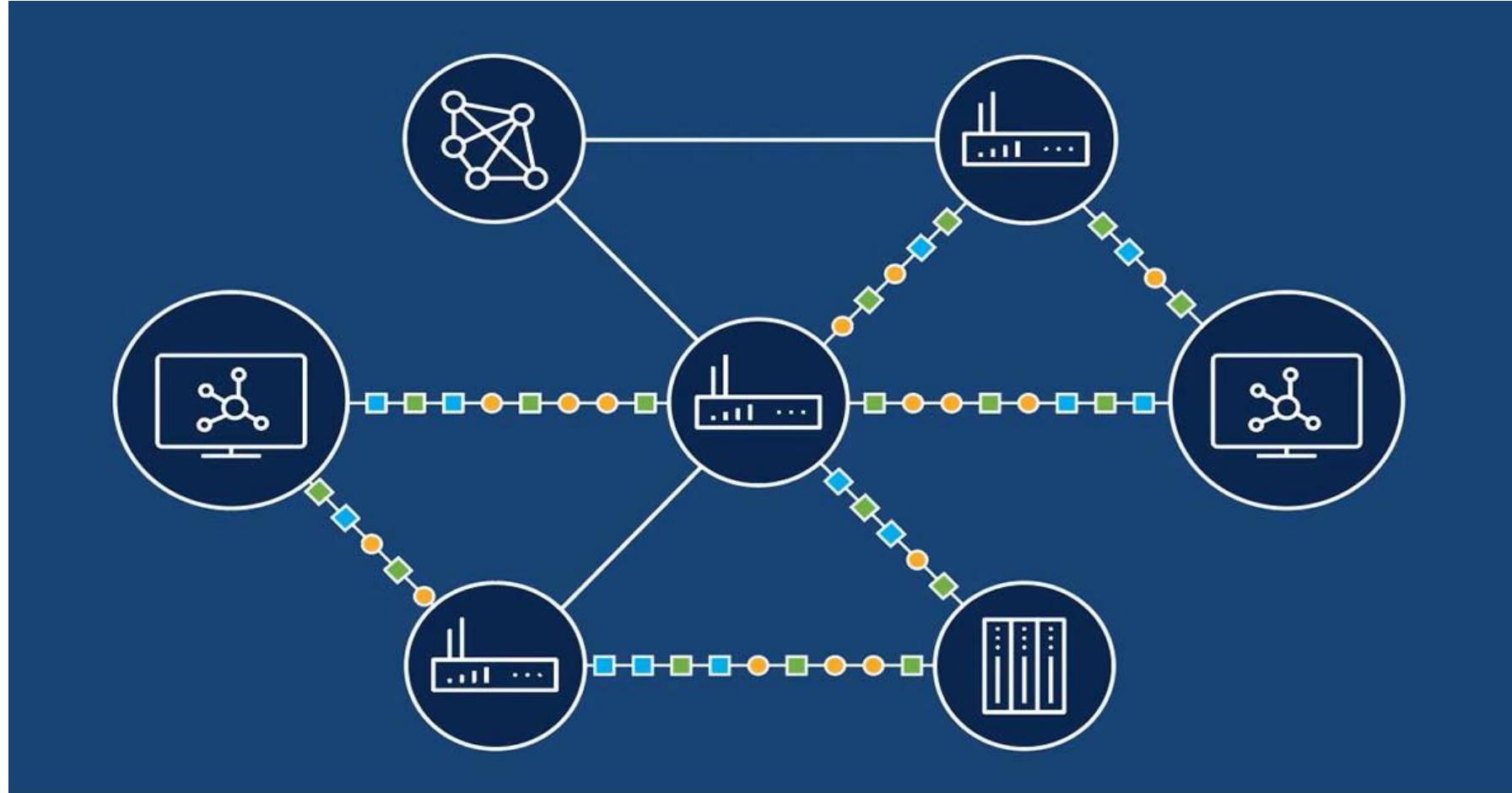


# CAN201: Introduction to Networking

## Lecture 8 - The Network Layer: Control Plane



Lecturer: Dr. Gordon Boateng & Dr. Fei Cheng

# Important Information

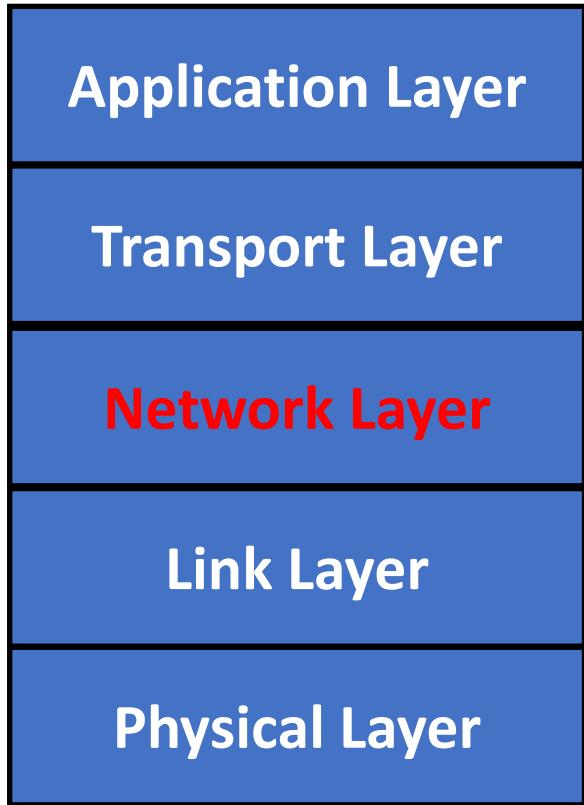
## ■ Contact:

- Email: [Gordon.Boateng@xjtu.edu.cn](mailto:Gordon.Boateng@xjtu.edu.cn)
- Office No.: SC 554A

## ■ Office Hours (Strictly via appointment)

- Tuesday: 14:00-15:00
- Wednesday: 14:00-15:00

# Recap: Top-Down Approach



# Network-Layer Functions

- **routing:** determine route taken by packets from source to destination
- **forwarding:** move packets from router's input to appropriate router output

*control plane*

*data plane*

# Network Layer: Control Plane

- intra-ISP routing: OSPF
- routing among ISPs: BGP
- SDN control plane
- Internet Control Message Protocol
- Network management, configuration

# Making Routing Scalable

our routing study thus far:  
*idealized*

- all routers identical
- network “flat”

... not true in practice

**scale:** billions of destinations:

- can't store all destinations in routing tables!
- exchanging link-state or DV information would swamp links!

**administrative autonomy:**

- Internet: a network of networks
- each network admin may want to control routing in its own network

# Internet Approach to Scalable Routing

Aggregate routers into regions known as “autonomous systems” (AS) (a.k.a. “domains”)

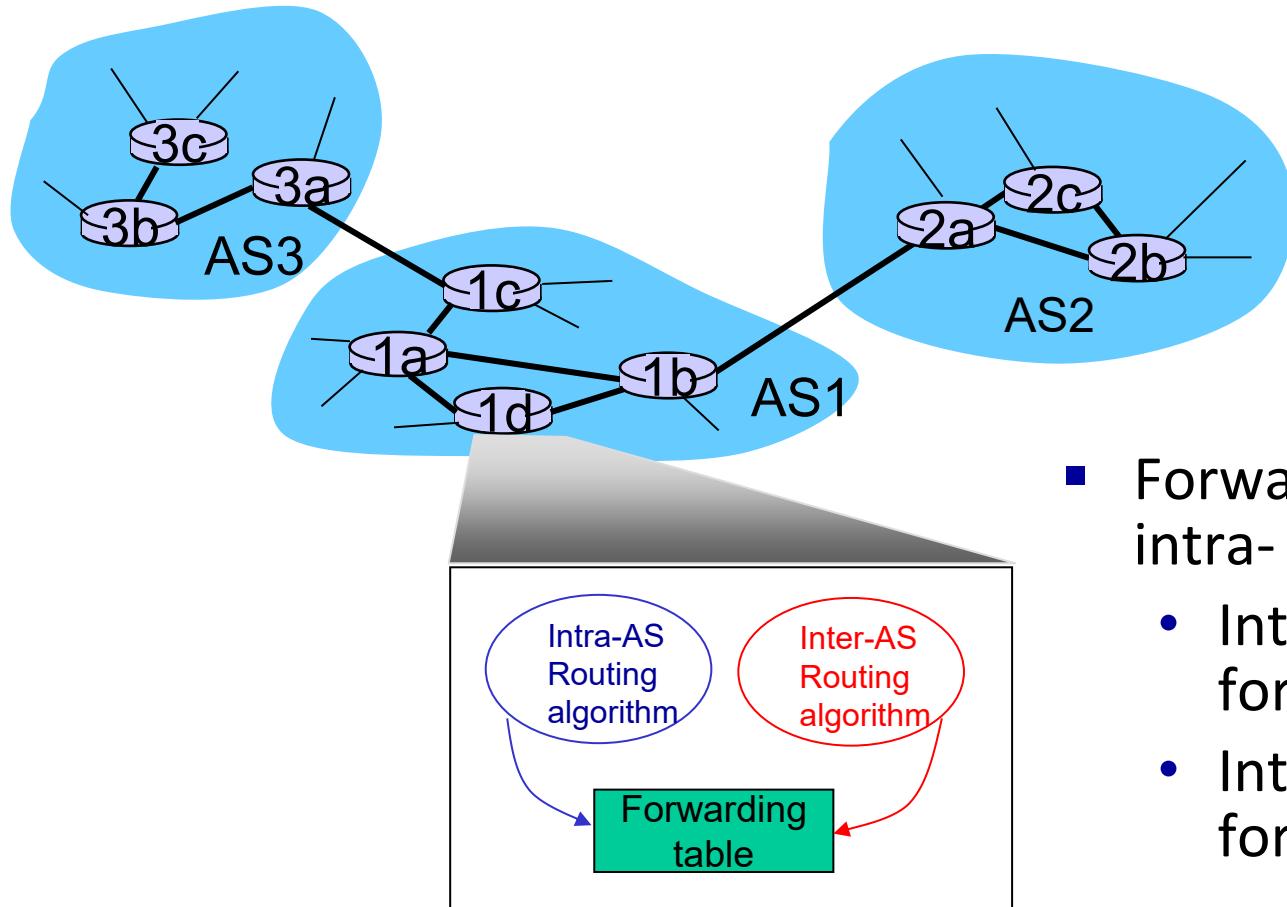
Intra-AS (aka “intra-domain”):  
routing among routers *within same AS (“network”)*

- all routers in AS must run same intra-domain protocol
- routers in different AS can run different intra-domain routing protocols
- **gateway router:** at “edge” of its own AS, has link(s) to router(s) in other AS'es

inter-AS (aka “inter-domain”):  
routing *among* AS'es

- gateways perform inter-domain routing (as well as intra-domain routing)

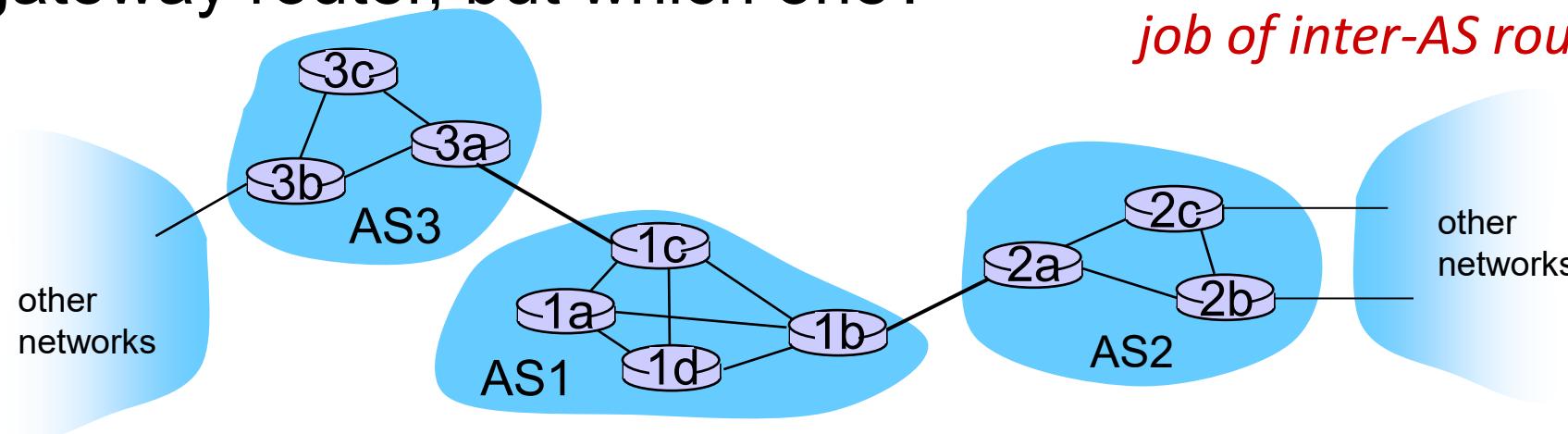
# Interconnected ASes



- Forwarding table configured by both intra- and inter-AS routing algorithm
  - Intra-AS routing determines entries for destinations within AS
  - Inter-AS & intra-AS determine entries for external destinations

# Inter-AS tasks

- Suppose router in AS1 receives datagram destined outside of AS1:
  - Router should forward packet to gateway router, but which one?



*AS1 must:*

1. Learn which destinations are reachable through AS2, which through AS3
2. Propagate this reachability info to all routers in AS1

*job of inter-AS routing!*

# Routing Within an AS

most common intra-AS routing protocols:

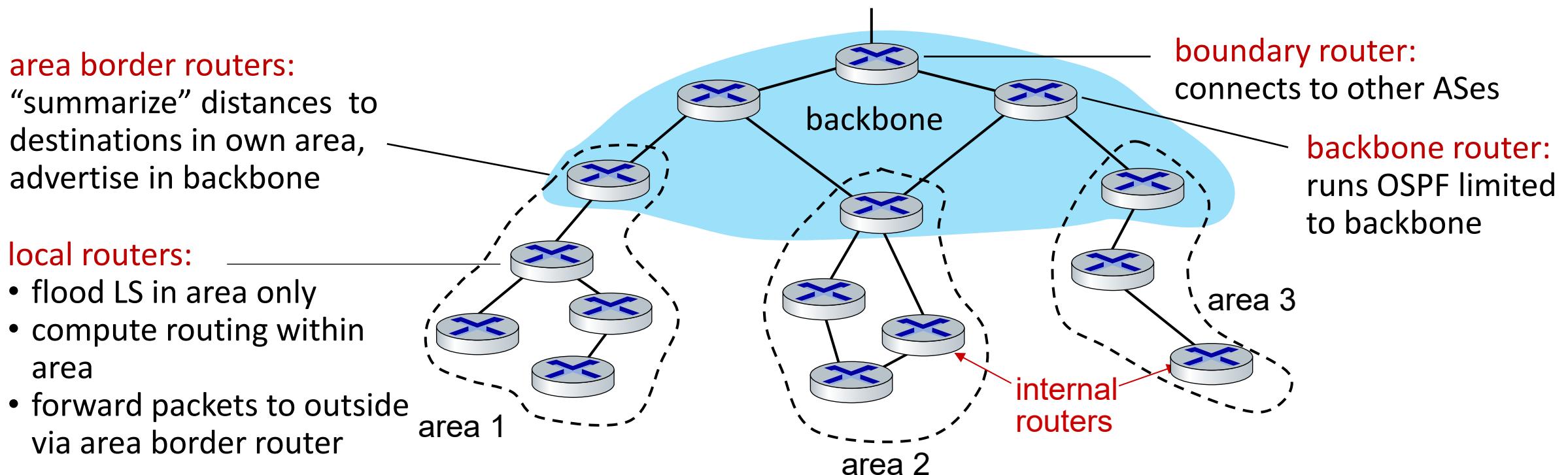
- **RIP: Routing Information Protocol [RFC 1723]**
  - classic DV: DVs exchanged every 30 secs
  - no longer widely used
- **OSPF: Open Shortest Path First [RFC 2328]**
  - classic link-state routing
  - IS-IS protocol (ISO standard, not RFC standard) essentially same as OSPF
- **EIGRP: Enhanced Interior Gateway Routing Protocol**
  - DV based
  - formerly Cisco-proprietary for decades
  - became open in 2013 [RFC 7868])

# OSPF (Open Shortest Path First) Routing

- “open”: publicly available
- classic link-state
  - each router floods OSPF link-state advertisements (directly over IP rather than using TCP/UDP) to all other routers in entire AS
  - multiple link costs metrics possible: bandwidth, delay
  - each router has full topology, uses Dijkstra’s algorithm to compute forwarding table
- *security*: all OSPF messages authenticated (to prevent malicious intrusion)

# Hierarchical OSPF

- **two-level hierarchy:** local area, backbone.
  - link-state advertisements flooded only in area, or backbone
  - each node has detailed area topology; only knows direction to reach other destinations



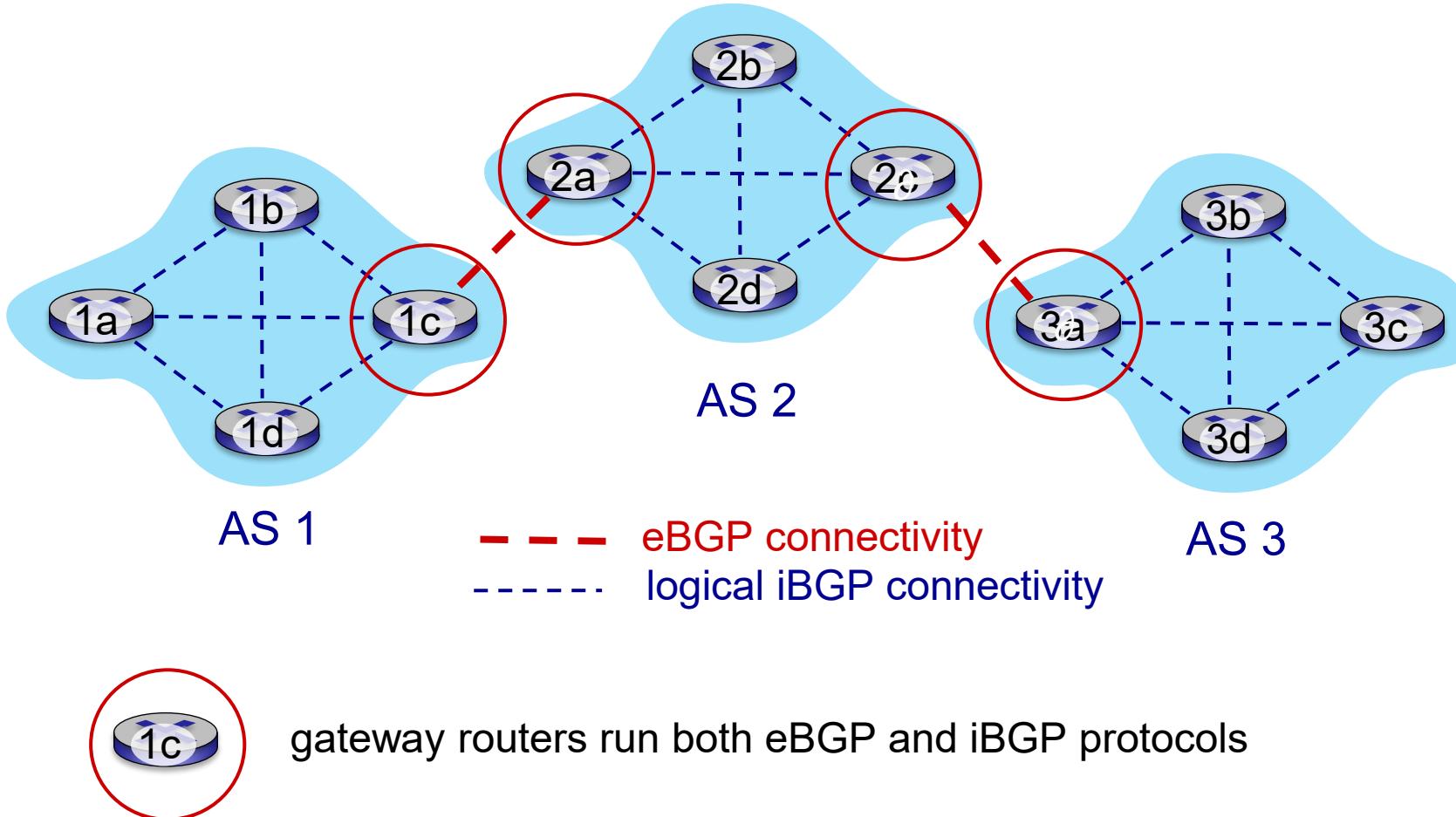
# Network Layer: Control Plane

- intra-ISP routing: OSPF
- **routing among ISPs: BGP**
- SDN control plane
- Internet Control Message Protocol
- Network management, configuration

# Internet inter-AS routing: BGP

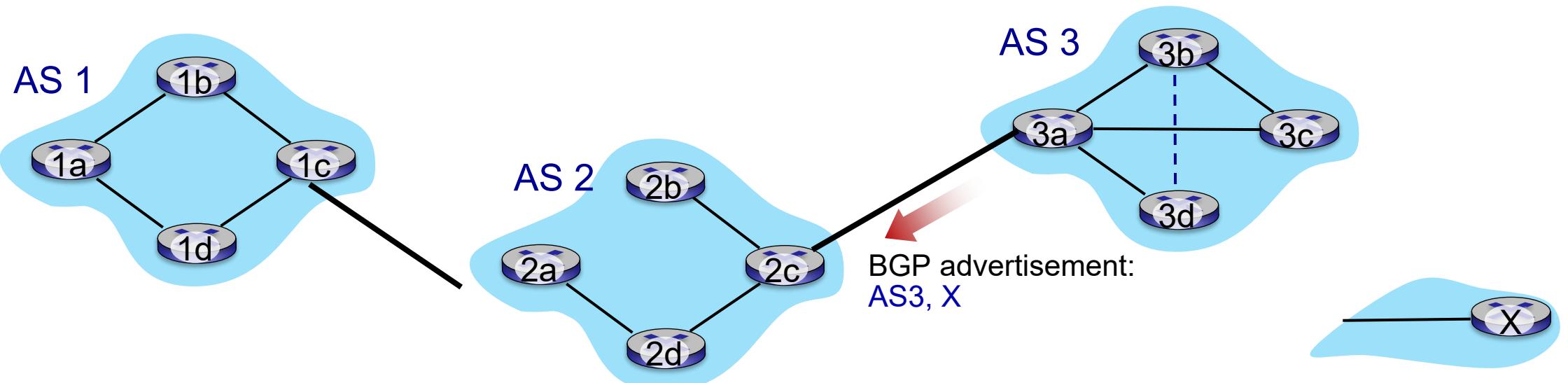
- BGP (Border Gateway Protocol): *the de facto* inter-domain routing protocol
  - “glue that holds the Internet together”
- allows subnet to advertise its existence, and the destinations it can reach, to rest of Internet: *“I am here, here is who I can reach, and how”*
- BGP provides each AS router a means to:
  - obtain destination network reachability info from neighboring ASes (**eBGP**)
  - determine routes to other networks based on reachability information and *policy*
  - propagate reachability information to all AS-internal routers (**iBGP**)
  - **advertise** (to neighboring networks) destination reachability info

# eBGP, iBGP Connections



# BGP Basics

- **BGP session:** two BGP routers (“peers, speakers”) exchange BGP messages over semi-permanent TCP connection:
  - advertising *paths* to different destination network prefixes (e.g., to a destination /16 network)
  - BGP is a “path vector” protocol
- when AS3 gateway 3a advertises **path AS3,X** to AS2 gateway 2c:
  - AS3 *promises* to AS2 it will forward datagrams towards X



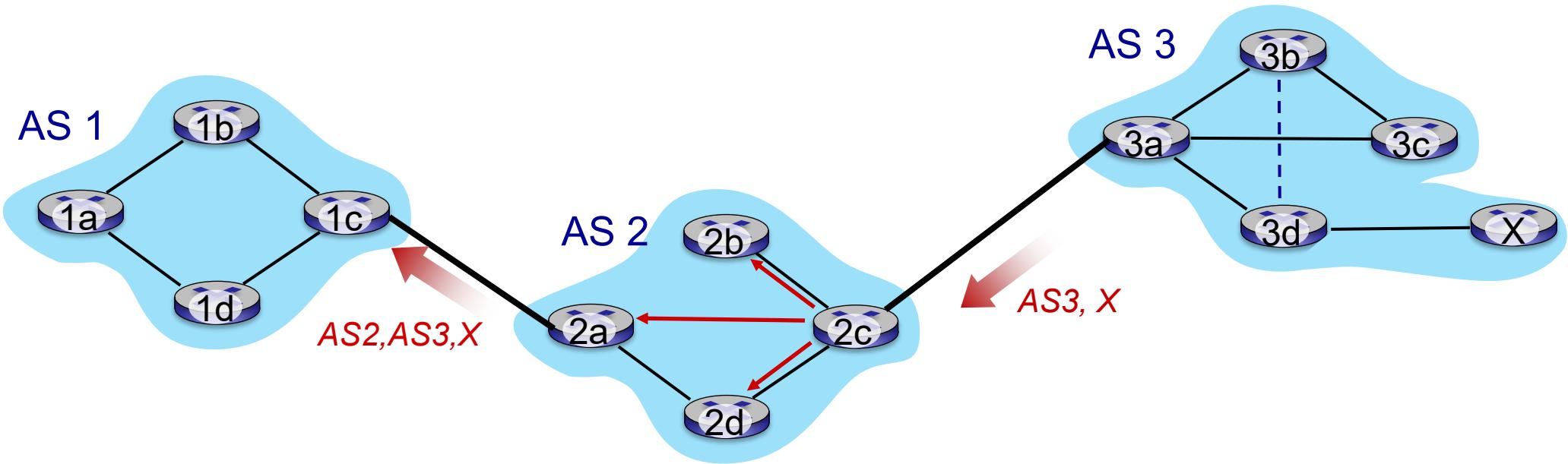
# BGP Protocol Messages

- BGP messages exchanged between peers over TCP connection
- BGP messages [RFC 4371]:
  - **OPEN**: opens TCP connection to remote BGP peer and authenticates sending BGP peer
  - **UPDATE**: advertises new path (or withdraws old)
  - **KEEPALIVE**: keeps connection alive in absence of UPDATES; also ACKs OPEN request
  - **NOTIFICATION**: reports errors in previous message; also used to close connection

# Path Attributes and BGP Routes

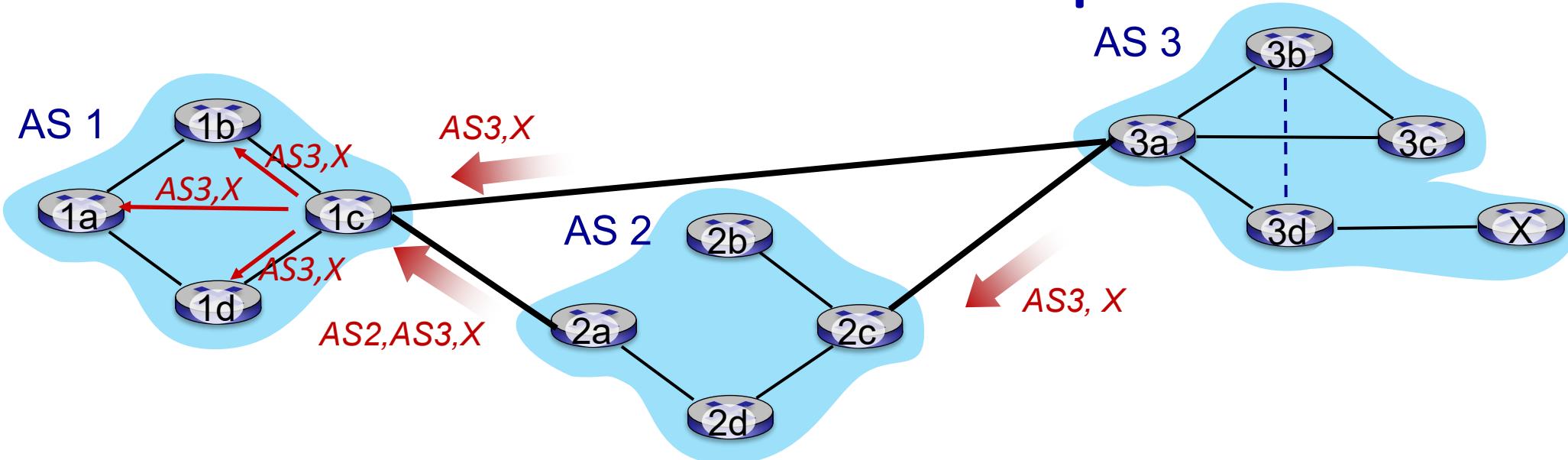
- BGP advertised path: prefix + attributes = “route”
  - path prefix: destination being advertised
  - two important attributes:
    - AS-PATH: list of ASes through which prefix advertisement has passed
    - NEXT-HOP: indicates specific internal-AS router to next-hop AS
- **policy-based routing:**
  - router receiving route advertisement to destination X uses *policy* to accept/reject a path (e.g., never route through AS W, or country Y).
  - router uses policy to decide whether to *advertise* a path to neighboring AS Z (does router want to route traffic forwarded from Z destined to X?)

# BGP Path Advertisement



- AS2 router 2c receives path advertisement **AS3,X** (via eBGP) from AS3 router 3a
- Based on AS2 policy, AS2 router 2c accepts path AS3,X, propagates (via iBGP) to all AS2 routers
- Based on AS2 policy, AS2 router 2a advertises (via eBGP) path **AS2, AS3, X** to AS1 router 1c

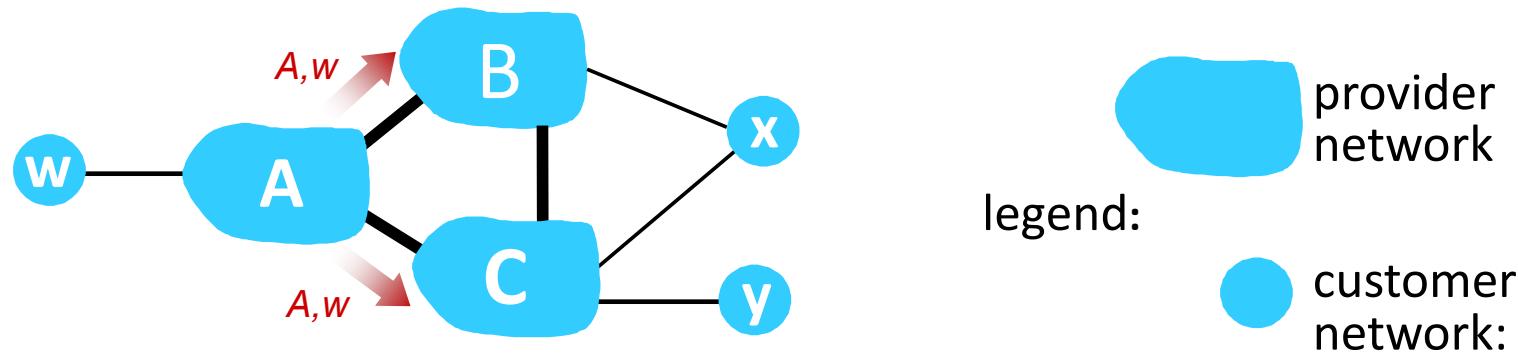
# BGP Path Advertisement: Multiple Paths



Gateway routers may learn about **multiple** paths to destination:

- AS1 gateway router 1c learns path ***AS2,AS3,X*** from 2a
- AS1 gateway router 1c learns path ***AS3,X*** from 3a
- Based on *policy*, AS1 gateway router 1c chooses path ***AS3,X*** and advertises path within AS1 via iBGP

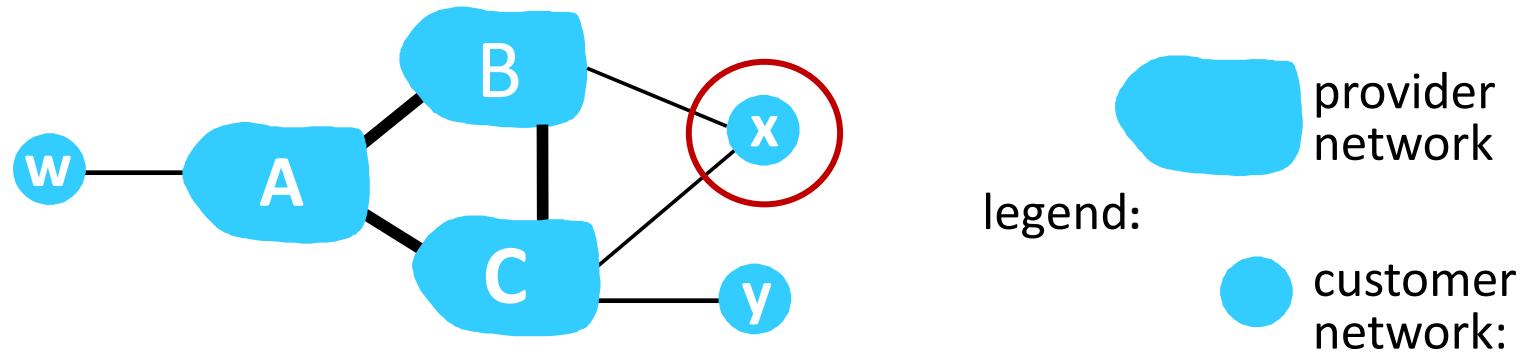
# BGP: Achieving Policy via Advertisements



ISP only wants to route traffic to/from its customer networks (does not want to carry transit traffic between other ISPs – a typical “real world” policy)

- A advertises path Aw to B and to C
- B *chooses not to advertise* BAw to C!
  - B gets no “revenue” for routing CBAw, since none of C, A, w is B's customer
  - C does *not* learn about CBAw path
- C will route CAw (not using B) to get to w

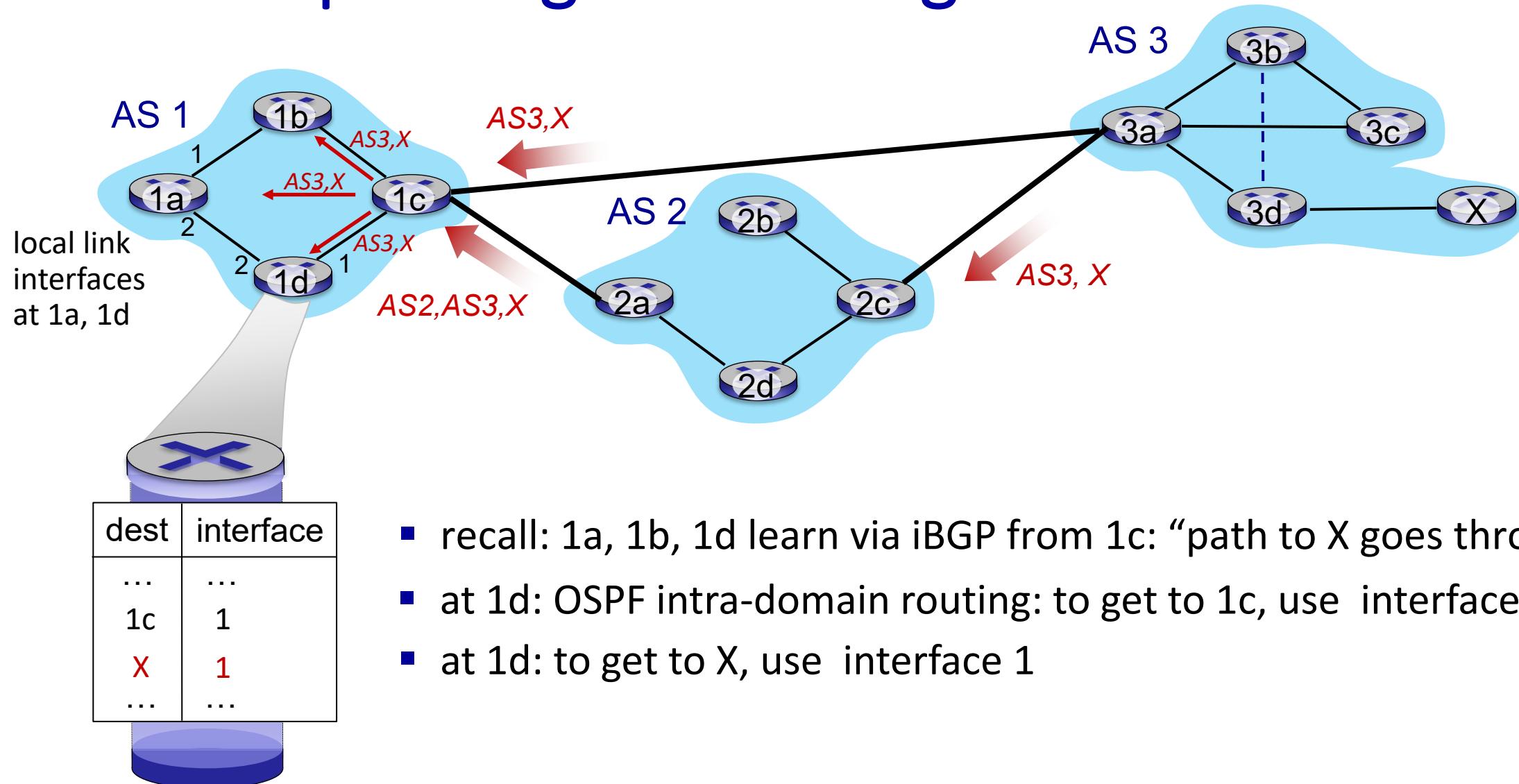
# BGP: Achieving Policy via Advertisements (more)



ISP only wants to route traffic to/from its customer networks (does not want to carry transit traffic between other ISPs – a typical “real world” policy)

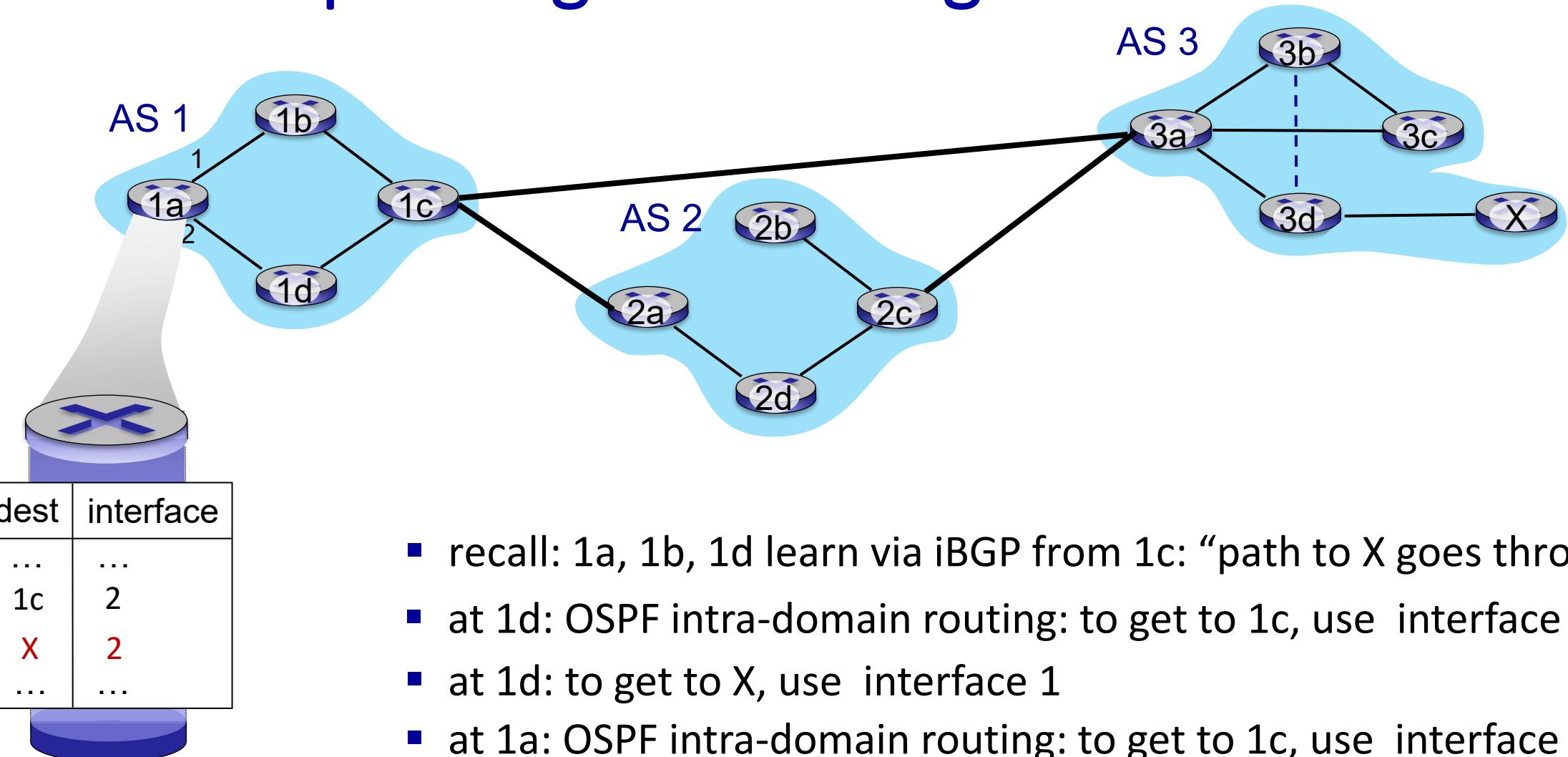
- A,B,C are **provider networks**
- x,w,y are **customer** (of provider networks)
- x is **dual-homed**: attached to two networks
- **policy to enforce**: x does not want to route from B to C via x
  - .. so x will not advertise to B a route to C

# BGP: Populating Forwarding Tables



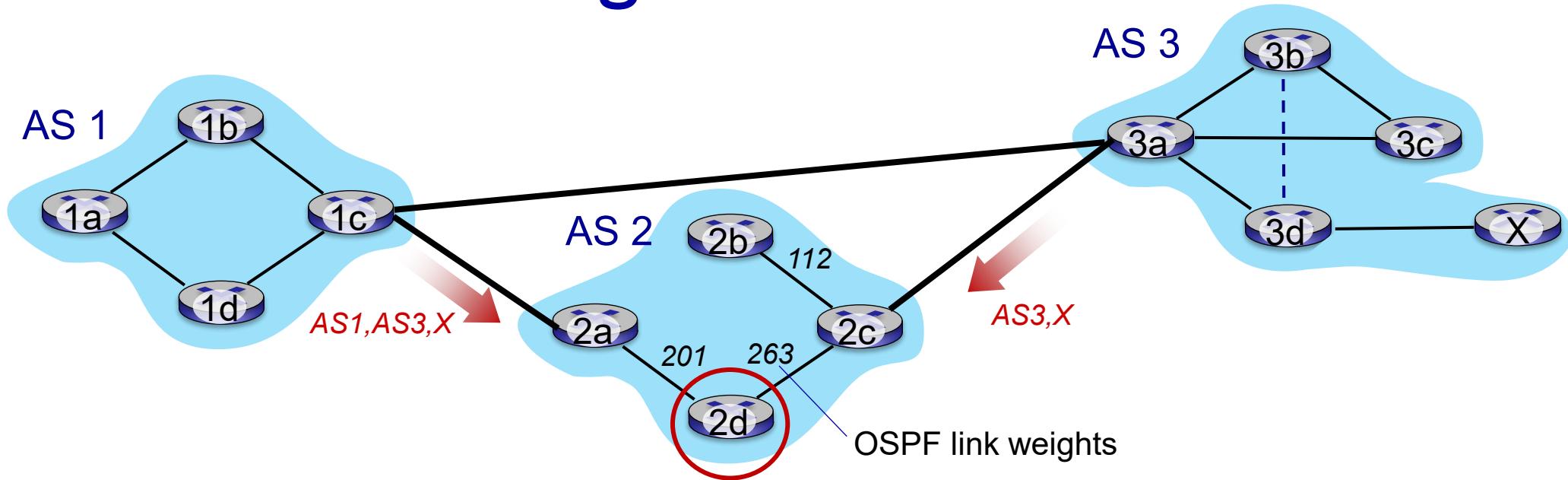
- recall: 1a, 1b, 1d learn via iBGP from 1c: “path to X goes through 1c”
- at 1d: OSPF intra-domain routing: to get to 1c, use interface 1
- at 1d: to get to X, use interface 1

# BGP: Populating Forwarding Tables



- recall: 1a, 1b, 1d learn via iBGP from 1c: “path to X goes through 1c”
- at 1d: OSPF intra-domain routing: to get to 1c, use interface 1
- at 1d: to get to X, use interface 1
- at 1a: OSPF intra-domain routing: to get to 1c, use interface 2
- at 1a: to get to X, use interface 2

# Hot Potato Routing



- 2d learns (via iBGP) it can route to X via 2a or 2c
- **hot potato routing:** choose local gateway that has least *intra-domain* cost (e.g., 2d chooses 2a, even though more AS hops to X): don't worry about inter-domain cost!

# Why Different Intra-, Inter-AS Routing ?

## policy:

- inter-AS: admin wants control over how its traffic is routed, who routes through its network
- intra-AS: single admin, so policy less of an issue

## scale: reducing forwarding table size, routing update traffic

- hierarchical routing: limiting the scope of full topological information
- BGP routing to CIDRized destination networks (summarized routes)

## performance:

- intra-AS: can focus on performance
- inter-AS: policy dominates over performance

# Network Layer: Control Plane

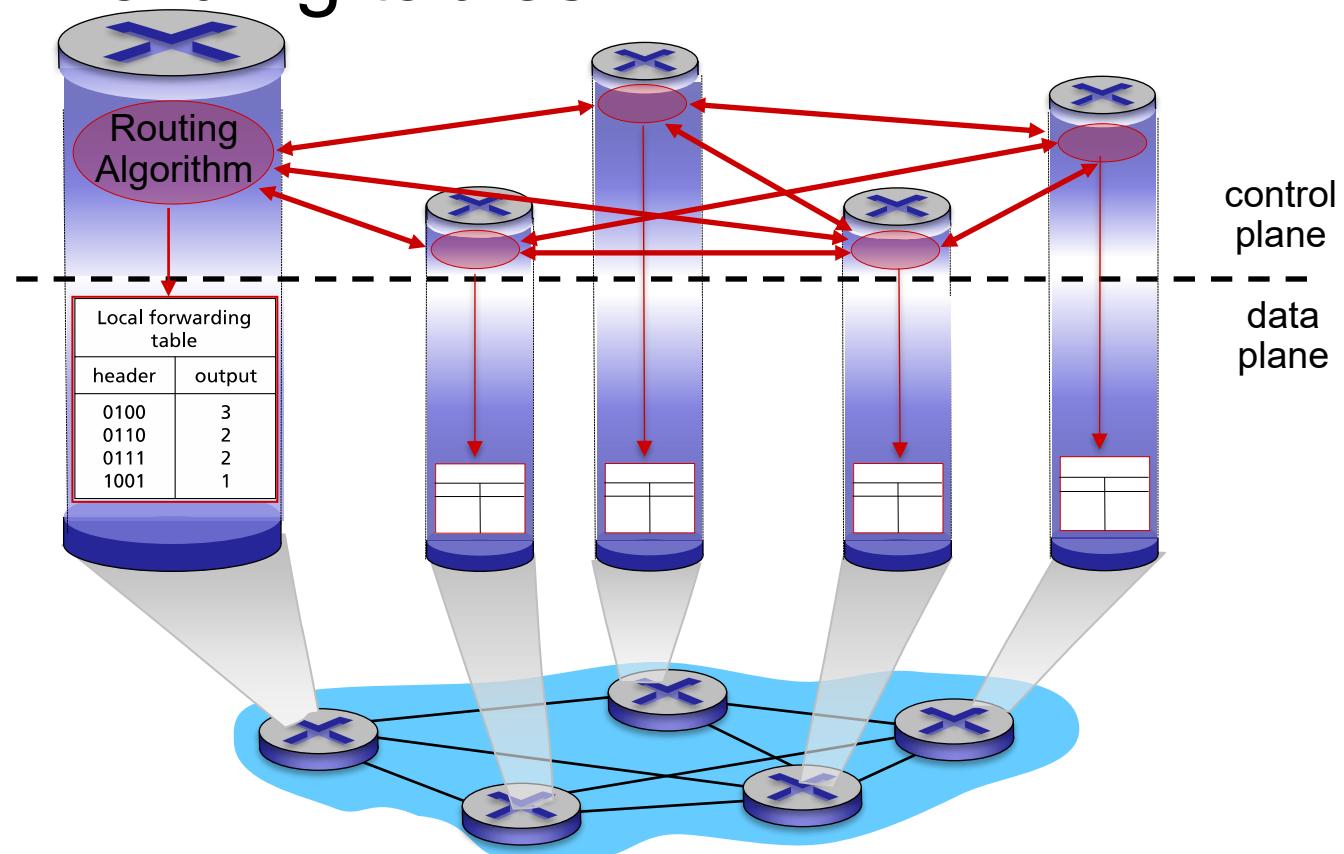
- intra-ISP routing: OSPF
- routing among ISPs: BGP
- **SDN control plane**
- Internet Control Message Protocol
- Network management, configuration

# Software Defined Networking (SDN)

- Internet network layer: historically has been implemented via distributed, per-router approach
  - *Monolithic* router contains switching hardware, runs proprietary implementation of Internet standard protocols (IP, RIP, IS-IS, OSPF, BGP) in proprietary router OS (e.g., Cisco IOS)
  - Different “middleboxes” for different network layer functions: firewalls, load balancers, NAT boxes, ..
- ~2005: Renewed interest in rethinking network control plane

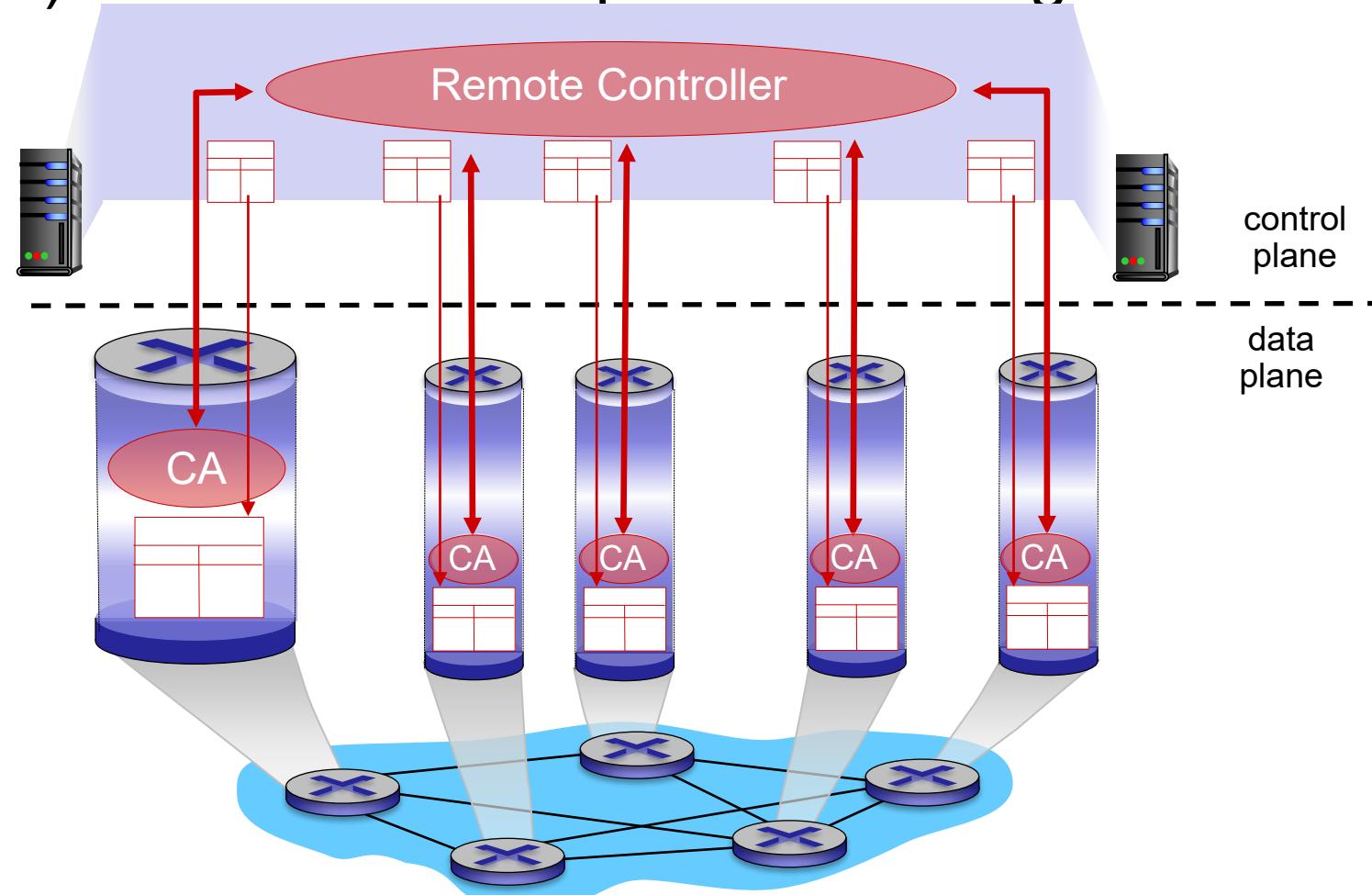
# Per-router Control Plane

- Individual routing algorithm components *in each and every router* interact with each other in control plane to compute forwarding tables

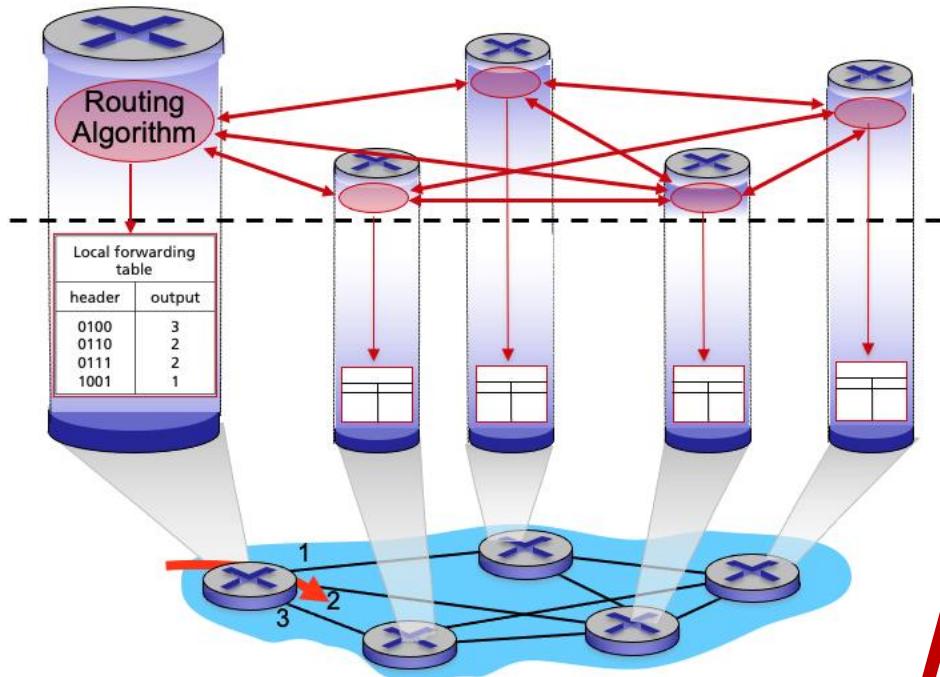


# Logically Centralized Control Plane

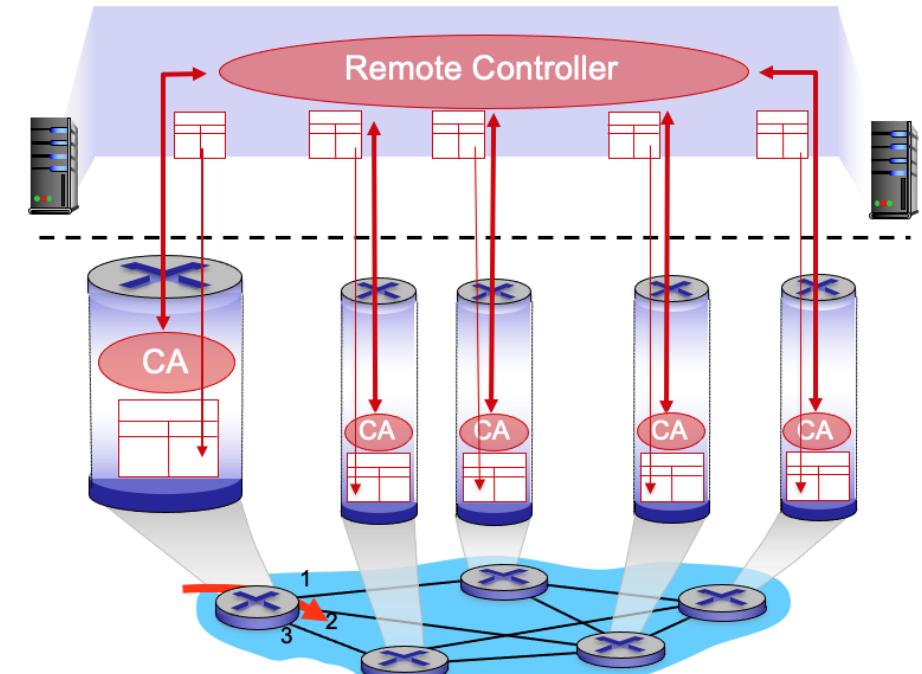
- A distinct (typically remote) controller interacts with local control agents (CAs) in routers to compute forwarding tables



# Per-router control plane



# SDN control plane

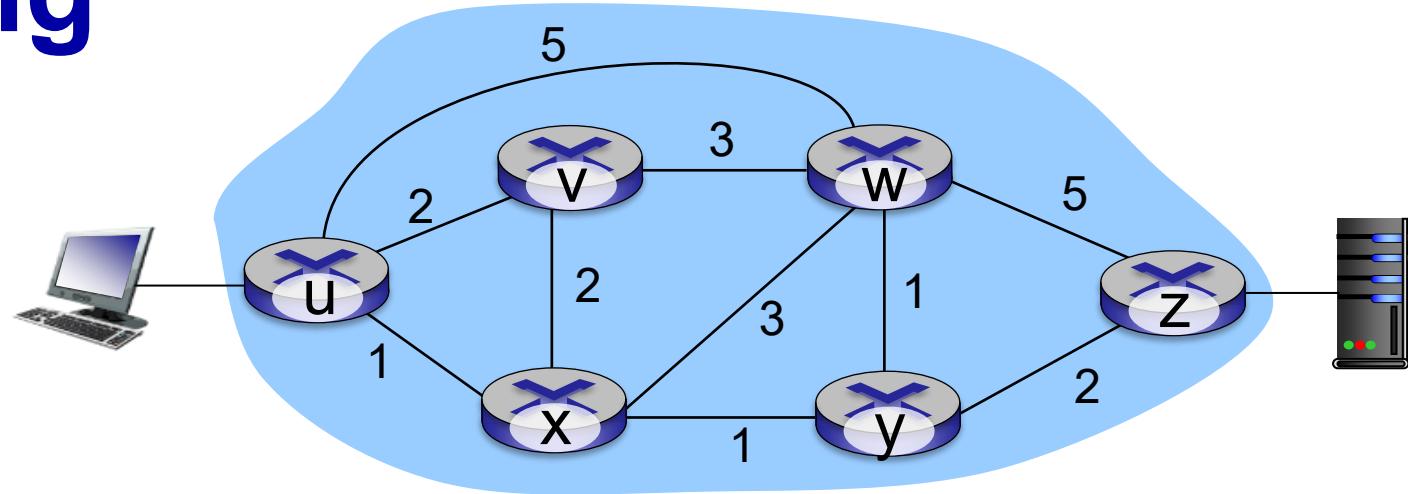


# Software Defined Networking (SDN)

*Why* a *logically centralized* control plane?

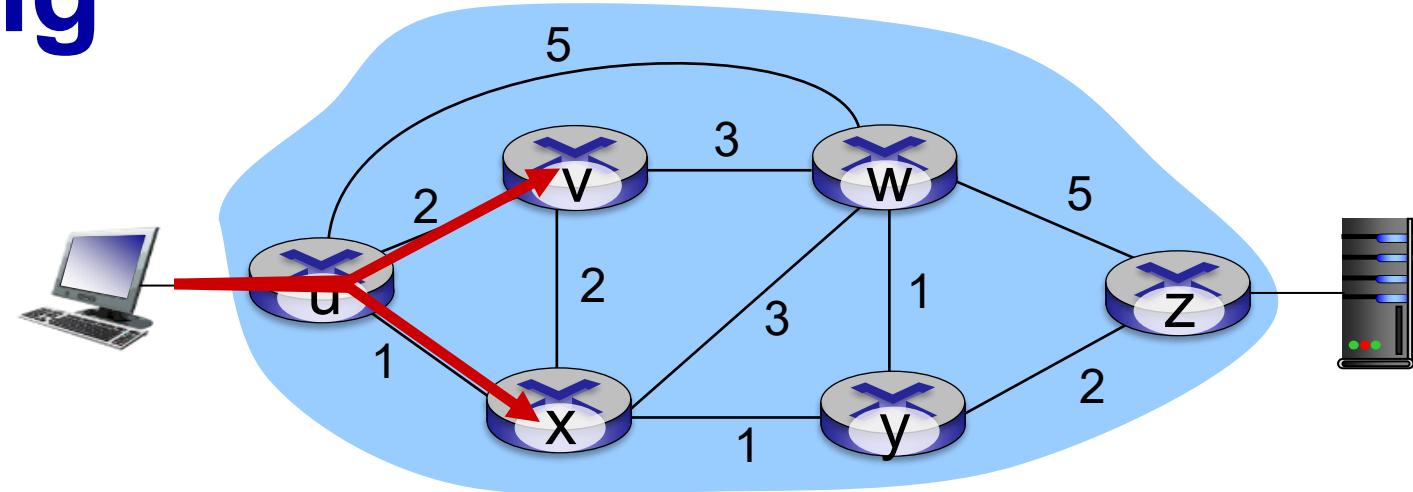
- Easier network management: avoid router misconfigurations, greater flexibility of traffic flows
- Table-based forwarding (recall OpenFlow API) allows “programming” routers
  - Centralized “programming” easier: compute tables centrally and distribute
  - Distributed “programming” more difficult: compute tables as result of distributed algorithm (protocol) implemented in each and every router
- Open (non-proprietary) implementation of control plane

# Traffic Engineering: Difficult Traditional Routing



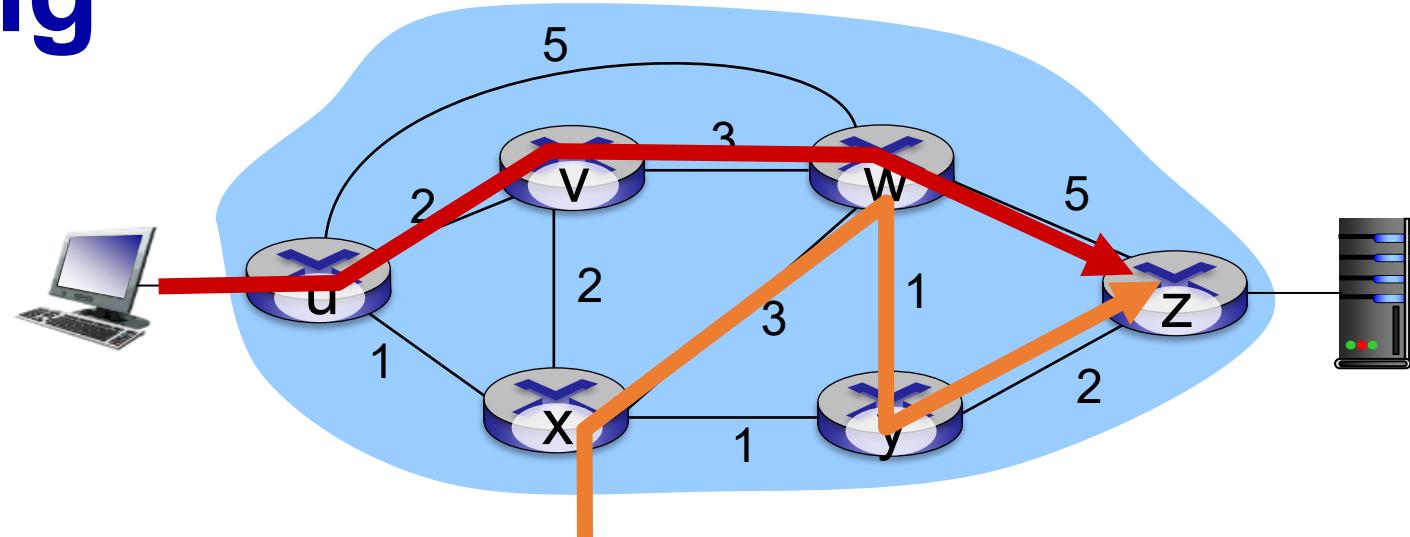
- Q: what if network operator wants u-to-z traffic to flow along *uvwz*, x-to-z traffic to flow along *xwyz*?

# Traffic Engineering: Difficult Traditional Routing



- **Q:** what if network operator wants to split  $u\text{-to-}z$  traffic along  $uvwz$  ***and***  $uxyz$  (load balancing)?

# Traffic Engineering: Difficult Traditional Routing



- **Q:** what if w wants to route red and orange traffic differently?

# Key Characteristics of SDN Architecture

**4. programmable control applications**

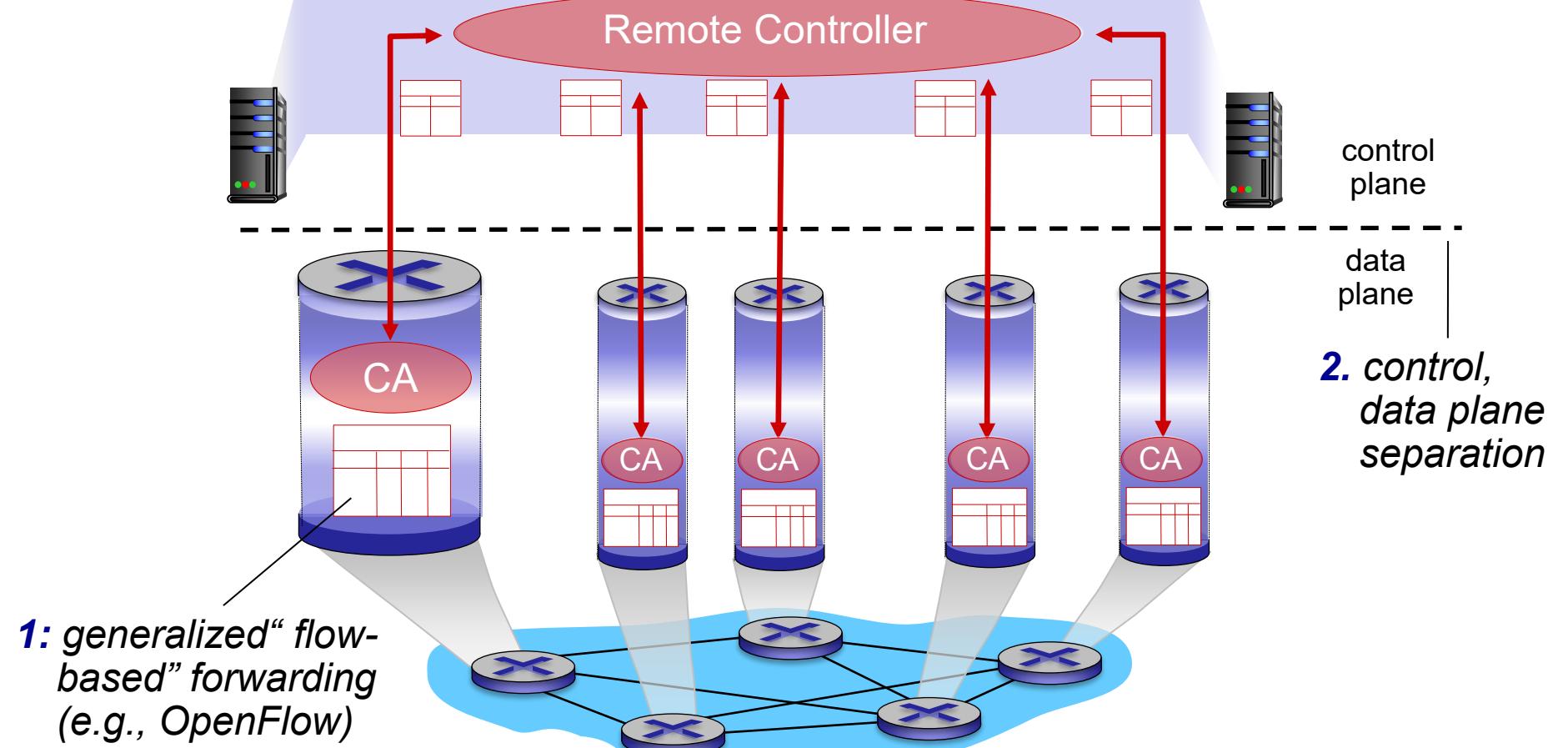
routing

access control

...

load balance

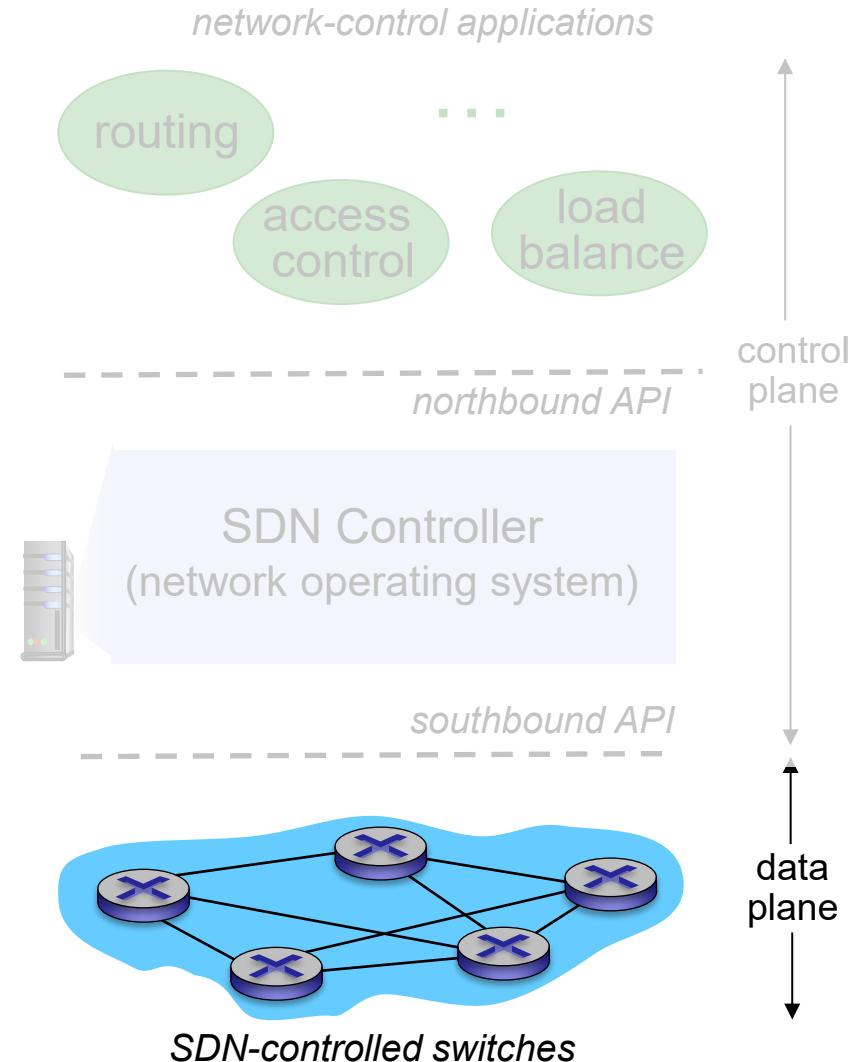
**3. control plane functions (software) external to data-plane switches**



# SDN Perspective: Data Plane Switches

## Data plane switches

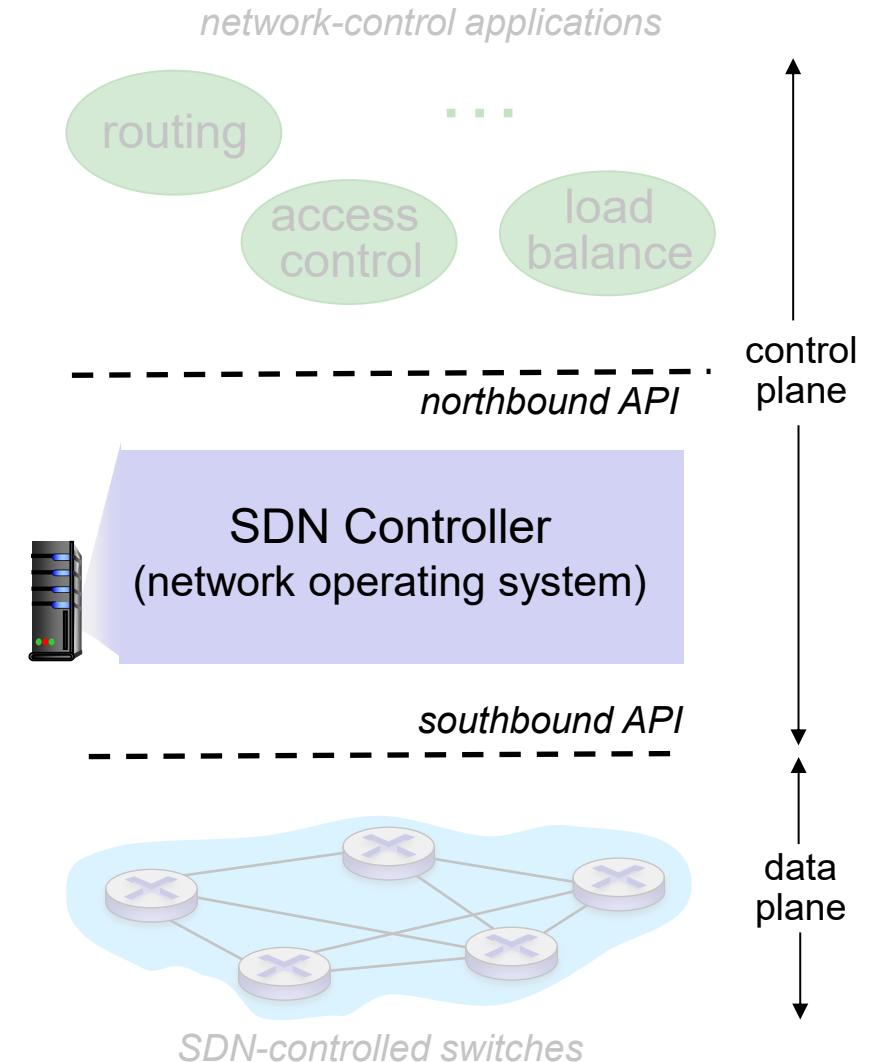
- Fast, simple, commodity switches implementing generalized data-plane forwarding in hardware
- Switch flow table computed, installed by controller
- API for table-based switch control (e.g., OpenFlow)
  - defines what is controllable and what is not
- Protocol for communicating with controller (e.g., OpenFlow)



# SDN Perspective: SDN Controller

## ***SDN controller (network OS):***

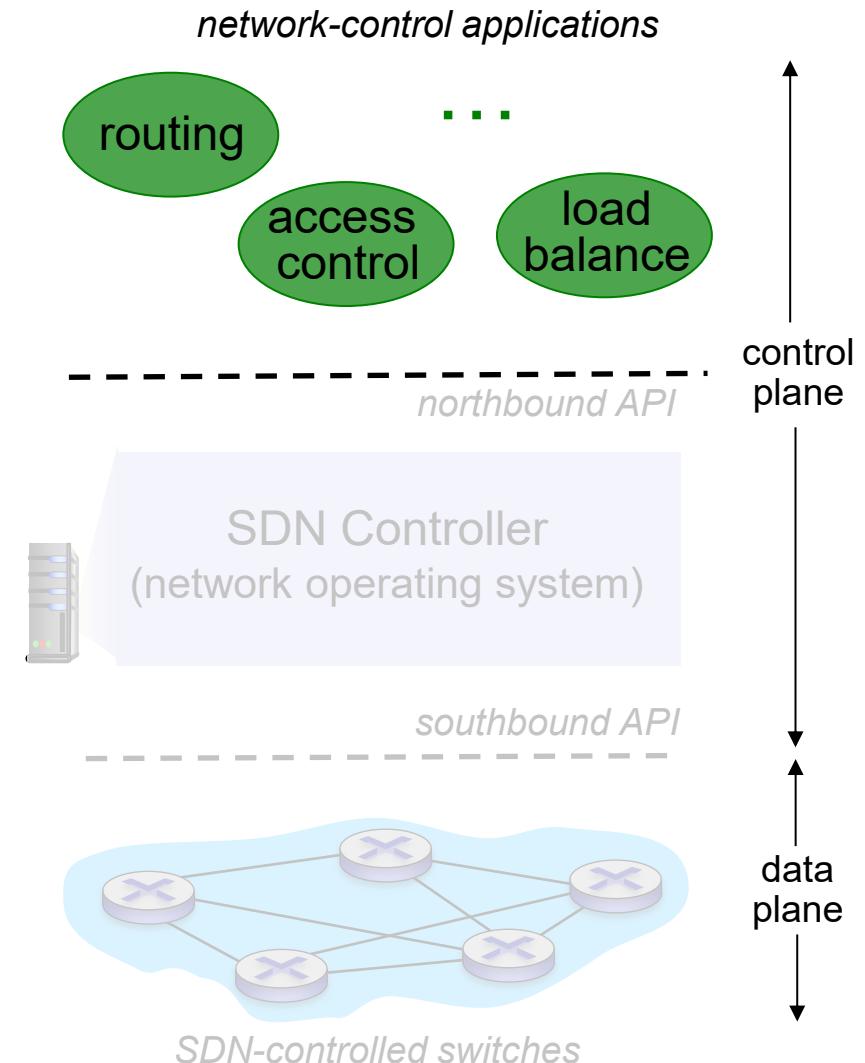
- Maintains network state information
- Interacts with network control applications “above” via northbound API
- Interacts with network switches “below” via southbound API
- Implemented as distributed system for performance, scalability, fault-tolerance, robustness



# SDN Perspective: Control Applications

## ***Network-control apps:***

- “**brains**” of control: implement control functions using lower-level services, API provided by SDN controller
- **Open**: can be provided by 3<sup>rd</sup> party: distinct from routing vendor, or SDN controller

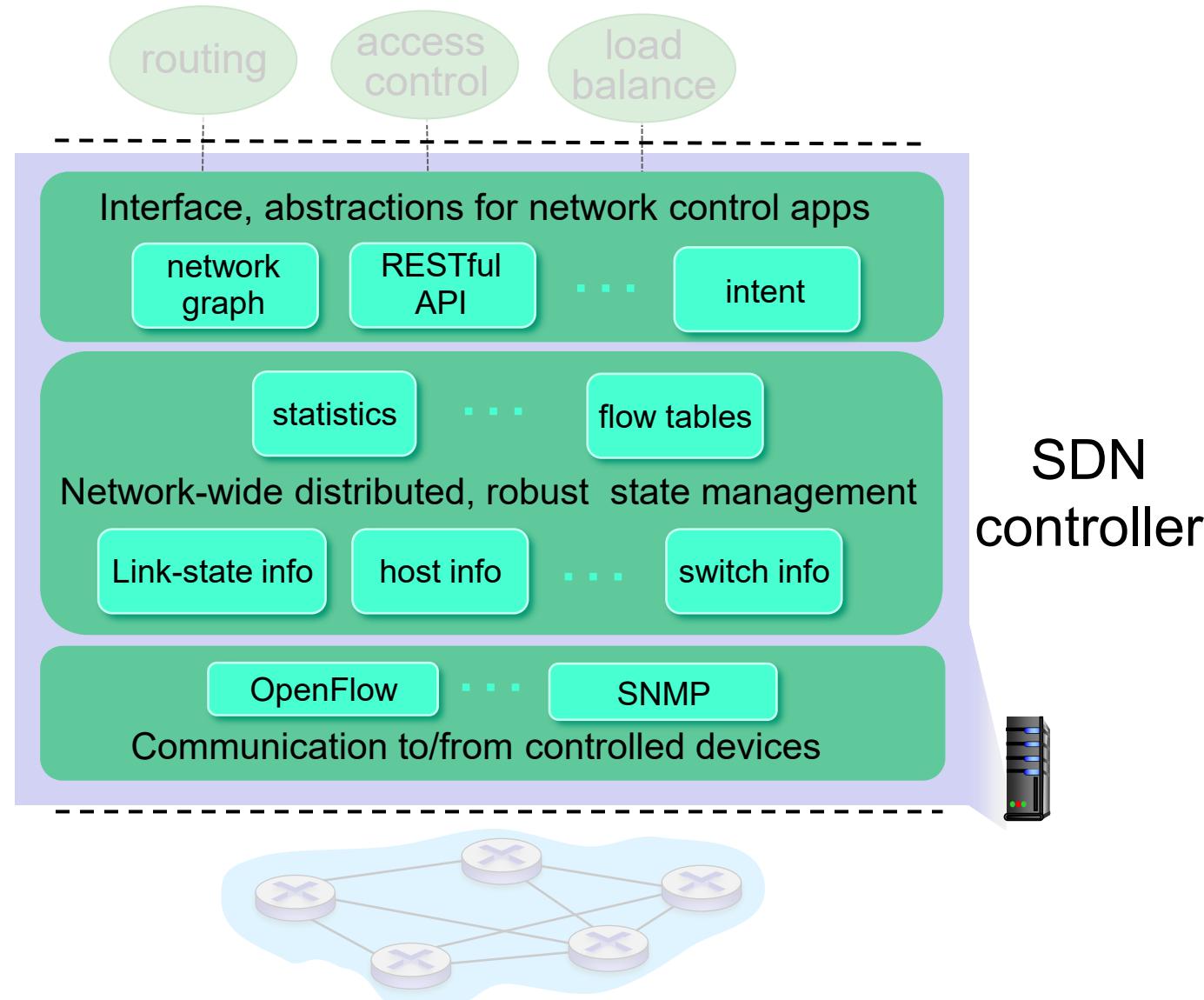


# Components of SDN Controller

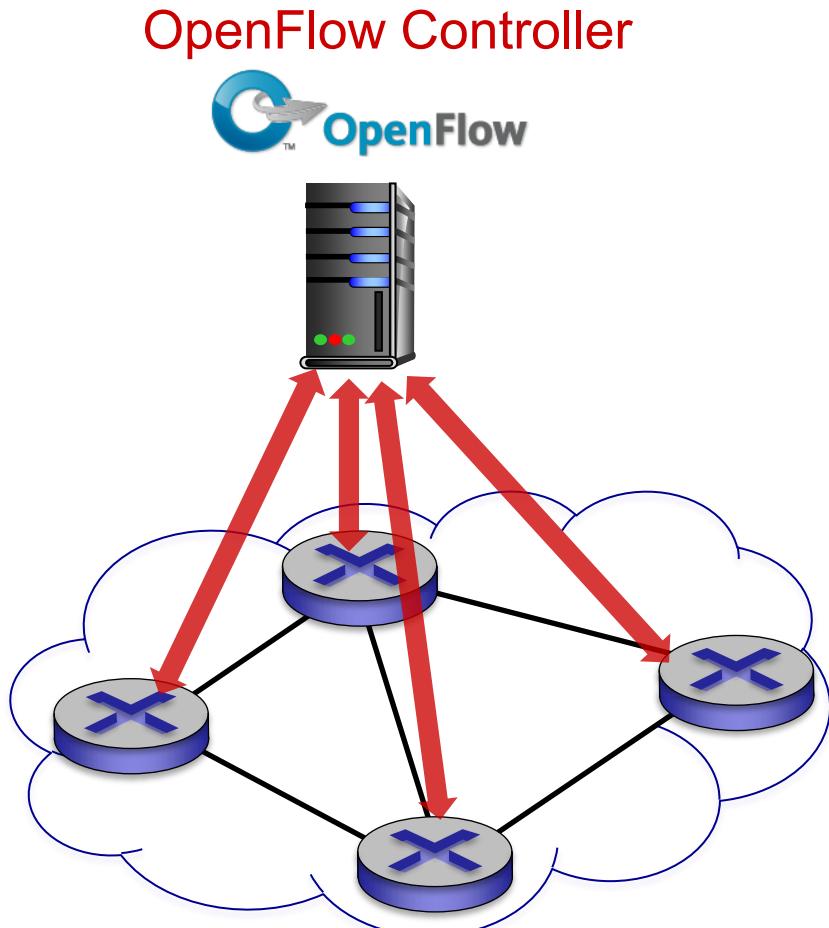
Interface layer to network control apps: abstractions API

Network-wide state management layer: state of networks links, switches, services: a *distributed database*

*communication layer:* communicate between SDN controller and controlled switches



# OpenFlow Protocol

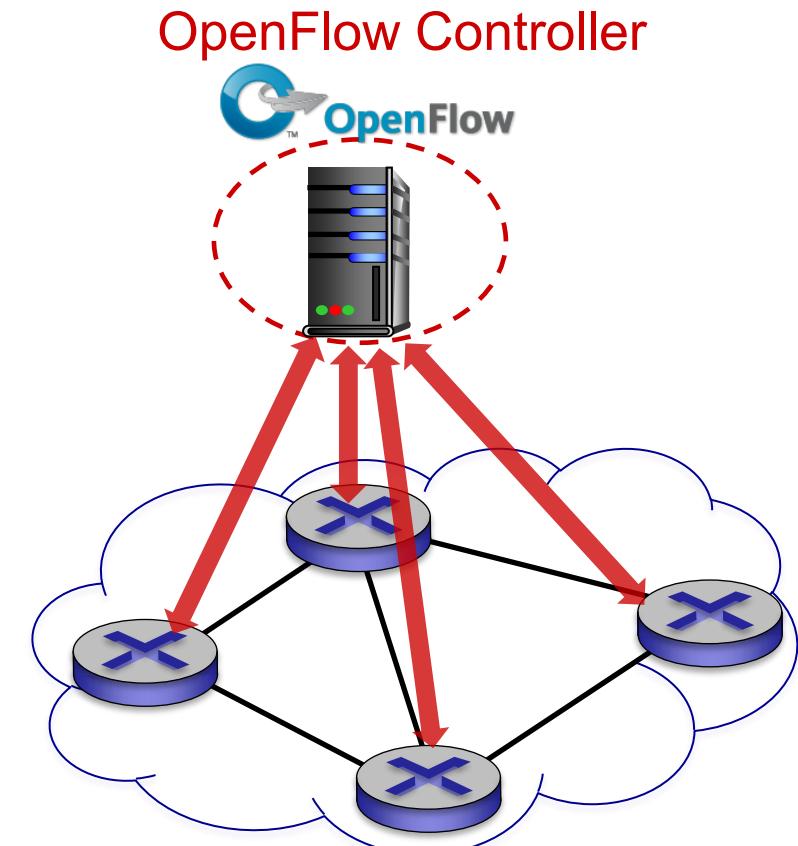


- Operates between controller, switch
- TCP used to exchange messages
  - optional encryption
- Three classes of OpenFlow messages:
  - controller-to-switch
  - asynchronous (switch to controller)
  - symmetric (misc)

# OpenFlow: Controller-to-Switch Messages

## *Key controller-to-switch messages*

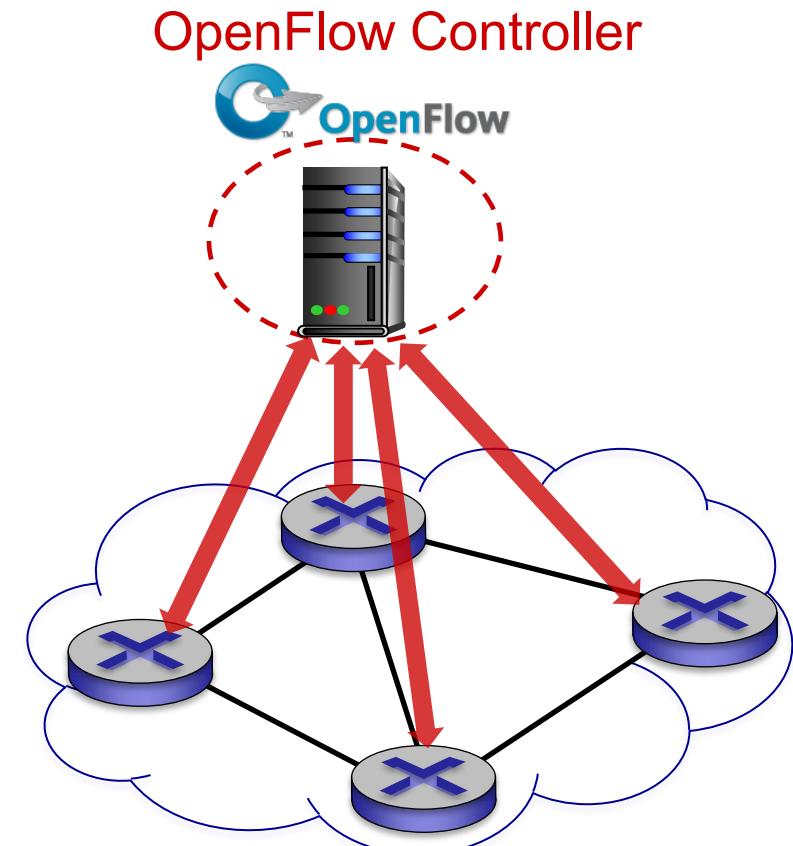
- **Features:** controller queries switch features, switch replies
- **Configure:** controller queries/sets switch configuration parameters
- **Modify-state:** add, delete, modify flow entries in the OpenFlow tables
- **Packet-out:** controller can send this packet out of specific switch port



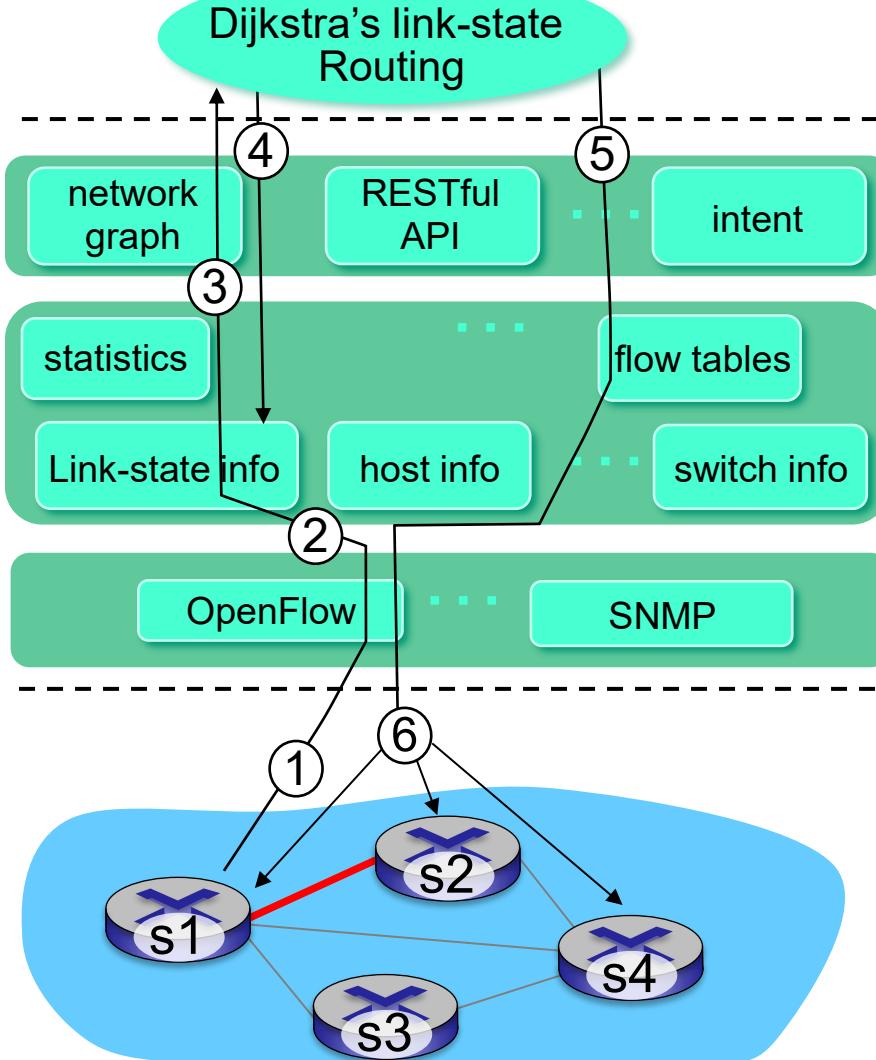
# OpenFlow: Switch-to-Controller Messages

## *Key switch-to-controller messages*

- **Packet-in:** transfer packet (and its control) to controller. See packet-out message from controller
- **Flow-removed:** flow table entry deleted at switch
- **Port status:** inform controller of a change on a port.



# SDN: Control/Data Plane Interaction Example



- ① S1, experiencing link failure using OpenFlow port status message to notify controller
- ② SDN controller receives OpenFlow message, updates link status info
- ③ Dijkstra's routing algorithm application has previously registered to be called whenever link status changes. It is called.
- ④ Dijkstra's routing algorithm access network graph info, link state info in controller, computes new routes
- ⑤ link state routing app interacts with flow-table-computation component in SDN controller, which computes new flow tables needed
- ⑥ Controller uses OpenFlow to install new tables in switches that need updating

# Network Layer: Control Plane

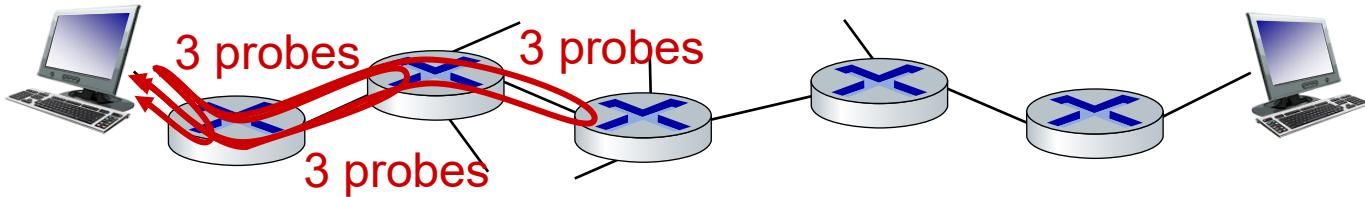
- intra-ISP routing: OSPF
- routing among ISPs: BGP
- SDN control plane
- **Internet Control Message Protocol**
- Network management, configuration

# ICMP: Internet Control Message Protocol

- used by hosts and routers to communicate network-level information
  - error reporting: unreachable host, network, port, protocol
  - echo request/reply (used by ping)
- network-layer “above” IP:
  - ICMP messages carried in IP datagrams, protocol number: 1
- *ICMP message*: type, code plus header and first 8 bytes of IP datagram causing error

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 and ICMP



- source sends sets of UDP segments to destination
  - 1<sup>st</sup> set has TTL =1, 2<sup>nd</sup> set has TTL=2, etc.
- datagram in *n*th set arrives to *n*th router:
  - router discards datagram and sends source ICMP message (type 11, code 0)
  - IP address of router where TTL expired is source IP address of datagram containing this ICMP message
- when ICMP message arrives at source: record RTTs

## Stopping criteria:

- UDP segment eventually arrives at destination host
- destination returns ICMP “port unreachable” message (type 3, code 3)
- source stops

# Network Layer: Control Plane

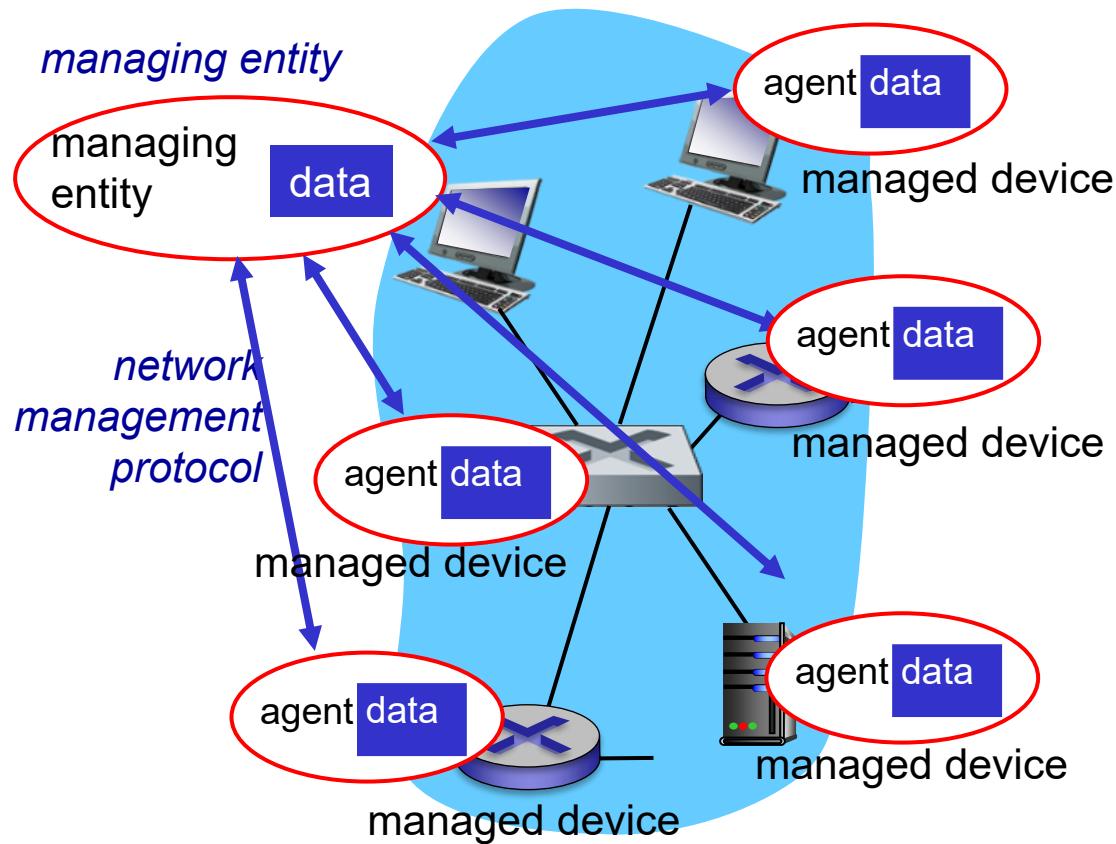
- intra-ISP routing: OSPF
- routing among ISPs: BGP
- SDN control plane
- Internet Control Message Protocol
- Network management, configuration

# What is Network Management?

- **Autonomous systems (aka “network”): 1000s of interacting hardware/software components**
- **Other complex systems requiring monitoring, control:**
  - Jet airplane
  - Nuclear power plant
  - others?

"Network management includes the deployment, integration and coordination of the hardware, software, and human elements to monitor, test, poll, configure, analyze, evaluate, and control the network and element resources to meet the real-time, operational performance, and Quality of Service requirements at a reasonable cost."                    -- Saydam 1996

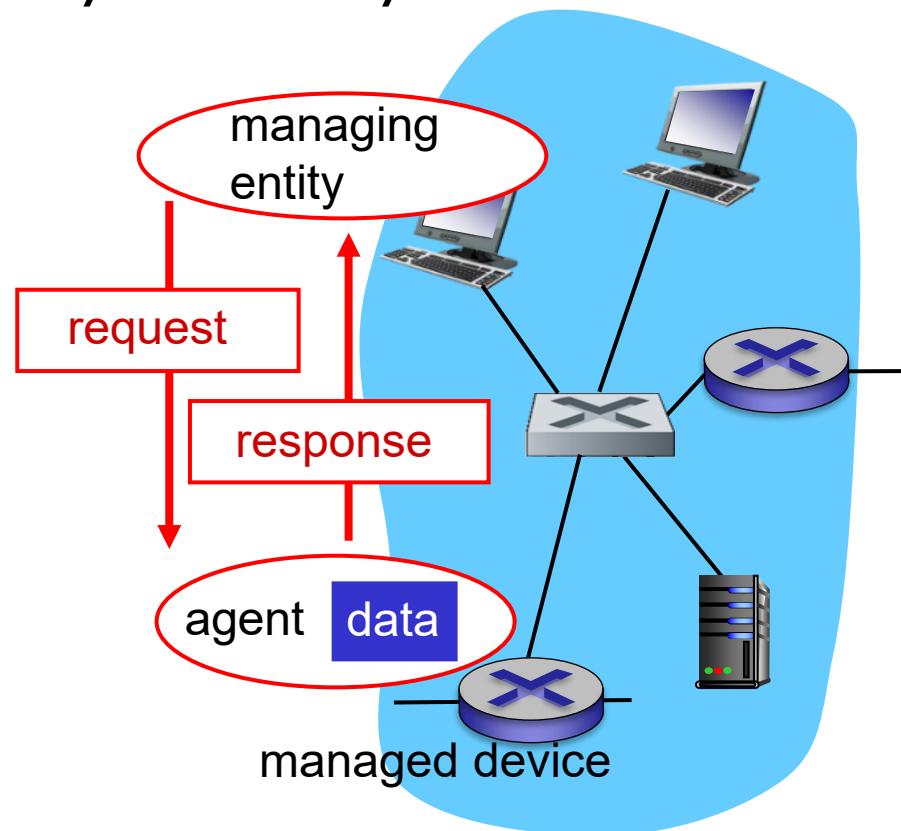
# Infrastructure for Network Management



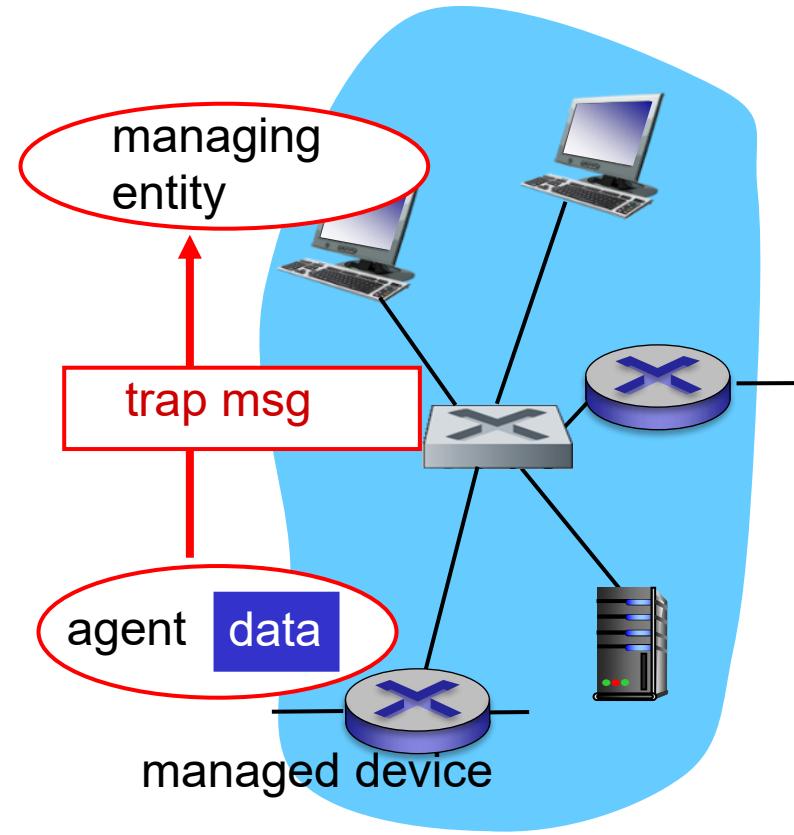
*Managed devices contain managed objects whose data is gathered into a **Management Information Base (MIB)***

# SNMP (Simple Network Management Protocol)

Two ways to convey MIB info, commands:



request/response mode



trap mode

# SNMP Protocol: Message Types

<u>Message type</u>	<u>Function</u>
GetRequest GetNextRequest GetBulkRequest	manager-to-agent: “get me data” (data instance, next data in list, block of data)
InformRequest	manager-to-manager: here’s MIB value
SetRequest	manager-to-agent: set MIB value
Response	Agent-to-manager: value, response to Request
Trap	Agent-to-manager: inform manager of exceptional event

# SNMP Protocol: Message Formats

