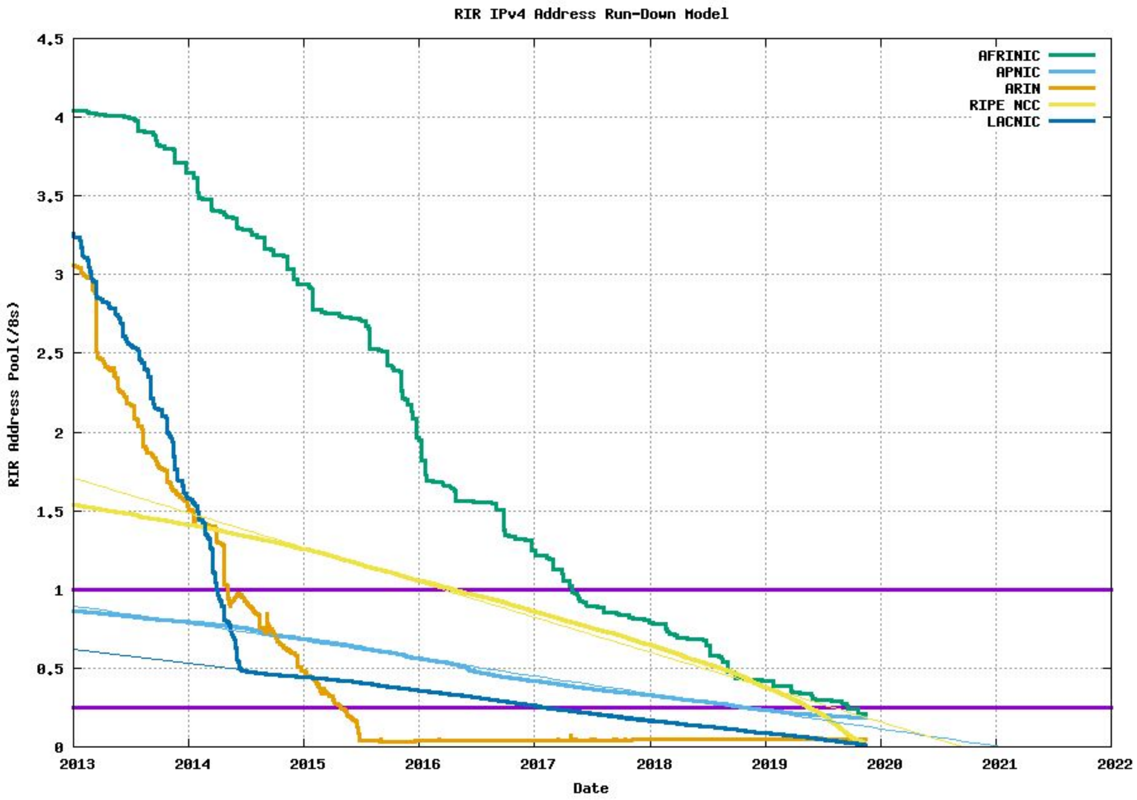


Why?

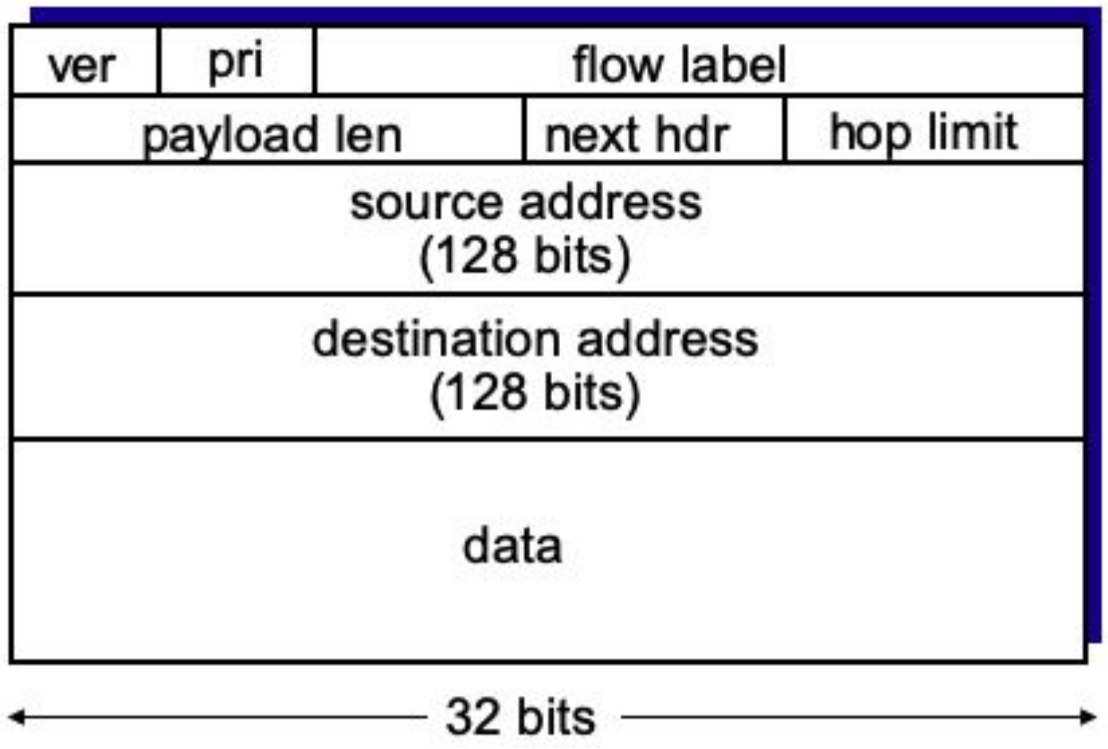
- Send packet from A to B
- Probability of a success delivery is p
- What happens in case of fragmentation?



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

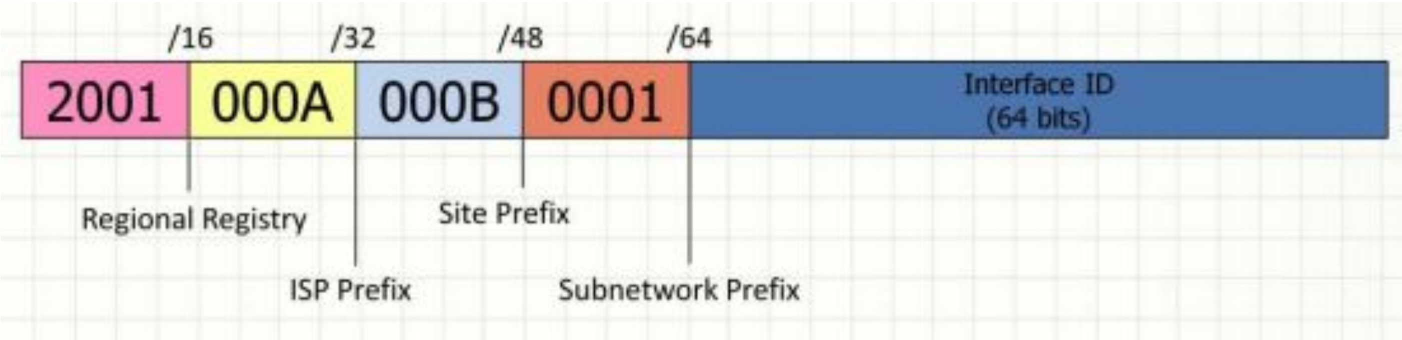
- It is likely that at first there may be many many zero’s in the address.
- FF01:0000:0000:0000:0000:0000:0000:0001
- FF01:0:0:0:0:0:0:1
- FF01:0:0::1
- FF01::1
- 0:0:0:0:0:0:0:1 = ::1
- 0:0:0:0:0:0:0:0 = ::

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



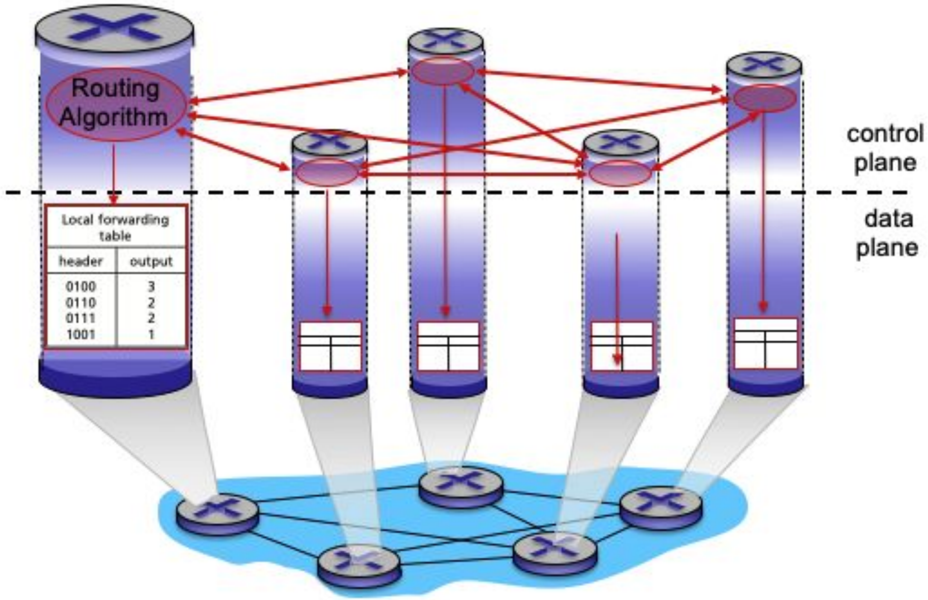
Routing protocols

Routing

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

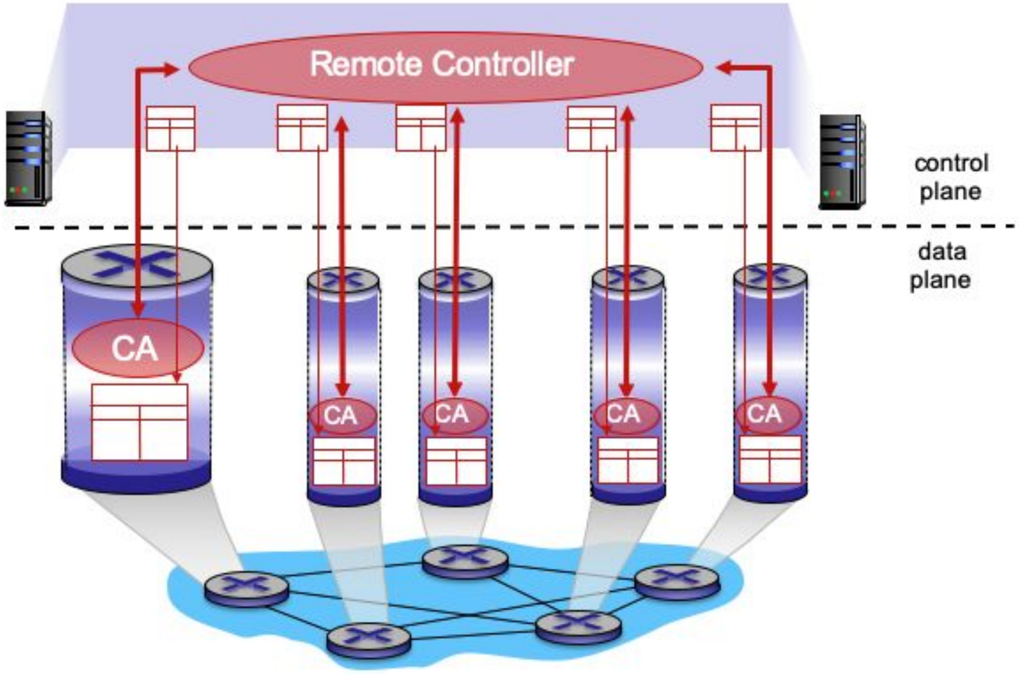
Per-router control

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



Logically centralized control

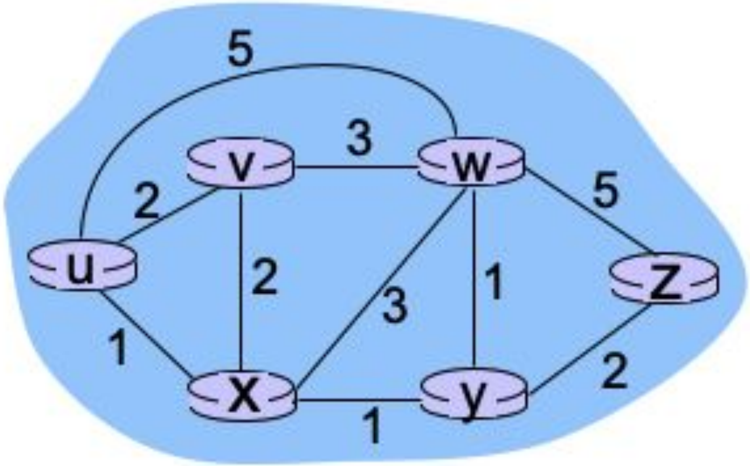
A distinct remote controller interacts with local control agents (CAs) in routers to compute forwarding tables



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

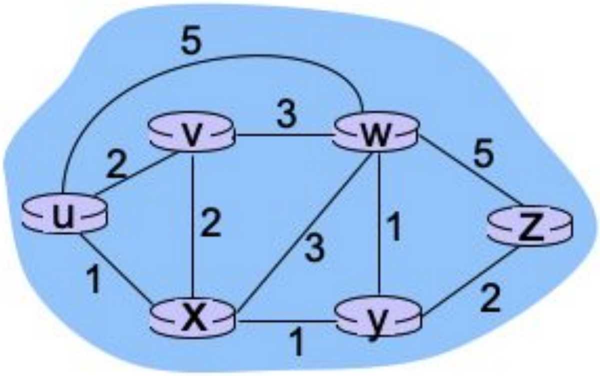


graph: G = (N,E)

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

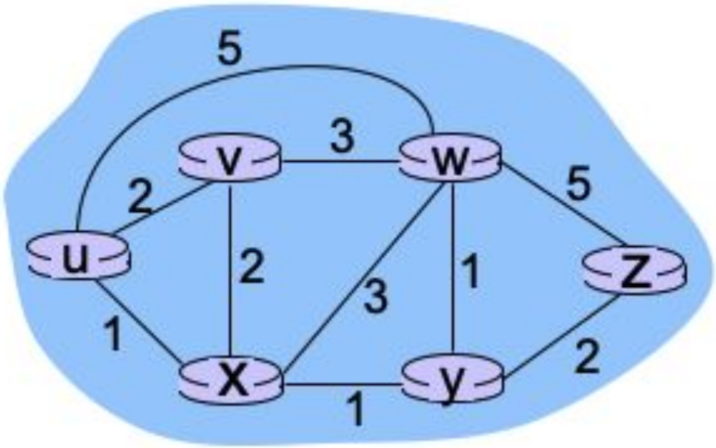
$$d_x(y) = \min_v \{c(x,v) + d_v(y)\}$$

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 $D_x = [D_x(y)$ for y 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 $D_v = [D_v(y)$ for all y in N]

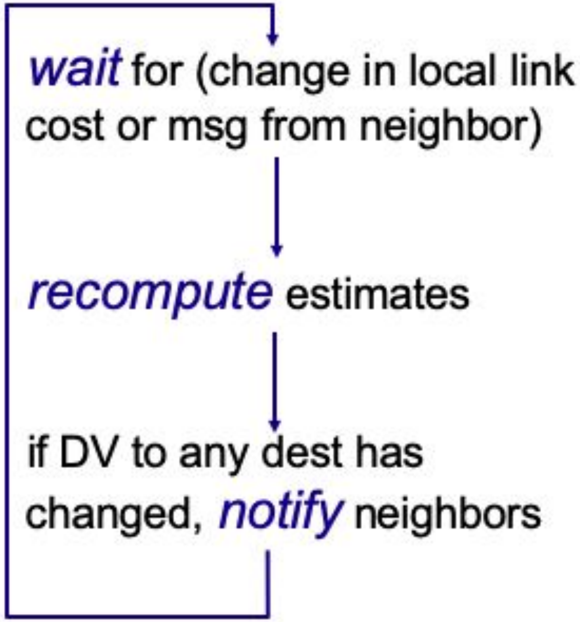
Distance vector algorithm

- 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

cost to

from	x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

cost to

from	x	y	z
x	0		
y	2	0	1
z	7	1	0

node y table

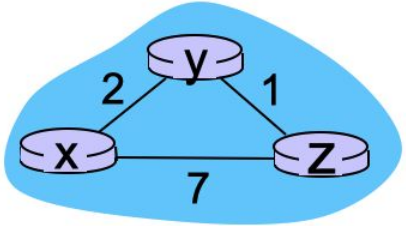
cost to

from	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

node z table

cost to

from	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	∞	∞	∞

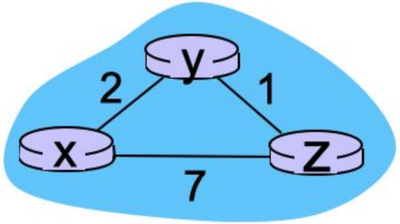
from \ cost to	x	y	z
x	0	2	
y	2	0	1
z	7	1	0

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	∞	∞	∞

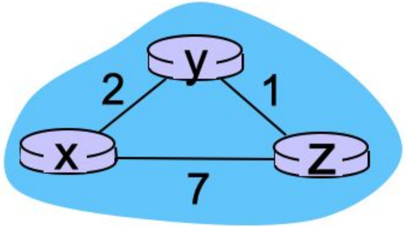
from \ cost to	x	y	z
x	0	2	3
y	2	0	1
z	7	1	0

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

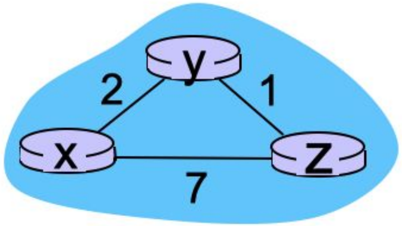
node z table

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

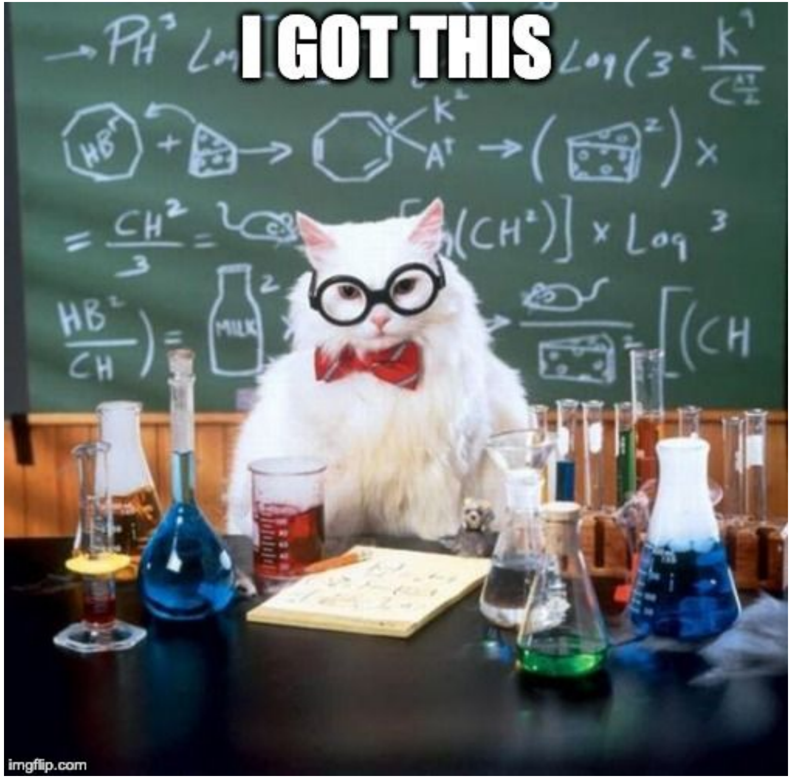
time

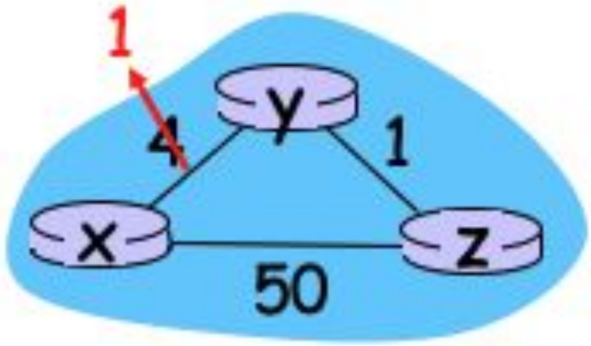


Link cost change

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

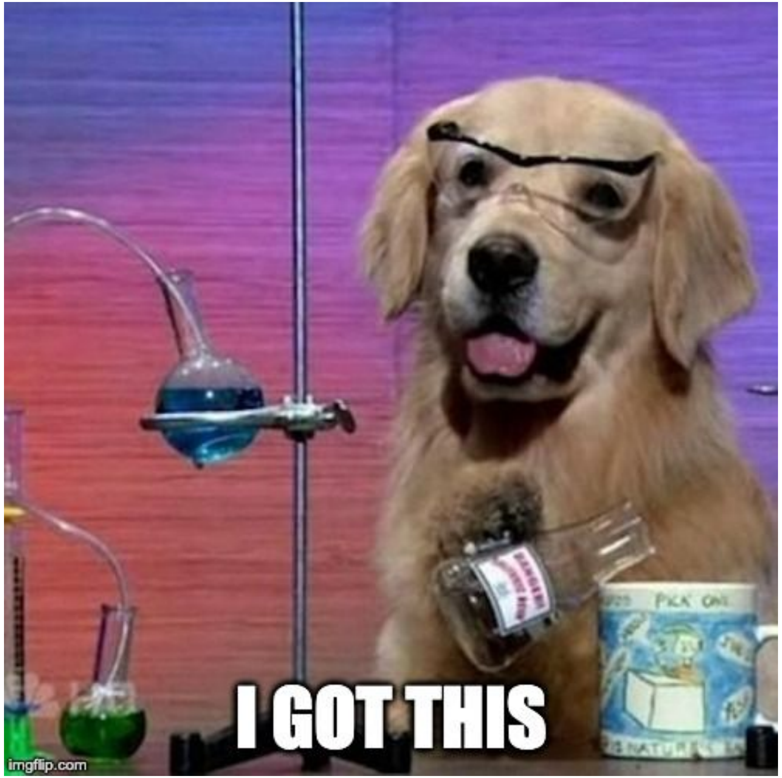
What happens when cost reduces?

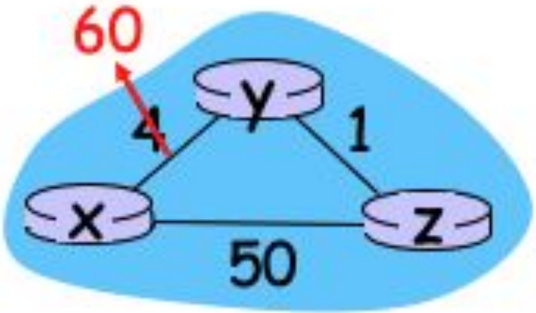




Distance vector algorithm

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

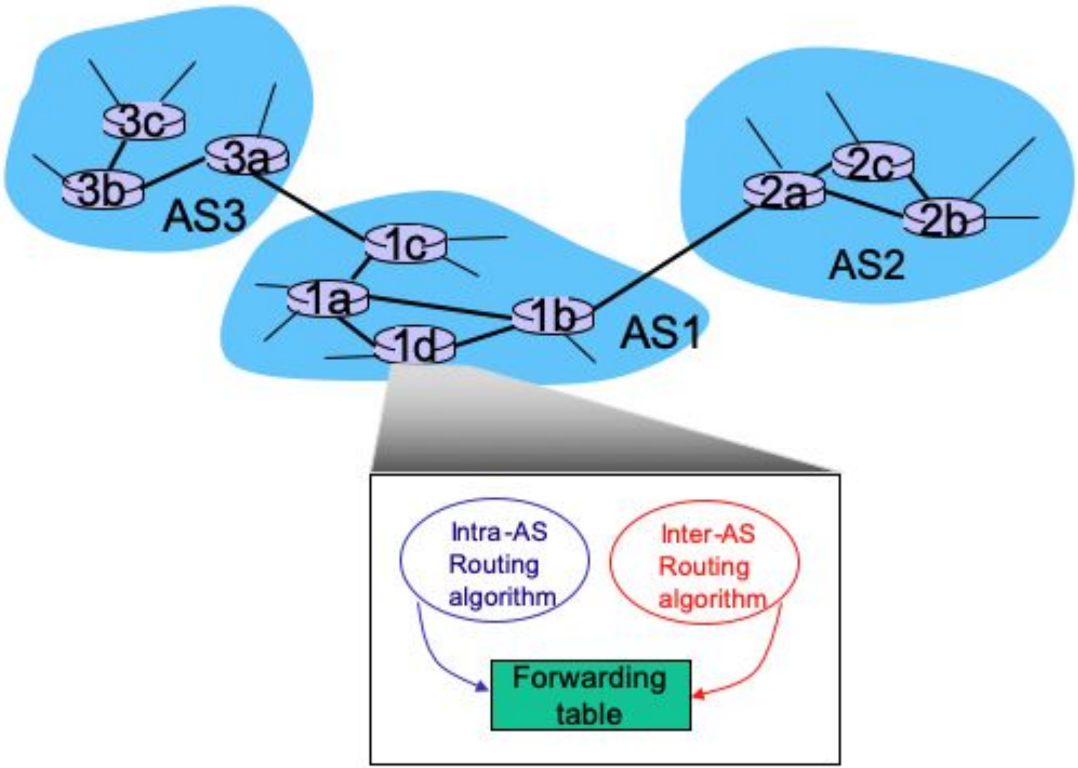
No!!!!!

- 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

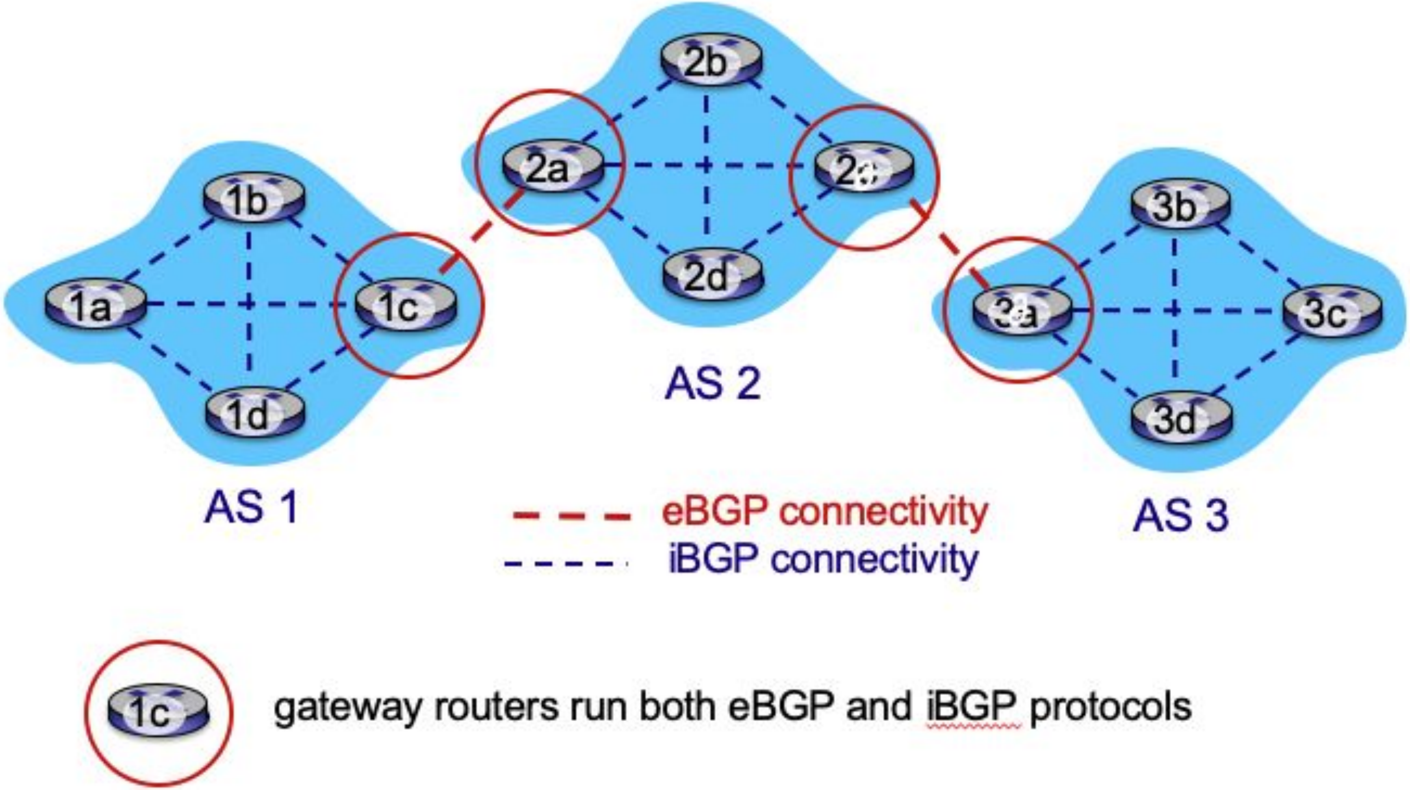
Inter-AS routing

- 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

BGP basics

- Designed to scale huge inter network like the Internet
- Send updates to manually defined neighbors
- Application layer protocol using TCP (port 179)
- AS-path information to prevent loop
- Path selection is complicated

BGP neighbors

- BGP neighbors are routers forming TCP connections to exchange BGP updates
    - manually configured
    - two types of neighbor relationship
        - iBGP (routers in the same AS)
        - eBGP (routers in different AS)

BGP neighbors

```
route-server> show ip bgp neighbors
BGP neighbor is 216.218.252.130, remote AS 6939, local AS 6939, internal link
  BGP version 4, remote router ID 216.218.252.130
  BGP state = Established, up for 00:41:47
  Last read 00:00:00, hold time is 180, keepalive interval is 60 seconds
  Neighbor capabilities:
    4 Byte AS: advertised and received
    Route refresh: advertised and received(old & new)
    Address family IPv4 Unicast: advertised and received
    Graceful Restart Capabilty: advertised
  Message statistics:
```

BGP neighbors

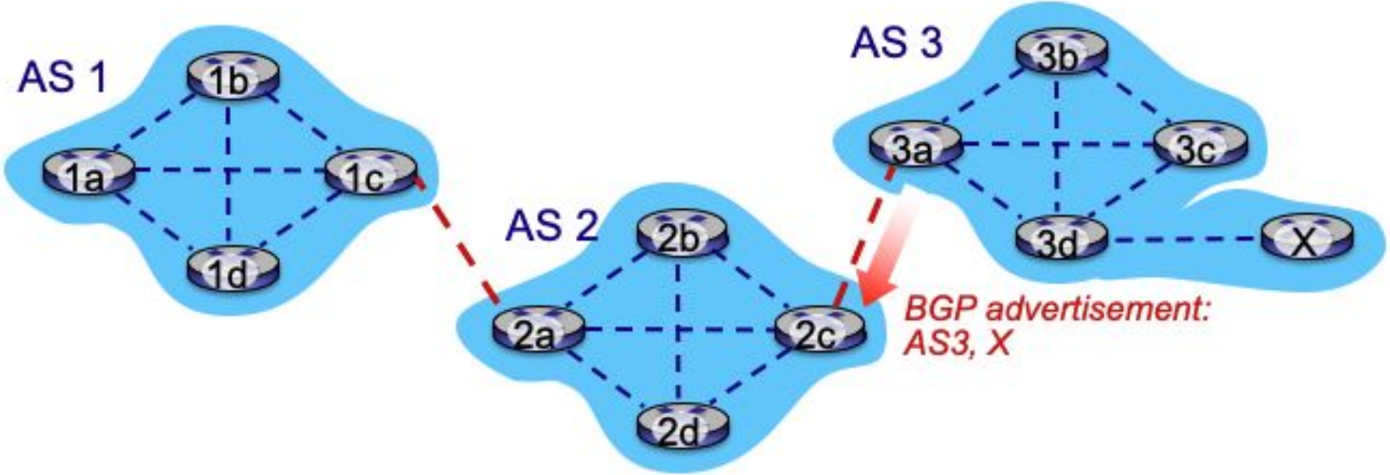
```
route-server> show ip bgp summary
BGP router identifier 64.62.142.154, local AS number 6939
RIB entries 1449658, using 155 MiB of memory
Peers 46, using 408 KiB of memory

Neighbor        V         AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
216.218.252.130 4  6939 3363817    6381        0    0    0 01:37:41   795891
216.218.252.147 4  6939 3645934    6379        0    0    0 01:37:39   796243
216.218.252.151 4  6939 3632194    6379        0    0    0 01:37:45   796245
216.218.252.154 4  6939 3260700    6382        0    0    0 01:37:25   795890
216.218.252.157 4  6939 3649388    6381        0    0    0 01:37:18   796245
216.218.252.164 4  6939 3505378    6379        0    0    0 01:38:15   796247
216.218.252.165 4  6939 3680065    6378        0    0    0 01:37:59   796247
216.218.252.167 4  6939 3453321    6382        0    0    0 01:37:57   795869
216.218.252.168 4  6939 3617330    6383        0    0    0 01:37:23   796244
216.218.252.169 4  6939 3123208    6378        0    0    0 01:37:32   796245
216.218.252.171 4  6939 3606008    6385        0    0    0 01:37:34   796243
216.218.252.173 4  6939 3243682    6382        0    0    0 01:37:21   795892
216.218.252.174 4  6939 3107783    6386        0    0    0 01:37:54   795888
216.218.252.176 4  6939 3363895    6382        0    0    0 01:37:16   796247
216.218.252.177 4  6939 3437999    6380        0    0    0 01:37:48   795869
216.218.252.178 4  6939 3526745    6382        0    0    0 01:37:30   796180
```

BGP route advertisement

- BGP routers send updates to neighbors
  - list of network prefixes is not enough
  - paths to different destination network prefixes
    - AS-PATH attribute
  - BGP is a “path vector” protocol

BGP route advertisement

- When AS3 gateway router 3a advertisers path “AS3,X” to AS2 gateway router 2c, AS3 promises to AS2 it will forward datagrams towards network X



BGP attributes

- Advertised prefix includes BGP attributes
  - prefix + attributes = “route”
- Two important attributes
  - AS-PATH: list of AS’es through which prefix advertisement has passed
  - NEXT-HOP: indicates specific internal-AS router to next hop AS

BGP attributes

```
route-server> show ip bgp
BGP table version is 0, local router ID is 64.62.142.154
Status codes: s suppressed, d damped, h history, * valid, > best, = multipath,
              i internal, r RIB-failure, S Stale, R Removed
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
* i1.0.0.0/24       216.218.252.168          1    100      0 13335 i
* i                 216.218.252.173          1    100      0 13335 i
* i                 198.32.146.195           1    100      0 13335 i
* i                 216.218.252.179          1    100      0 13335 i
* i                 216.218.252.169          1    100      0 13335 i
* i                 216.218.252.184          1    100      0 13335 i
*>i                 216.218.252.130          1    100      0 13335 i
```

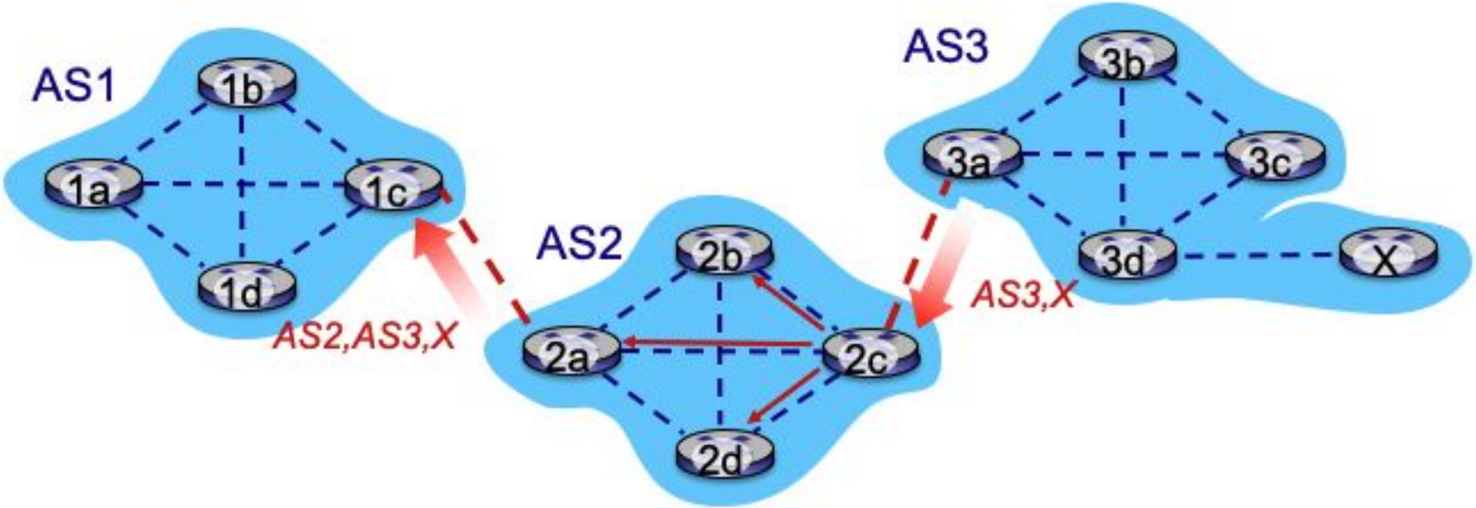
```
* i1.0.5.0/24       216.218.252.180          1    140      0 4826 38803 56203 i
* i                 64.71.184.46           100    140      0 4826 38803 56203 i
```

Policy-based routing

- Gateway receiving route advertisement uses import policy to accept/decline path
  - e.g. never route through AS Y
- AS policy also determines whether to advertise path to other neighboring AS'es

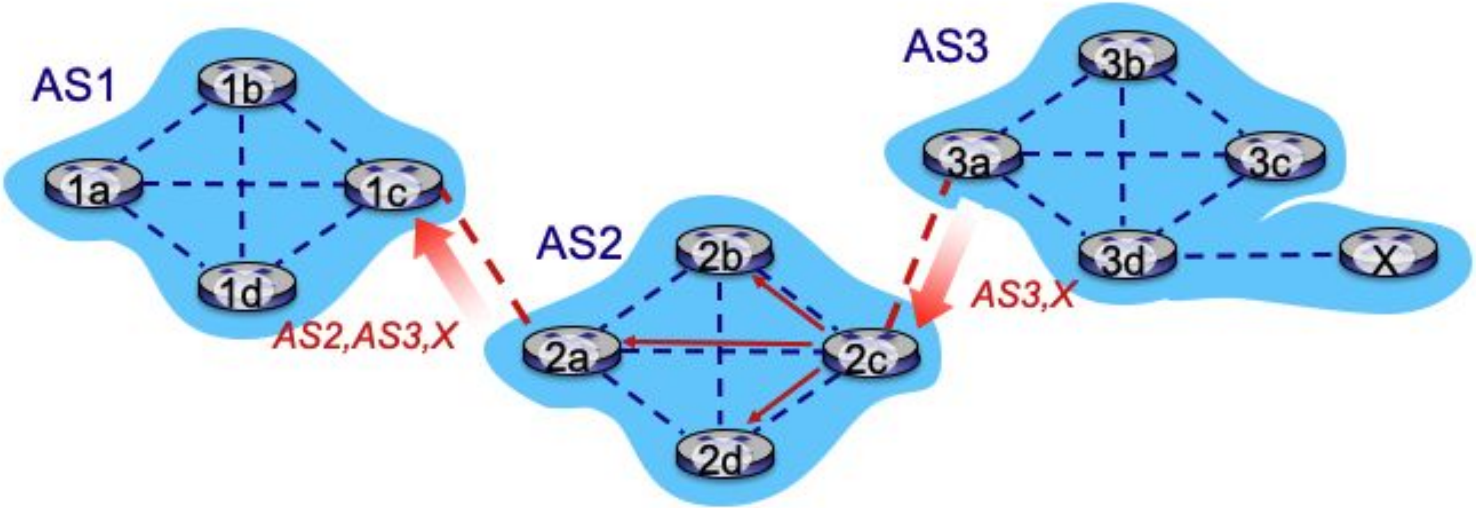
BGP advertisement

- AS2 router 2c receives path advertisement “AS3,X” (via eBGP) from AS3 router 3a



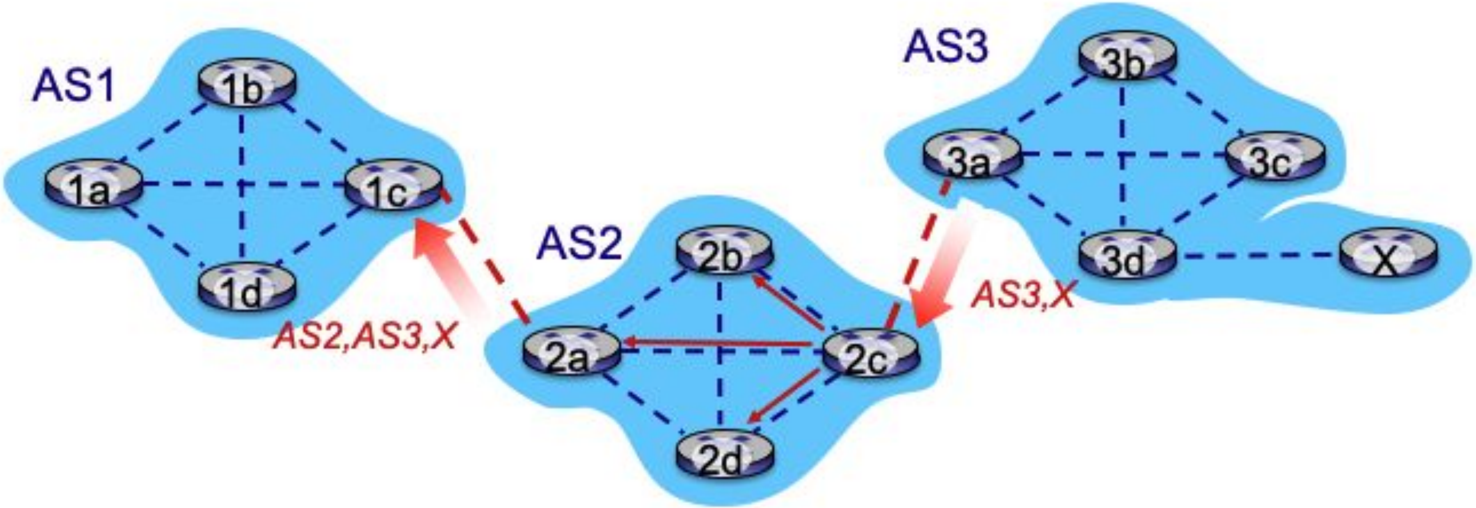
BGP advertisement

- Based on AS2 policy, AS2 router 2c accepts path “AS3,X” and propagates (via iBGP) to AS2 routers



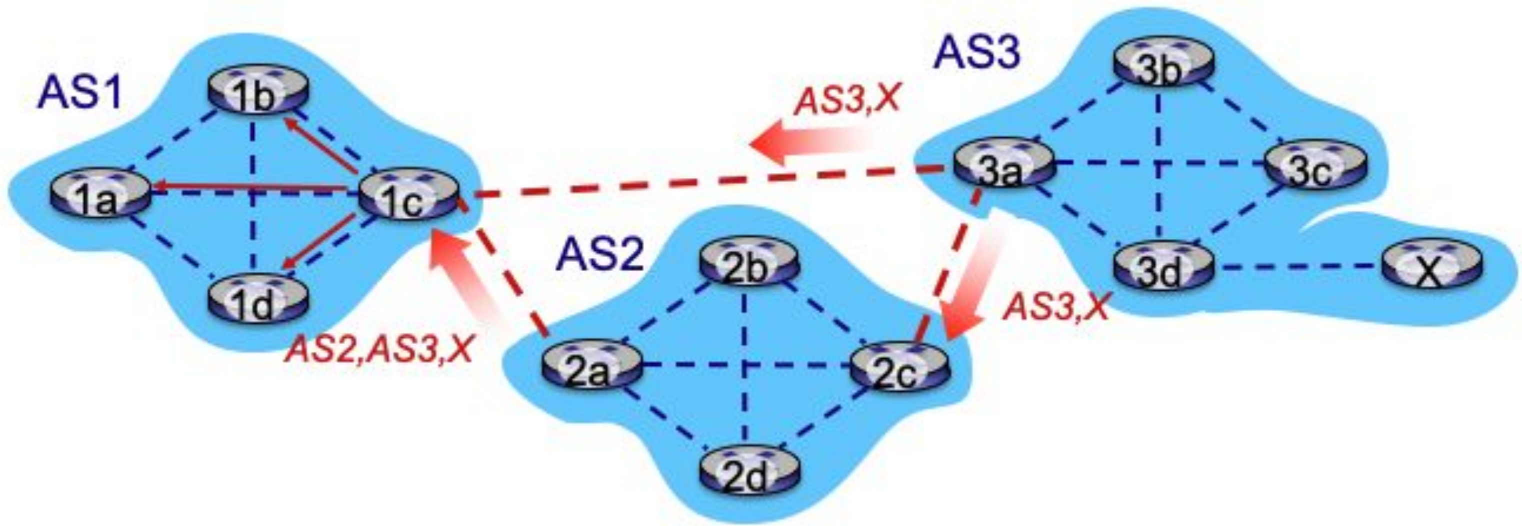
BGP advertisement

- Based on AS2 policy, AS2 router 2a advertises (via eBGP) path “AS2,AS3,X” to AS1 router 1c



BGP advertisement

- Gateway router may learn about multiple paths to destination

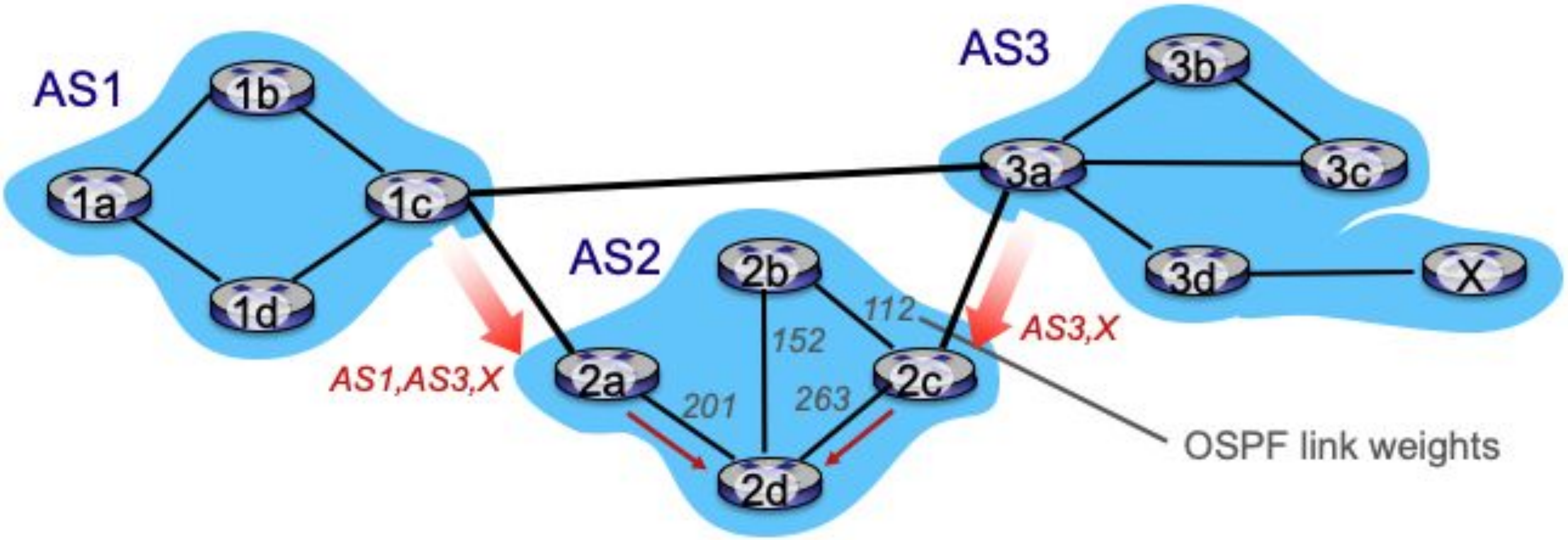


BGP route selection

- Router may learn about more than one route to destination prefix, select route based on
    - local preference value attribute
    - shortest AS-PATH
    - closest NEXT-HOP router
    - additional criteria

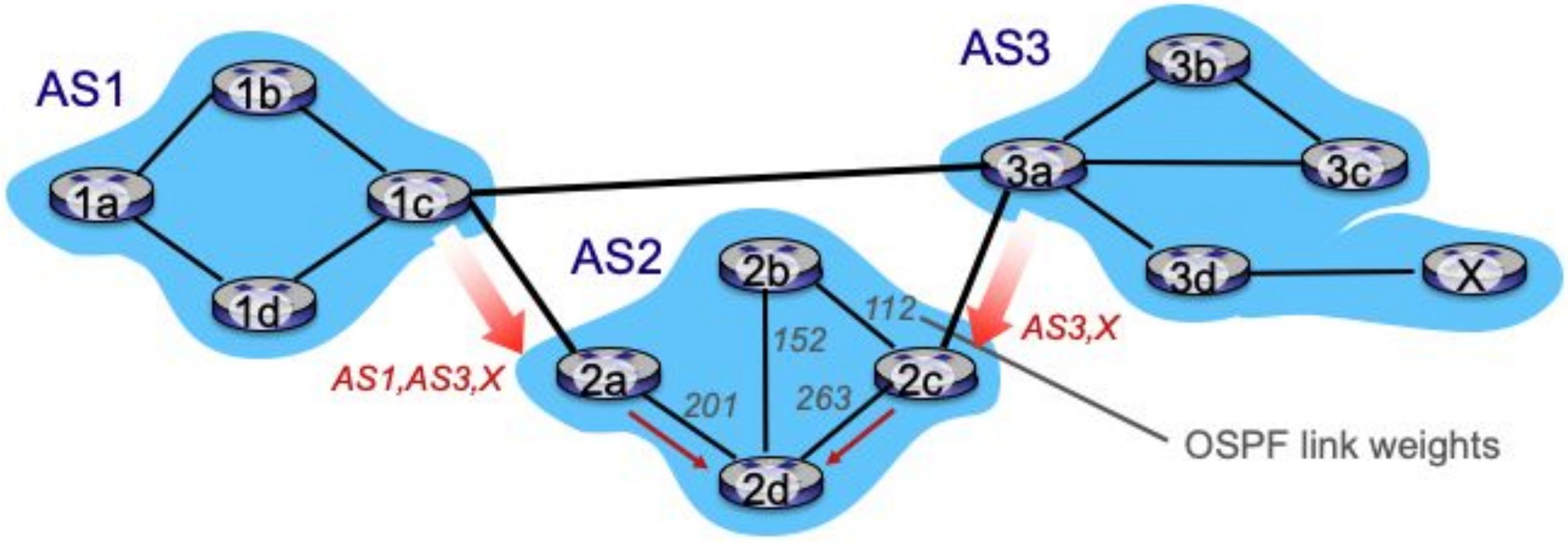
Hot potato routing

- 2d learns (via iBGP) it can route to X via 2a or 2c
- Which one should 2d choose?



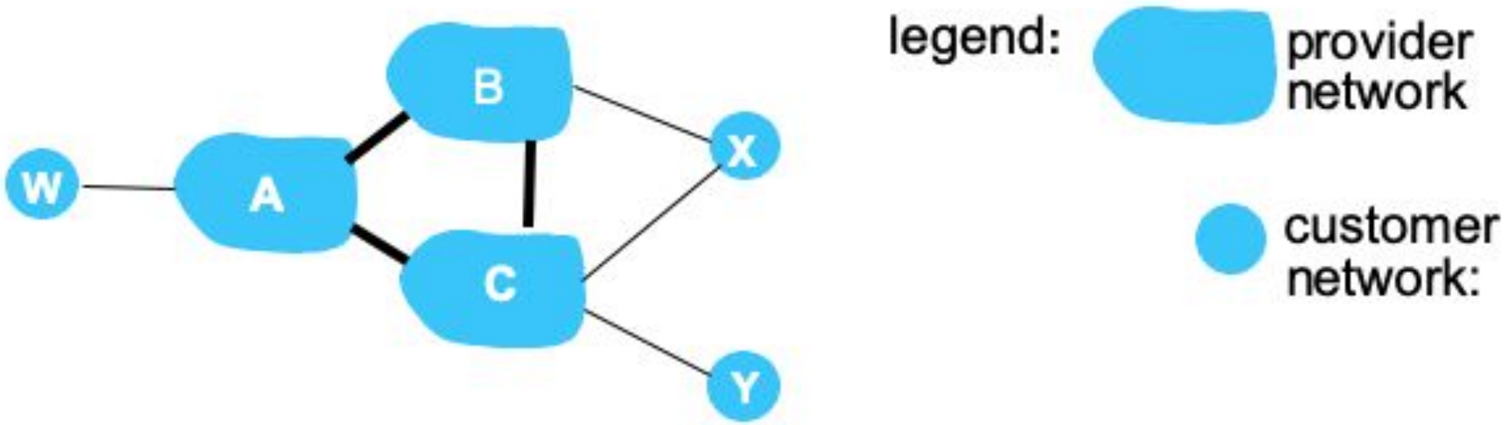
Hot potato routing

- Hot potato routing: choose local gateway that has the least intra domain cost



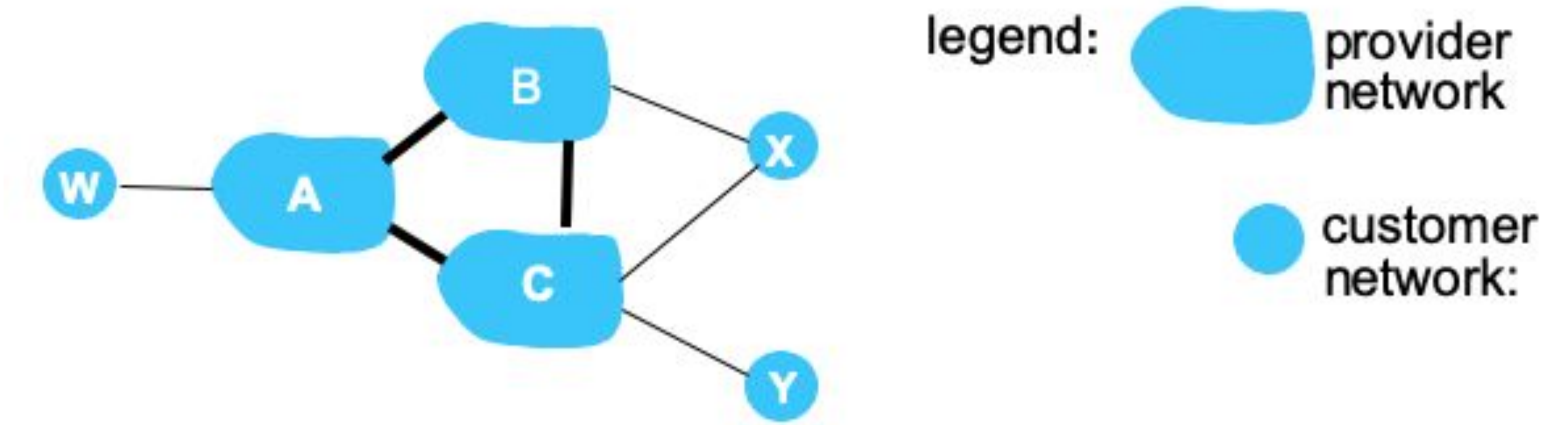
BGP prefix advertisement

- A, B, C are provider networks
- X, Y, Z are customers
  - X is dual-homed



BGP prefix advertisement

- X needs to be careful when designing policy
- X does not want to route from B to C via X
  - so X will not advertise to B a route to C



So far...

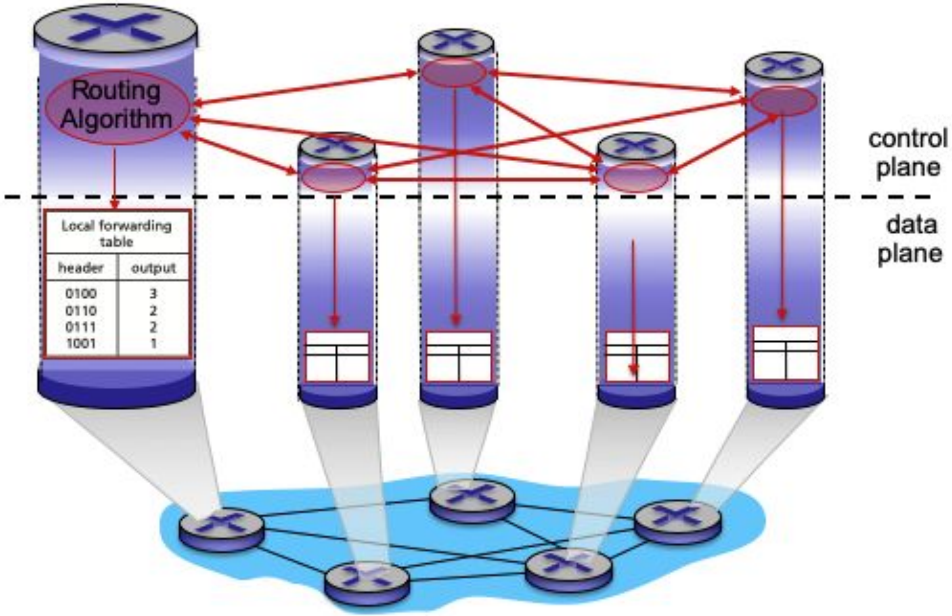
- Main functionality in network layer
    - data plane: forwarding
    - control plane: routing
- Routing algorithms
    - intra-domain
        - link-state (e.g. OSFP), distance vector (e.g. RIP)
    - inter-domain
        - BGP

SDN - software defined networking

- Internet network layer
  - Historically has been implemented via distributed, per-router approach
  - Monolithic router contains contains switching hardware, runs proprietary implementation of standard protocols (e.g. OSPF, BGP) in proprietary router OS

SDN

- Individual routing algorithm components in each and every router interact with each other in control plane

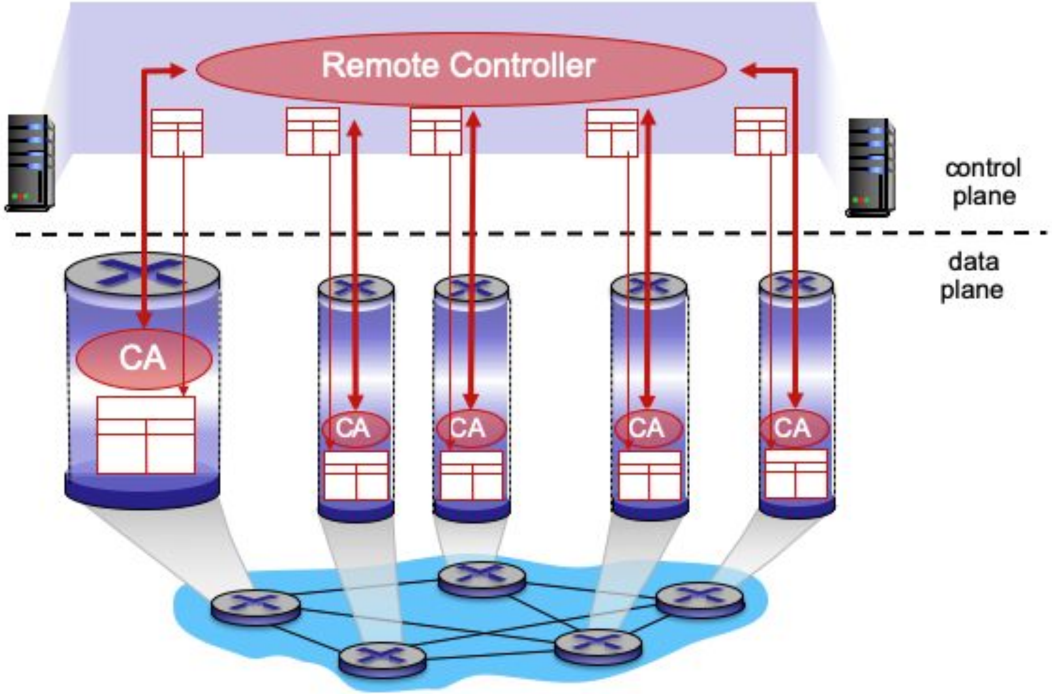


That implies

- Limitation in scalability
- DevOps headache
- Harder to make changes
- Incompatibility between different vendors
- etc. etc.

SDN

- Logically centralized control plane

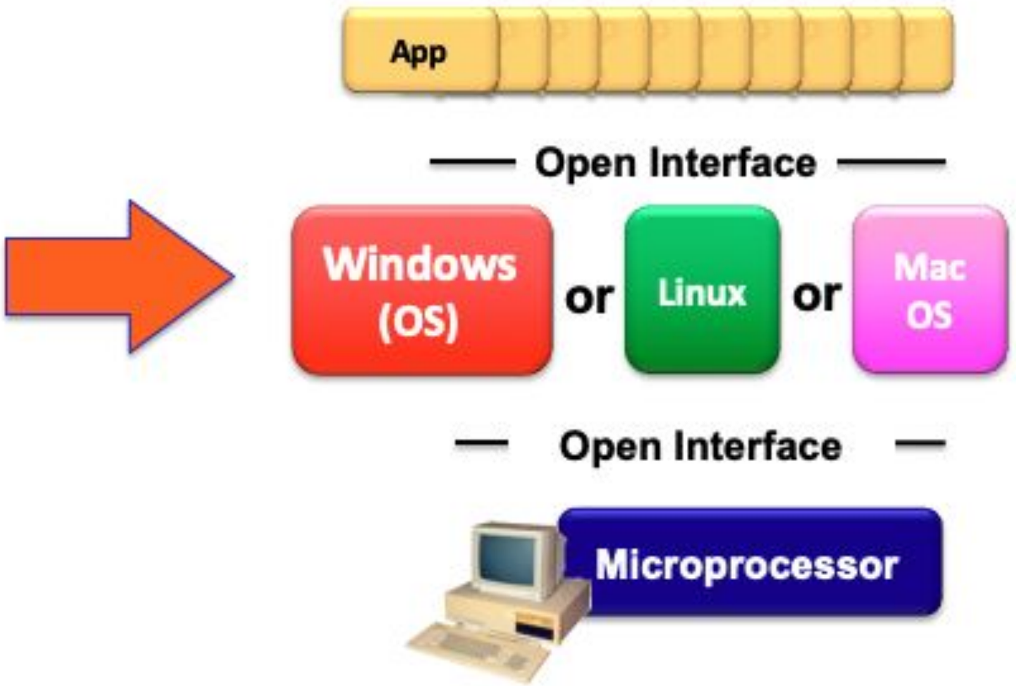


SDN - goals

- Easier network management
- Programmable
  - centralized “programming” is easier
    - compute forwarding table centrally and distribute
  - distributed “programming” is much harder
- Open (non-proprietary)

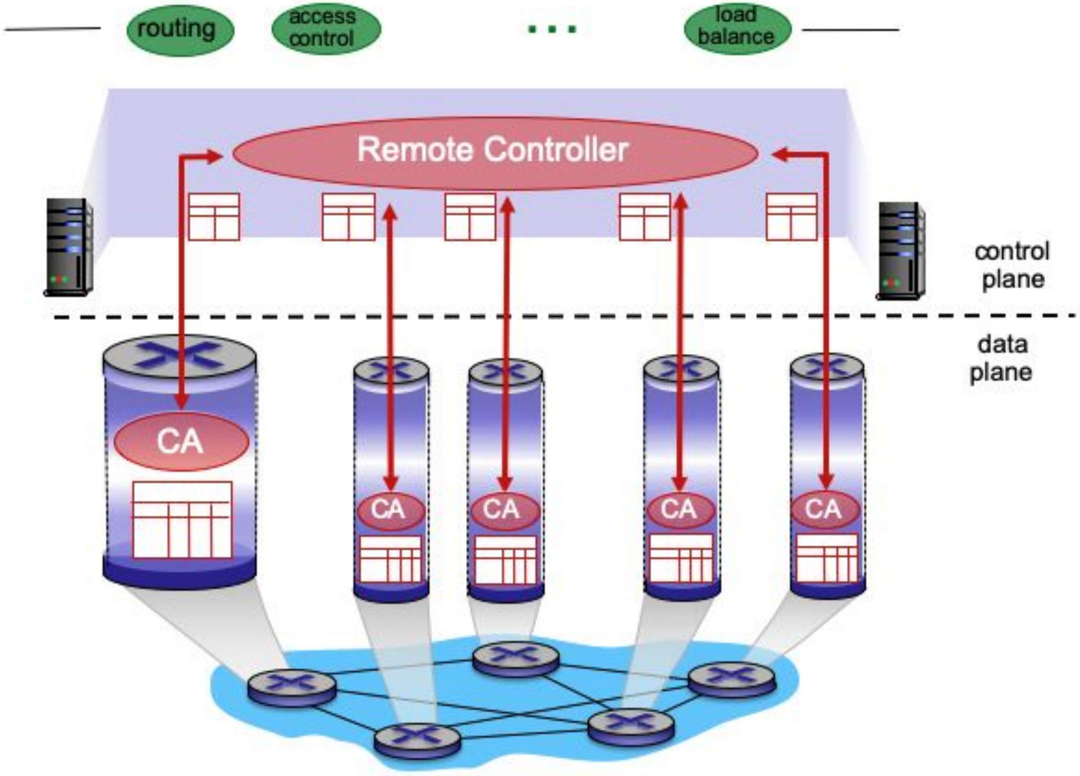
Analogy

Specialized Applications
Specialized Operating System
Specialized Hardware



App
Open Interface
Windows (OS)
or
Linux
or
Mac OS
Open Interface
Microprocessor

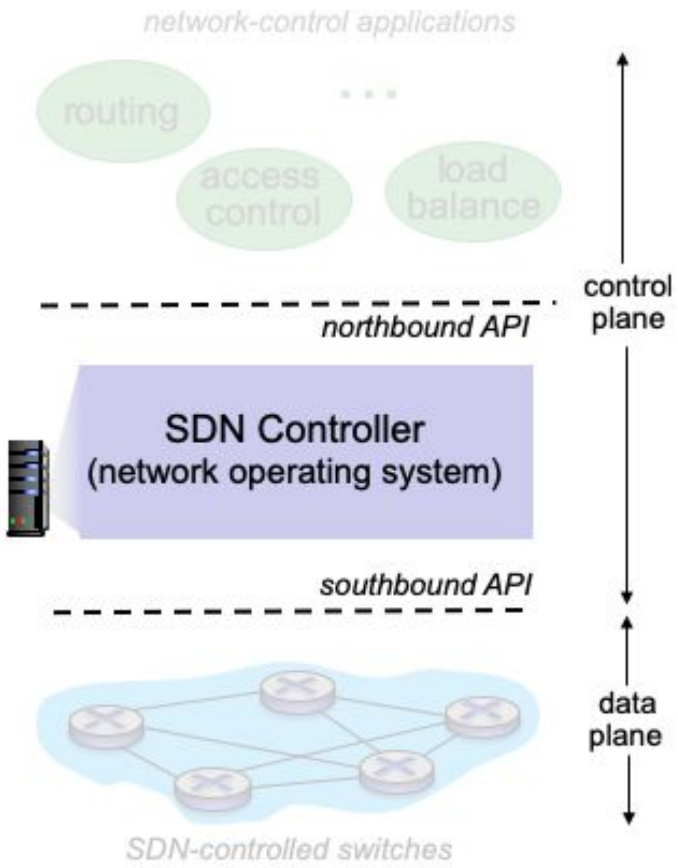
Analogy



routing
access control
...
load balance
Remote Controller
control plane
data plane
CA
CA
CA
CA
CA

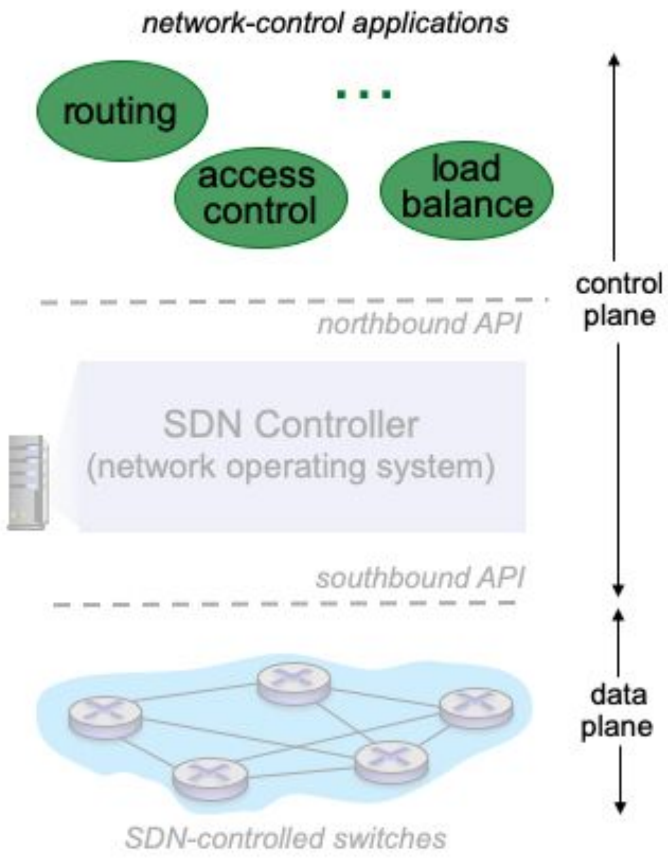
SDN - controller

- Maintain network state information
- Interact with network devices "below" via southbound API
- Interact with application "above" via northbound API
- Implemented as distributed system for scalability, availability



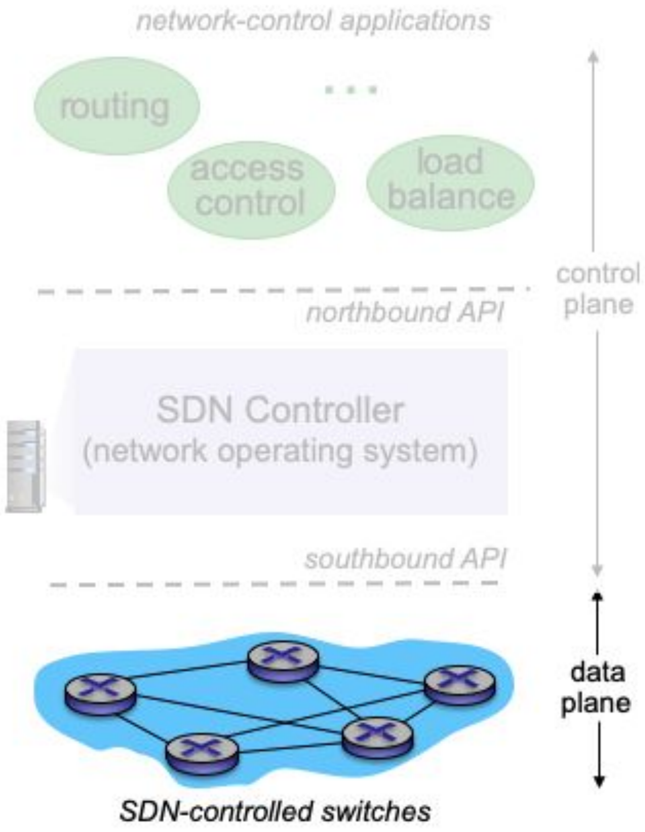
SDN - application

- Implement control functions using lower-level services provided by SDN controller



SDN - data plane

- Fast, simple, commodity layer 2 & 3 devices implementing generalized data-plane forwarding
- Table computed and installed by controller
- Protocol for communicating with controller (e.g. OpenFlow)



ICMP

ICMP

- Internet Control Message Protocol
- Used by hosts & routers to communicate network level information
  - e.g. echo request/reply (used by ping)
- Network layer “above” IP
  - carried in IP datagram

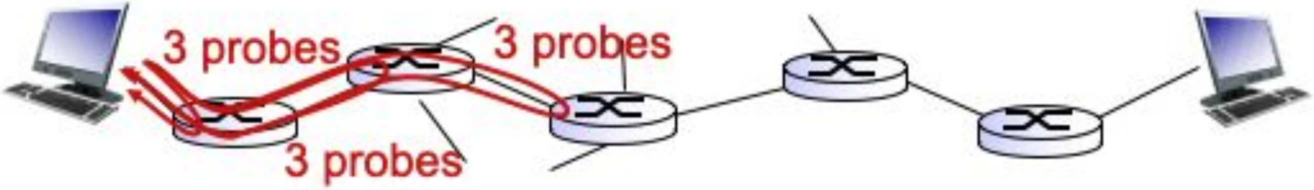
ICMP

- ICMP message
  - type/code
  - checksum
  - other header fields depends on (type/code)
  - partial original IP datagram

Type	Code	description
0	0	echo reply (ping)
3	0	dest. network unreachable
3	1	dest host unreachable
3	2	dest protocol unreachable
3	3	dest port unreachable
3	6	dest network unknown
3	7	dest host unknown
4	0	source quench (congestion control - not used)
8	0	echo request (ping)
9	0	route advertisement
10	0	router discovery
11	0	TTL expired
12	0	bad IP header

Traceroute

- Source sends series of UDP segment to destination
    - Set TTL=1, TTL=2, etc.
- When datagram with TTL=n arrives to nth router
    - router discards datagram and sends source ICMP message (type 11, code 0)
    - ICMP message include name of router & IP address



Summary

- Forwarding & routing
- IP v4 and v6
- Routing protocols
  - Intra-domain & inter-domain
- ICMP

Questions?