南京大学
NANJING UNIVERSITY

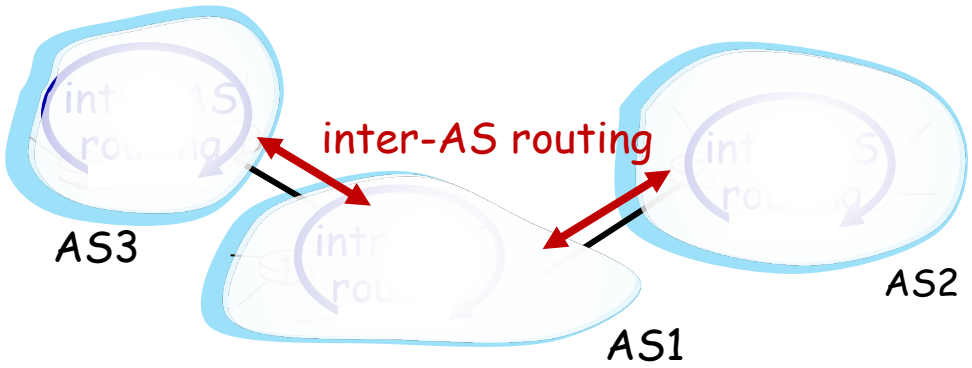
网络层: 控制平面

殷亚凤
智能软件与工程学院
苏州校区南雍楼东区225
yafeng@nju.edu.cn，https://yafengnju.github.io/

Outline

- Routing among ISPs: BGP
- SDN control plane
- Internet Control Message Protocol
- Network management, configuration

Interconnected ASes



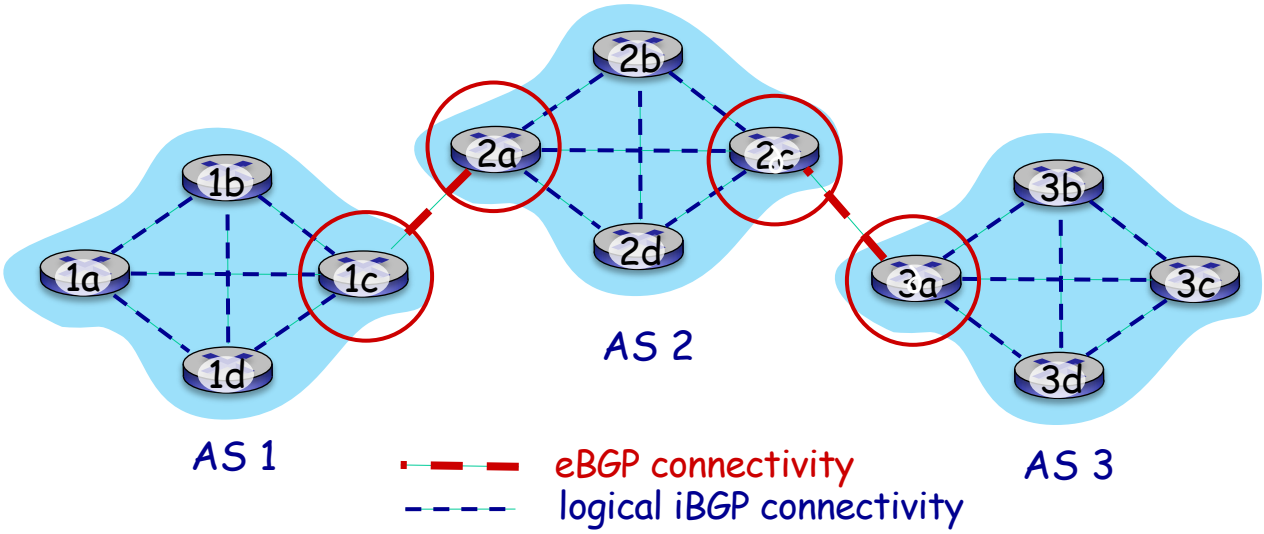
intra-AS (aka "intra-domain"): routing among routers within same AS ("network")

inter-AS (aka "inter-domain"): routing among AS'es

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 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

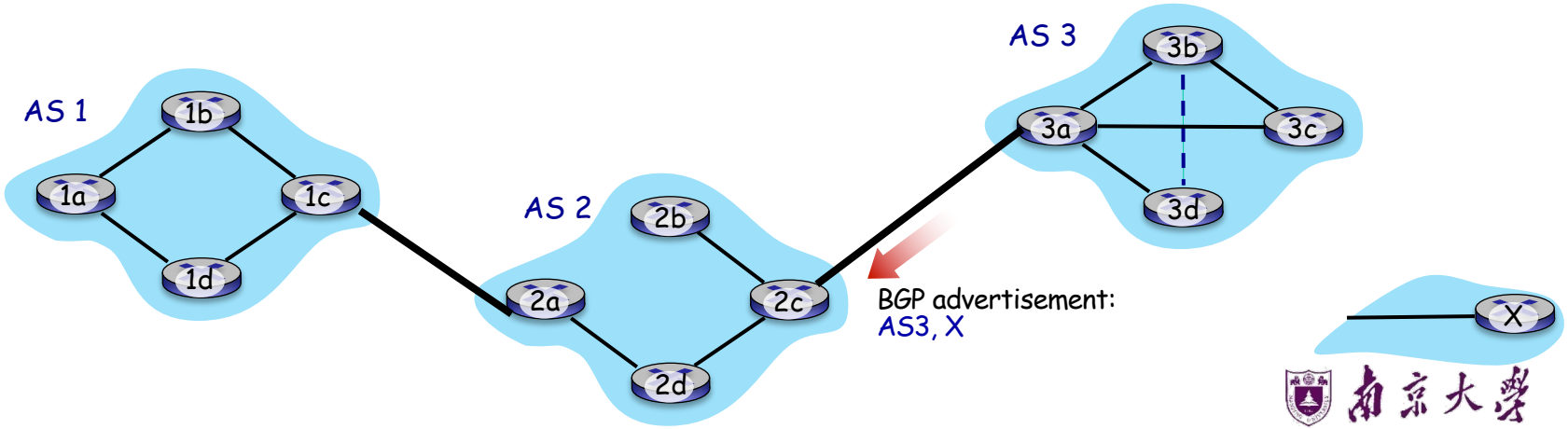
eBGP, iBGP connections



gateway routers run both eBGP and iBGP protocols

BGP basics

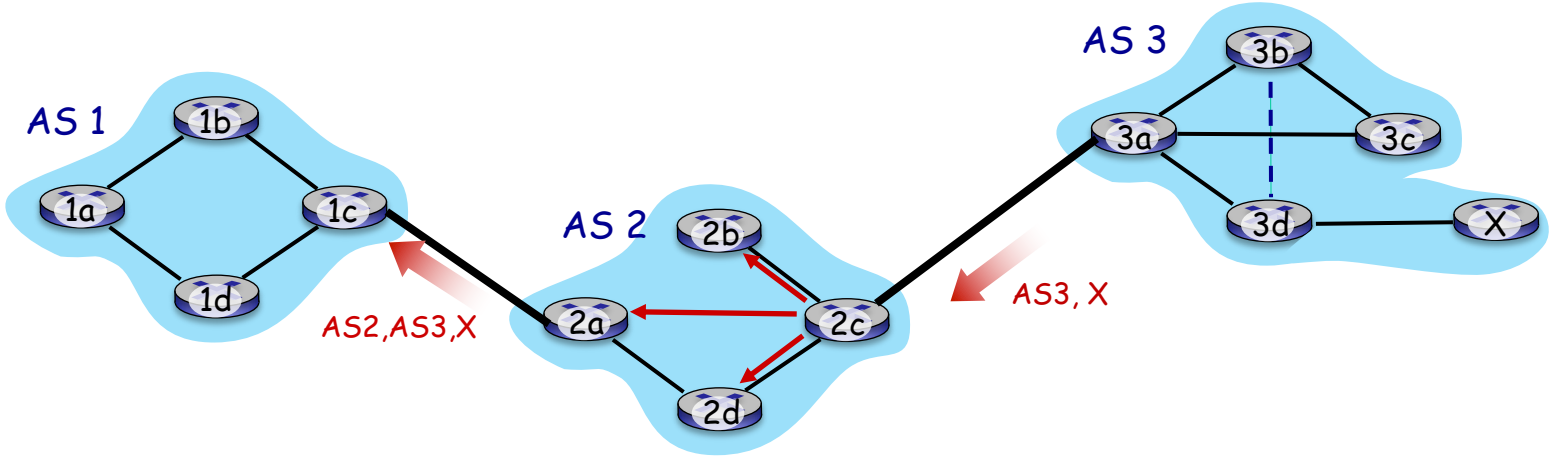
- BGP session: two BGP routers ("peers") exchange BGP messages over semi-permanent TCP connection:
  - advertising paths to different destination network prefixes (BGP is a "path vector" protocol)
- when AS3 gateway 3a advertises path AS3,X to AS2 gateway 2c:
  - AS3 promises to AS2 it will forward datagrams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



Path attributes and BGP routes

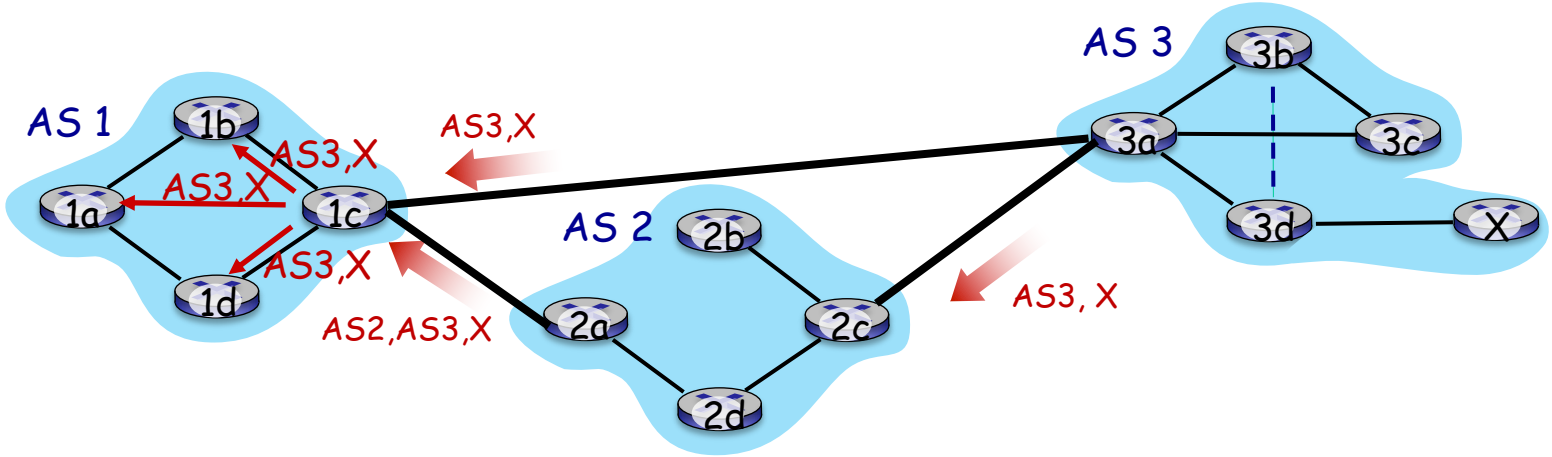
- BGP advertised route: prefix + attributes
  - prefix: destination being advertised
  - two important attributes:
  - AS-PATH: list of ASes through which prefix advertisement has passed
  - NEXT-HOP: indicates specific internal-AS router to next-hop AS
- policy-based routing:
  - gateway receiving route 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 other neighboring ASes

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

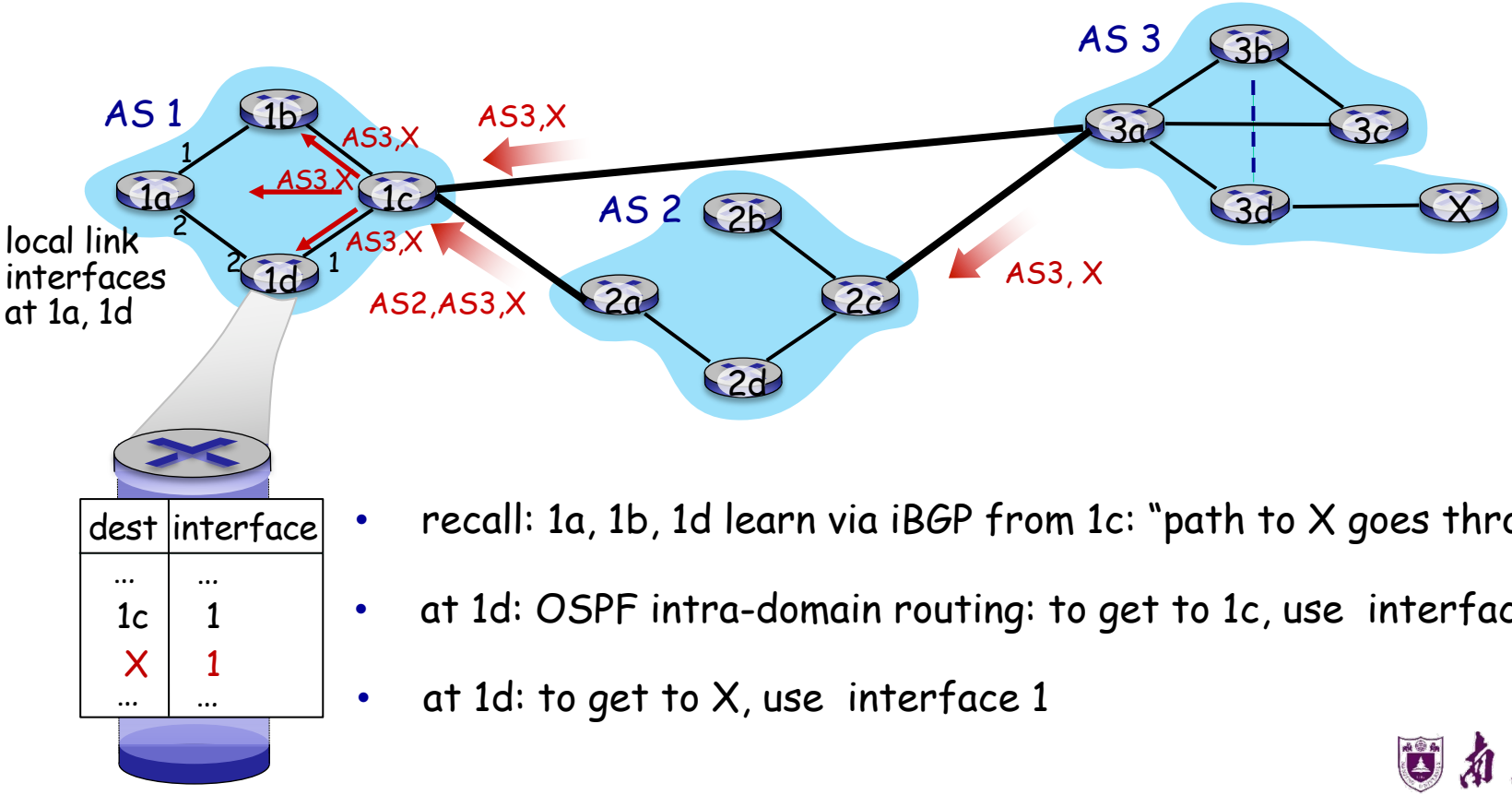
BGP path advertisement: multiple paths



gateway router may learn about multiple paths to destination:

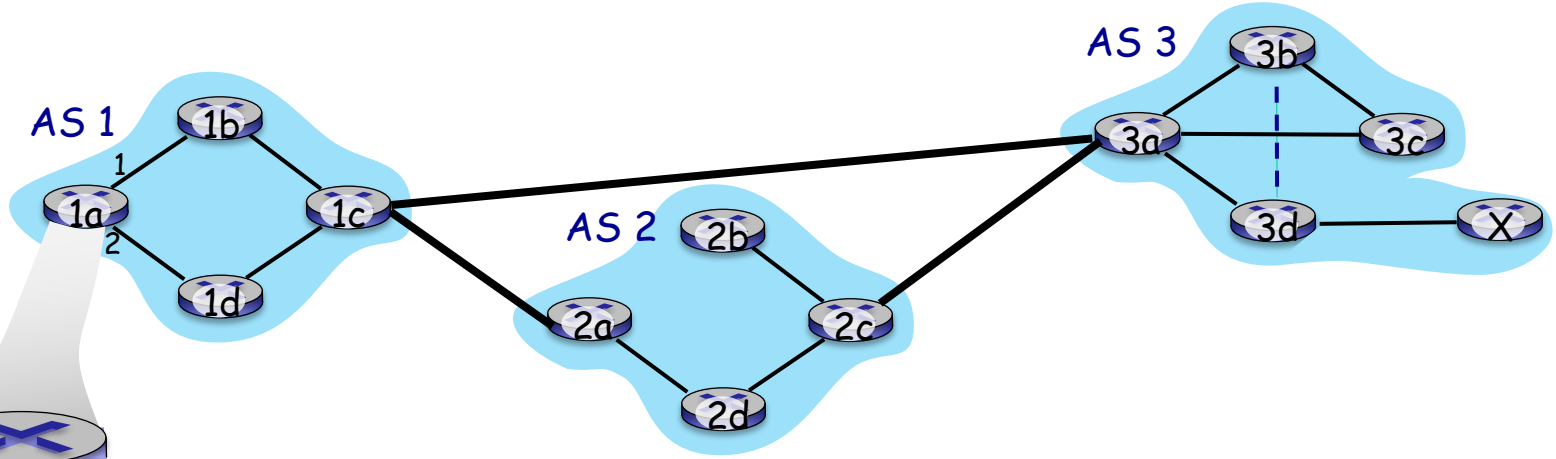
- AS1 gateway router 1c learns path AS2,AS3,X from 2a
- AS1 gateway router 1c learns path AS3,X from 3a
- based on policy, AS1 gateway router 1c chooses path AS3,X and advertises path within AS1 via iBGP

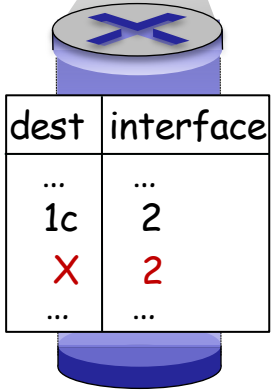
BGP: populating forwarding tables



- 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

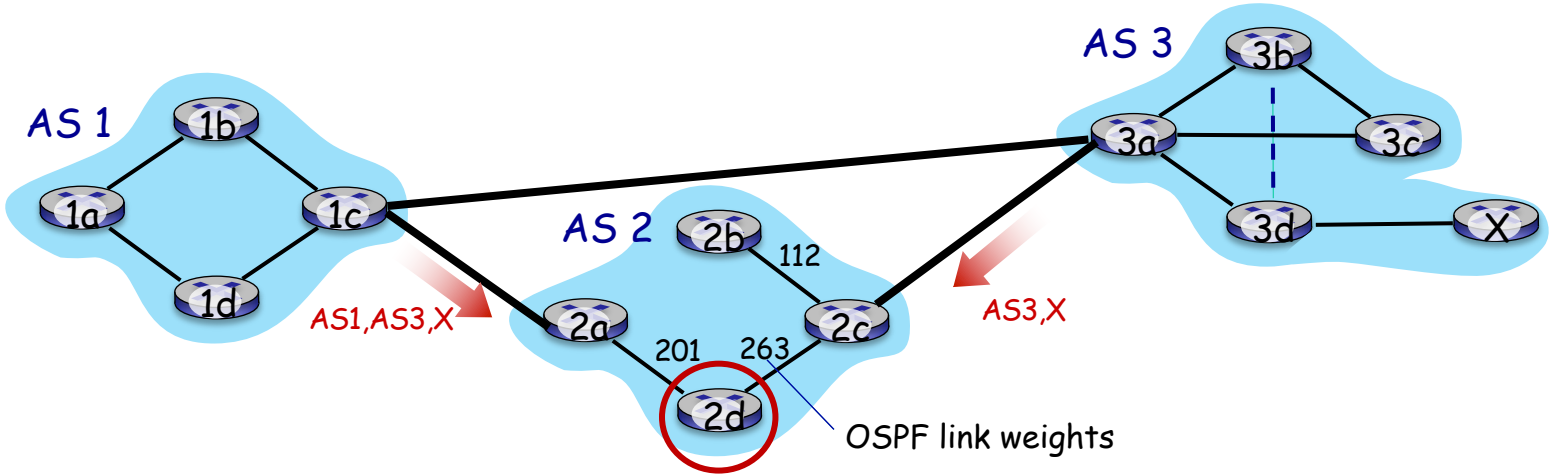
BGP: populating forwarding tables





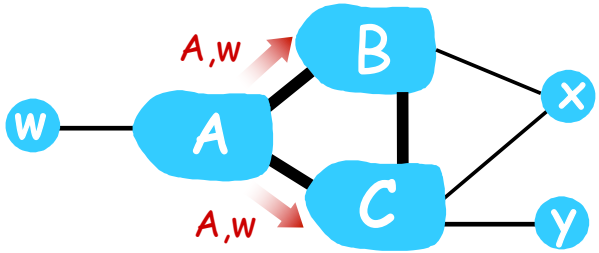
- recall: 1a, 1b, 1d learn via iBGP from 1c: "path to X goes through 1c"
- at 1d: OSPF intra-domain routing: to get to 1c, use interface 1
- at 1d: to get to X, use interface 1
- at 1a: OSPF intra-domain routing: to get to 1c, use interface 2
- at 1a: to get to X, use interface 2

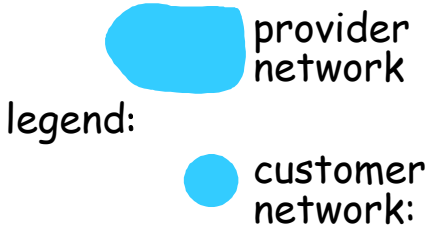
Hot potato routing



- 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!

BGP: achieving policy via advertisements

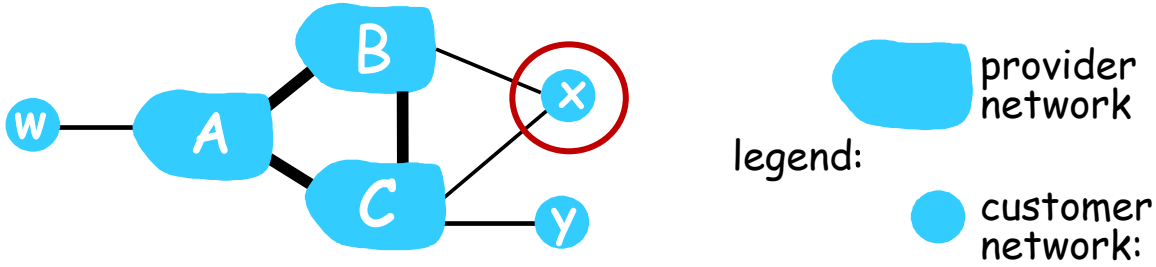




ISP only wants to route traffic to/from its customer networks (does not want to carry transit traffic between other ISPs – a typical "real world" policy)

- 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 are B's customers
  - 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

Why different Intra-, Inter-AS routing ?

policy:

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

scale:

- hierarchical routing saves table size, reduced update traffic

performance:

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

BGP route selection

- router may learn about more than one route to destination AS, selects route based on:
  1. local preference value attribute: policy decision
  2. shortest AS-PATH
  3. closest NEXT-HOP router: hot potato routing
  4. additional criteria

Outline

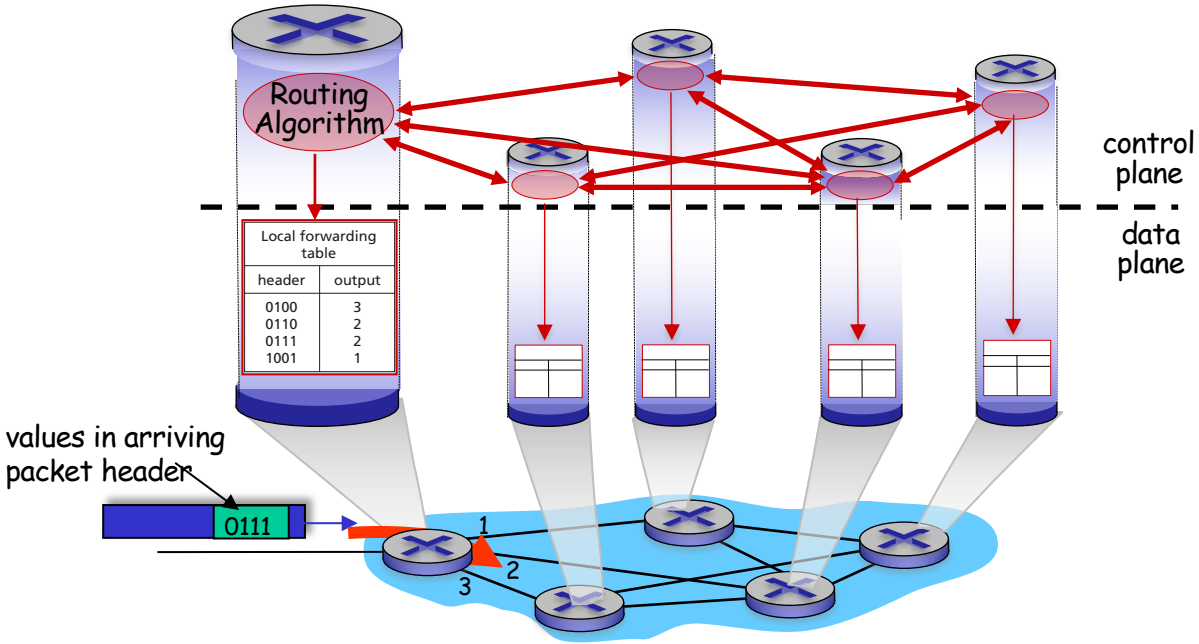
- Routing among ISPs: BGP
- SDN control plane
- Internet Control Message Protocol
- Network management, configuration

Software defined networking (SDN)

- Internet network layer: historically implemented via distributed, per-router control 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

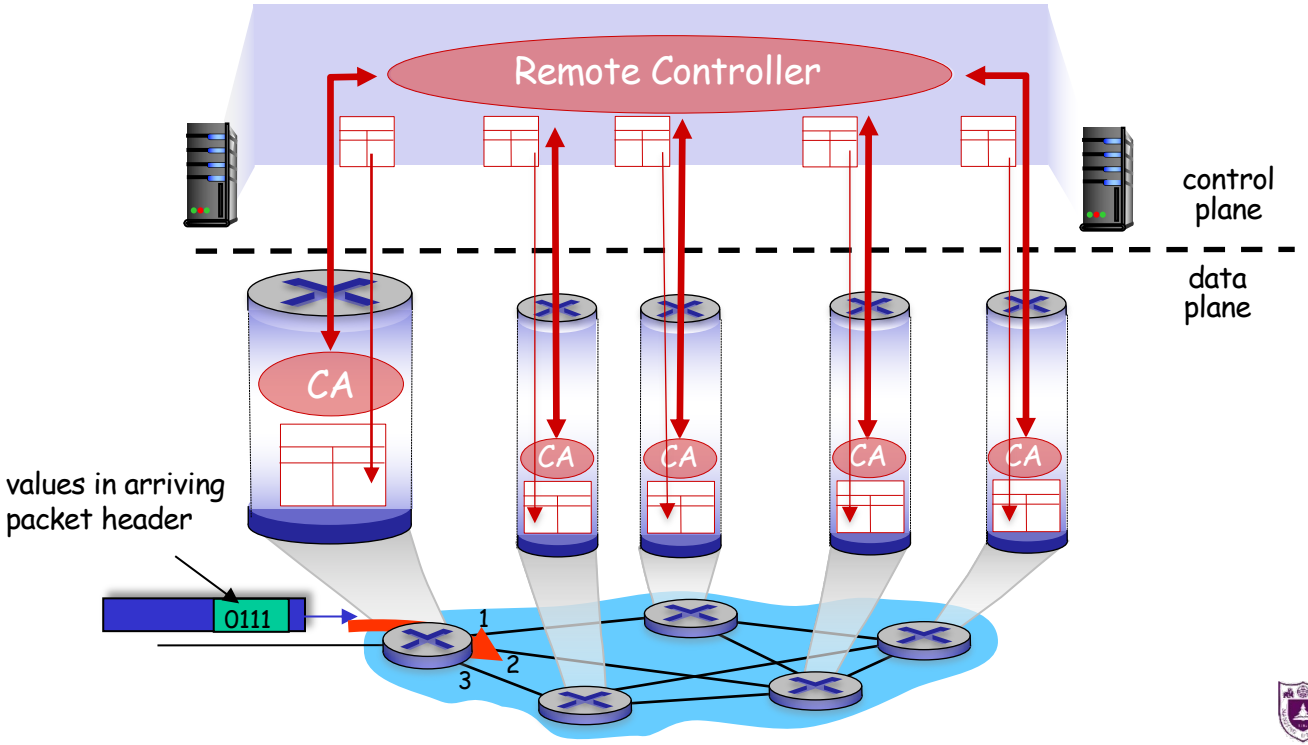
Per-router control plane

Individual routing algorithm components in each and every router interact in the control plane to computer forwarding tables



Software-Defined Networking (SDN) control plane

Remote controller computes, installs forwarding tables in routers



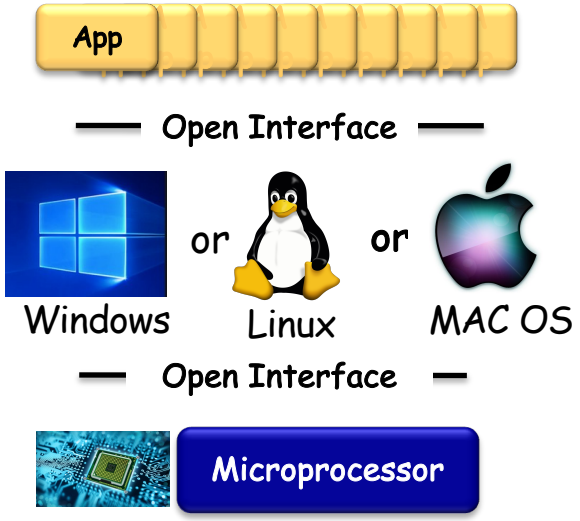
Software defined networking (SDN)

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
  - foster innovation: let 1000 flowers bloom

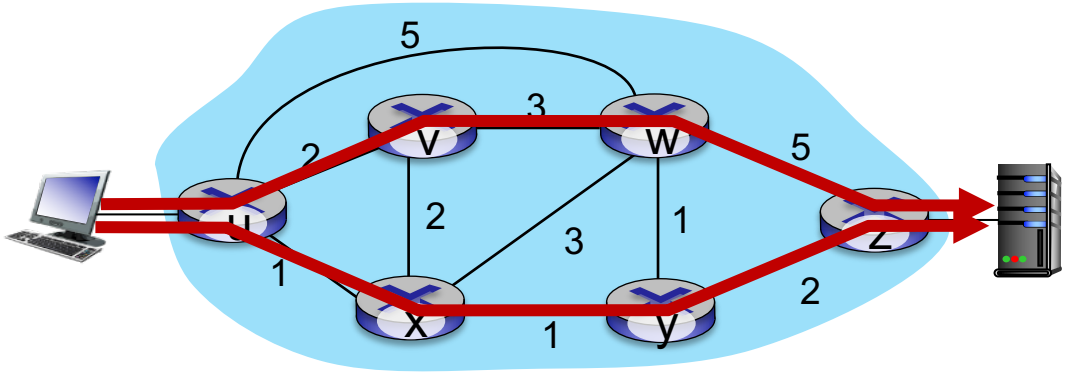
SDN analogy: mainframe to PC revolution

Vertically integrated
Closed, proprietary
Slow innovation
Small industry



Horizontal
Open interfaces
Rapid innovation
Huge industry

Traffic engineering: difficult with traditional routing

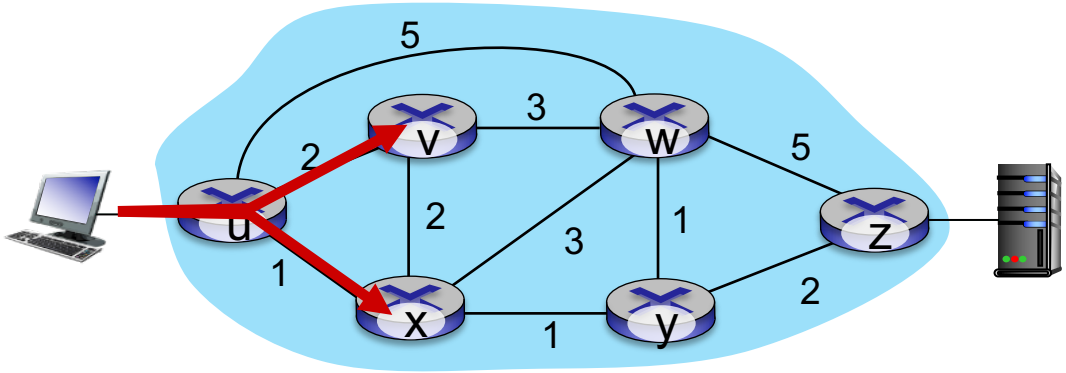


Q: what if network operator wants u-to-z traffic to flow along uvwz, rather than uxyz?

A: need to re-define link weights so traffic routing algorithm computes routes accordingly (or need a new routing algorithm)!

link weights are only control "knobs": not much control!

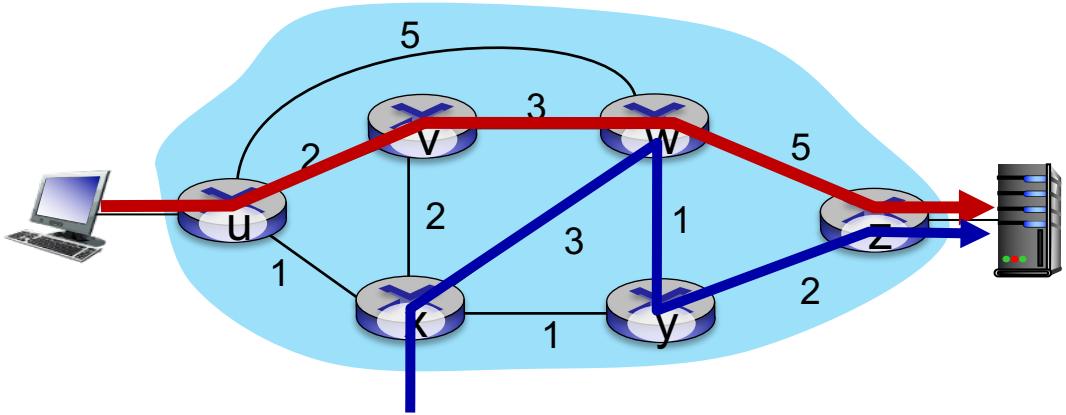
Traffic engineering: difficult with traditional routing



Q: what if network operator wants to split u-to-z traffic along uvwz and uxyz (load balancing)?

A: can't do it (or need a new routing algorithm)

Traffic engineering: difficult with traditional routing

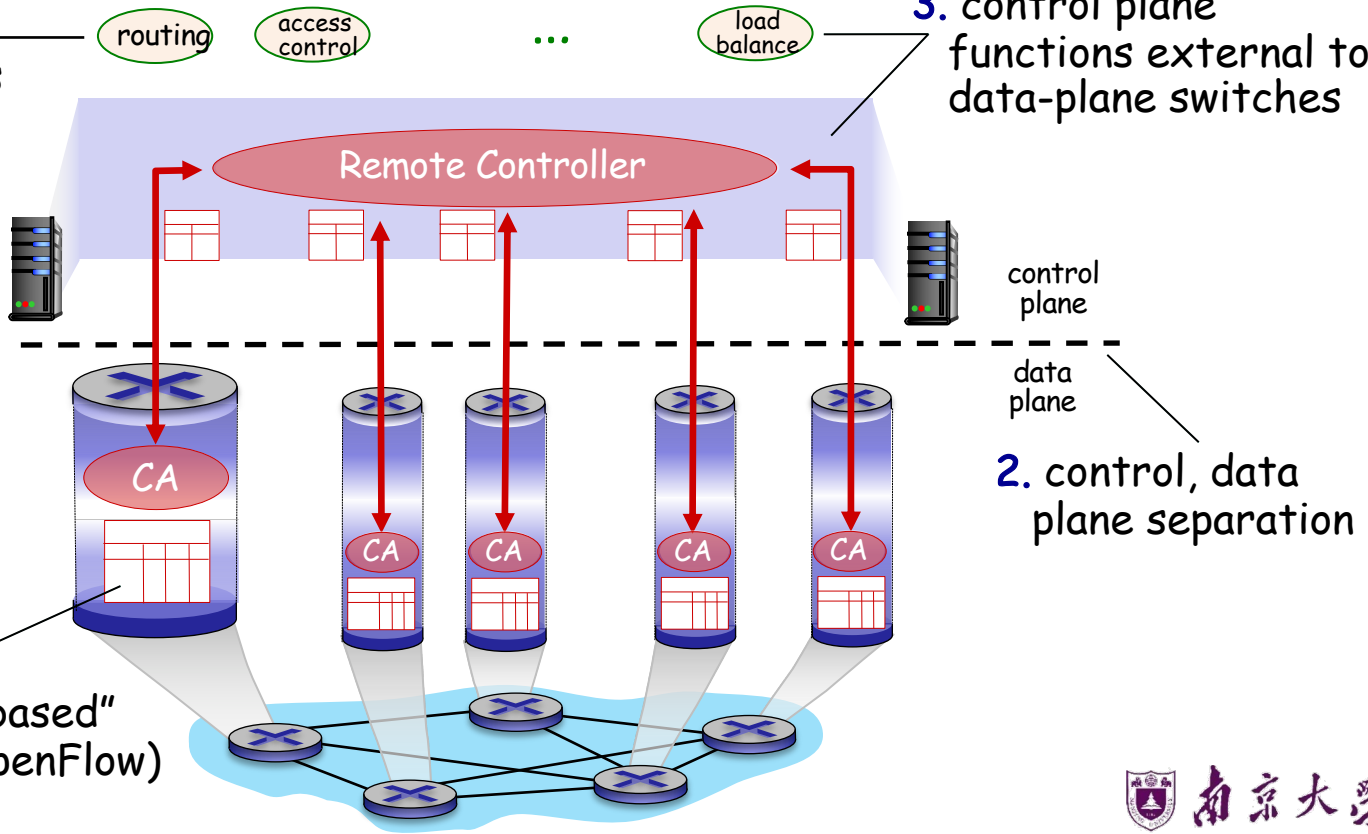


Q: what if w wants to route blue and red traffic differently from w to z?

A: can't do it (with destination-based forwarding, and LS, DV routing)

We learned in Chapter 4 that generalized forwarding and SDN can be used to achieve any routing desired

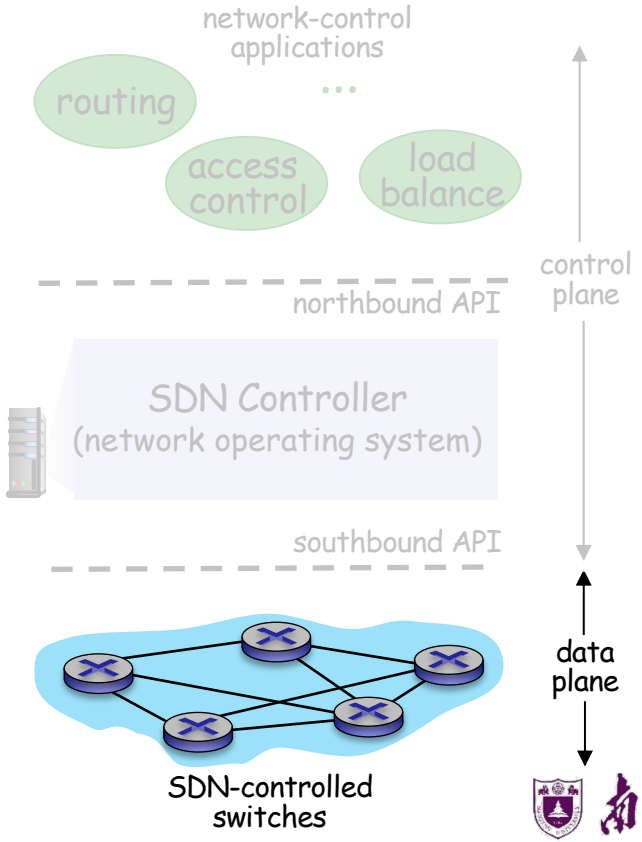
Software defined networking (SDN)



Software defined networking (SDN)

Data-plane switches:

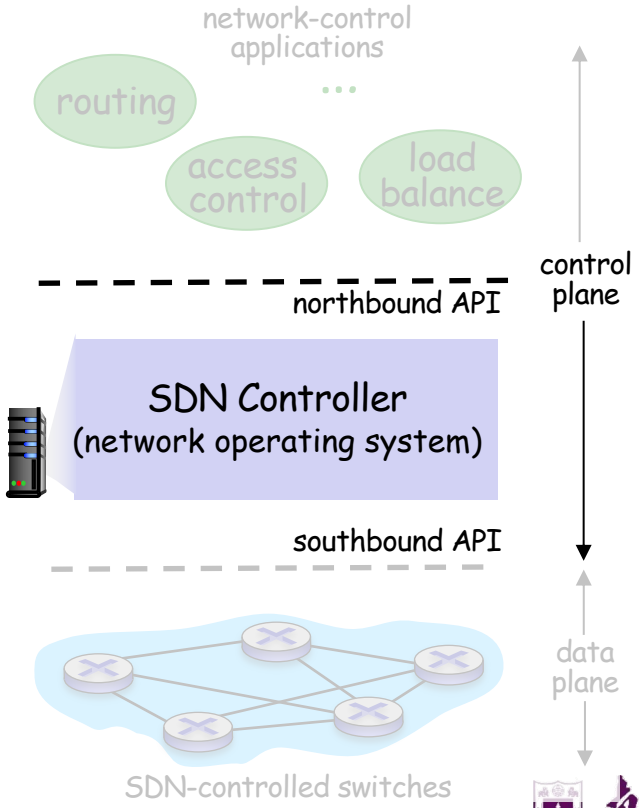
- fast, simple, commodity switches implementing generalized data-plane forwarding (Section 4.4) in hardware
- flow (forwarding) table computed, installed under controller supervision
- API for table-based switch control (e.g., OpenFlow)
  - defines what is controllable, what is not
- protocol for communicating with controller (e.g., OpenFlow)



Software defined networking (SDN)

SDN controller (network OS):

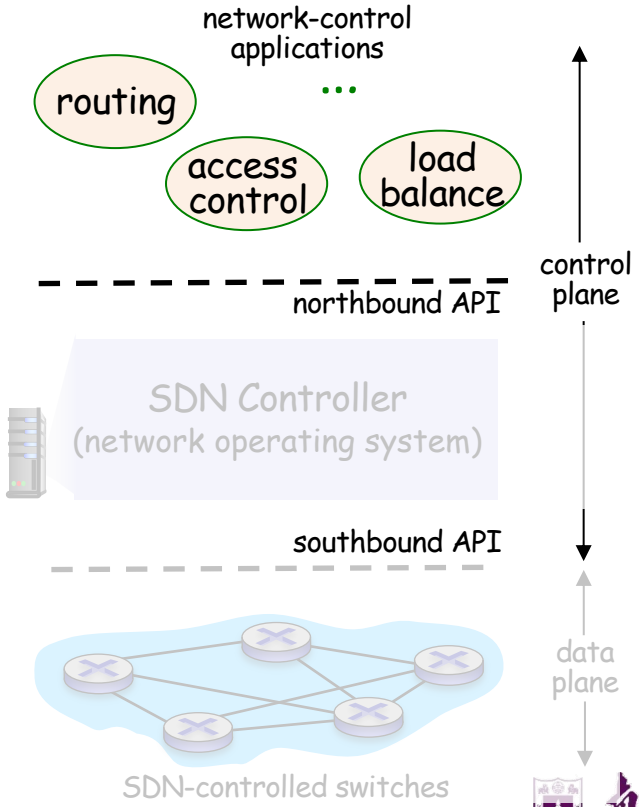
- maintain network state information
- interacts with network control applications "above" via northbound API
- interacts with network switches "below" via southbound API
- implemented as distributed system for performance, scalability, fault-tolerance, robustness



Software defined networking (SDN)

network-control apps:

- "brains" of control: implement control functions using lower-level services, API provided by SDN controller
- unbundled: can be provided by 3rd party: distinct from routing vendor, or SDN controller

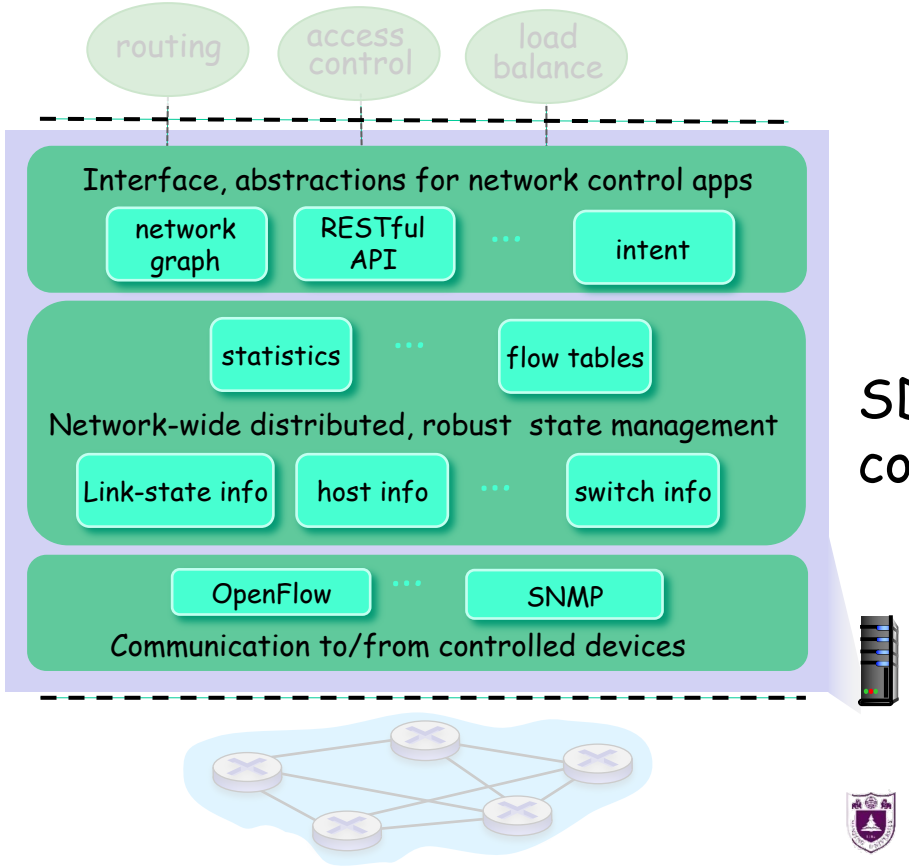


Components of SDN controller

interface layer to network control apps:
- abstractions API

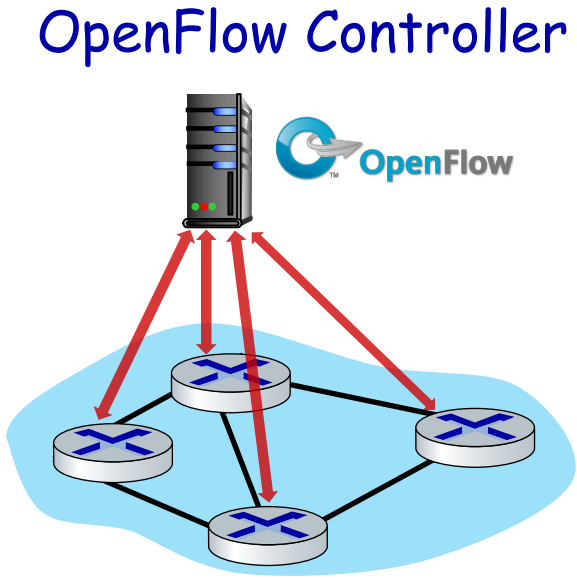
network-wide state management :
- state of networks links, switches, services: a distributed database

communication:
- 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.)
- distinct from OpenFlow API
  - API used to specify generalized forwarding actions

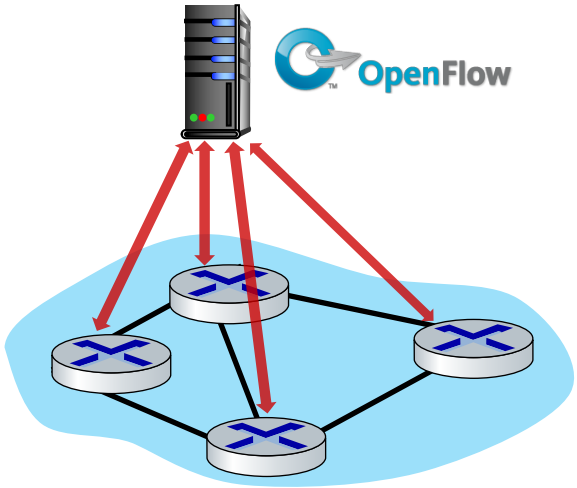


OpenFlow: controller-to-switch messages

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 Controller

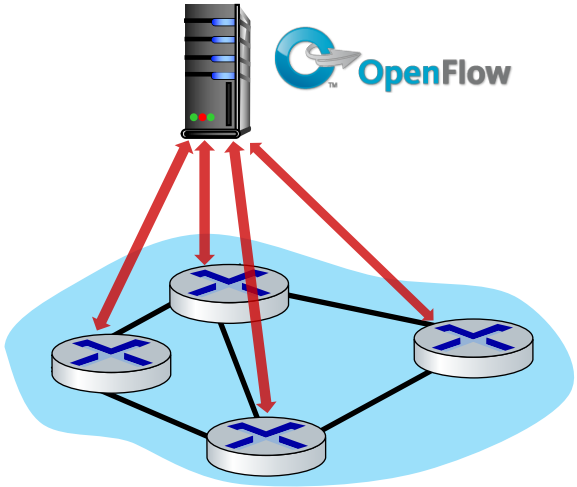


OpenFlow: switch-to-controller messages

Key switch-to-controller messages

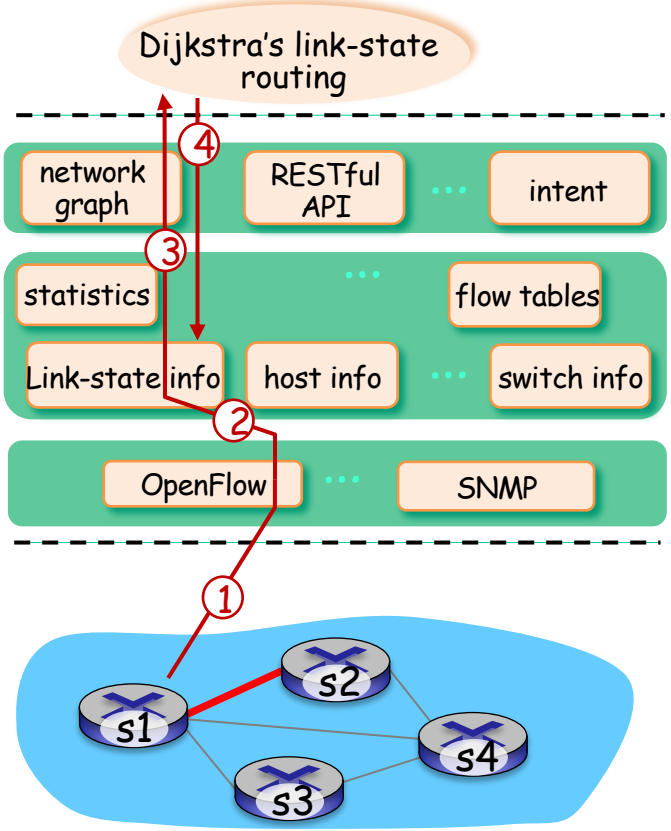
- packet-in: transfer packet (and its control) to controller. See packet-out message from controller
- flow-removed: flow table entry deleted at switch
- port status: inform controller of a change on a port.

OpenFlow Controller



Fortunately, network operators don't "program" switches by creating/sending OpenFlow messages directly. Instead use higher-level abstraction at controller

SDN: control/data plane interaction example



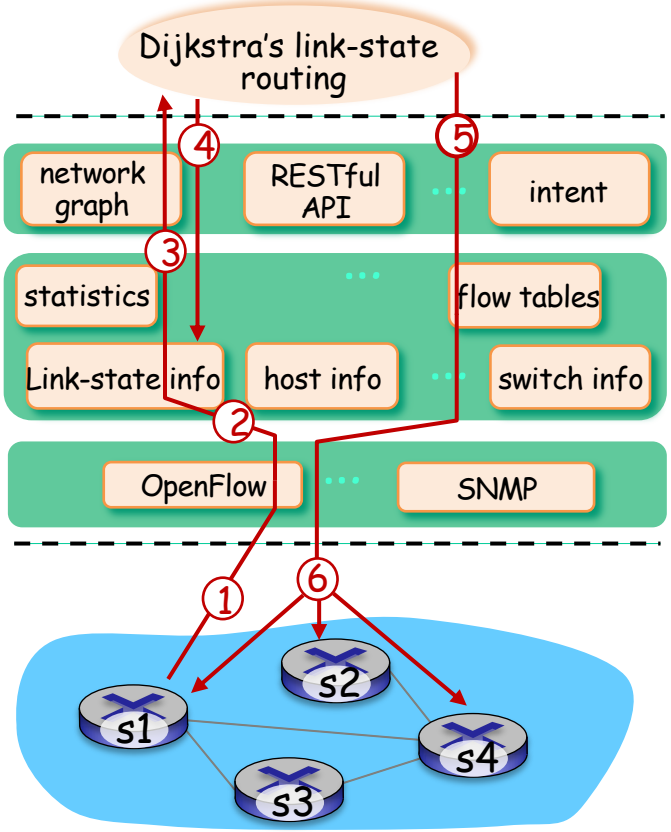
① S1, experiencing link failure uses OpenFlow port status message to notify controller

② SDN controller receives OpenFlow message, updates link status info

③ Dijkstra's routing algorithm application has previously registered to be called when ever link status changes. It is called.

④ Dijkstra's routing algorithm access network graph info, link state info in controller, computes new routes

SDN: control/data plane interaction example



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

Outline

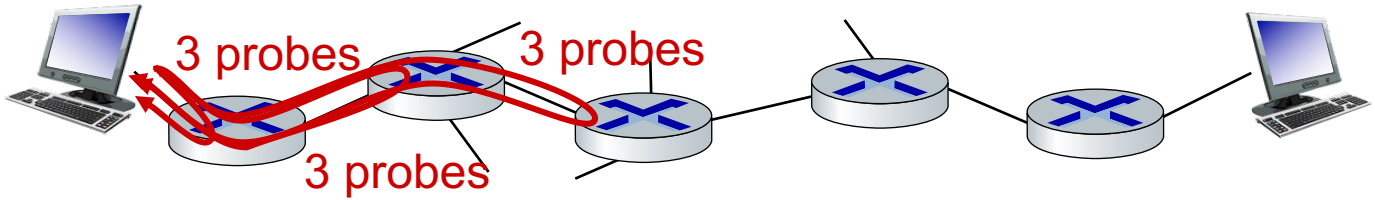
- Routing among ISPs: BGP
- SDN control plane
- Internet Control Message Protocol
- Network management, configuration

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
- ICMP message: type, code plus 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
  - 1st set has TTL =1, 2nd set has TTL=2, etc.
- datagram in *n*th set arrives to nth router:
  - router discards datagram and sends source ICMP message (type 11, code 0)
  - ICMP message possibly includes name of router & IP address
- 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

Outline

- Routing among ISPs: BGP
- SDN control plane
- Internet Control Message Protocol
- Network management, configuration

What is network management?

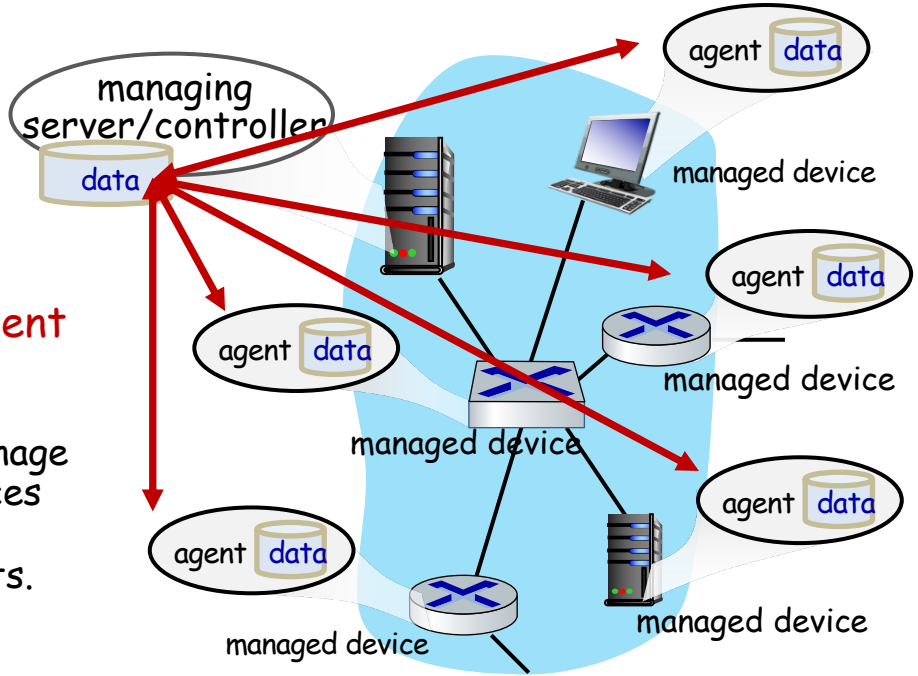
- autonomous systems (aka "network"): 1000s of interacting hardware/software components
- other complex systems requiring monitoring, configuration, control:
  - jet airplane, nuclear power plant, others?

"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."

Components of network management

Managing server: application, typically with network managers (humans) in the loop

Network management protocol: used by managing server to query, configure, manage device; used by devices to inform managing server of data, events.



Managed device: equipment with manageable, configurable hardware, software components

Data: device "state" configuration data, operational data, device statistics

Network operator approaches to management

CLI (Command Line Interface)

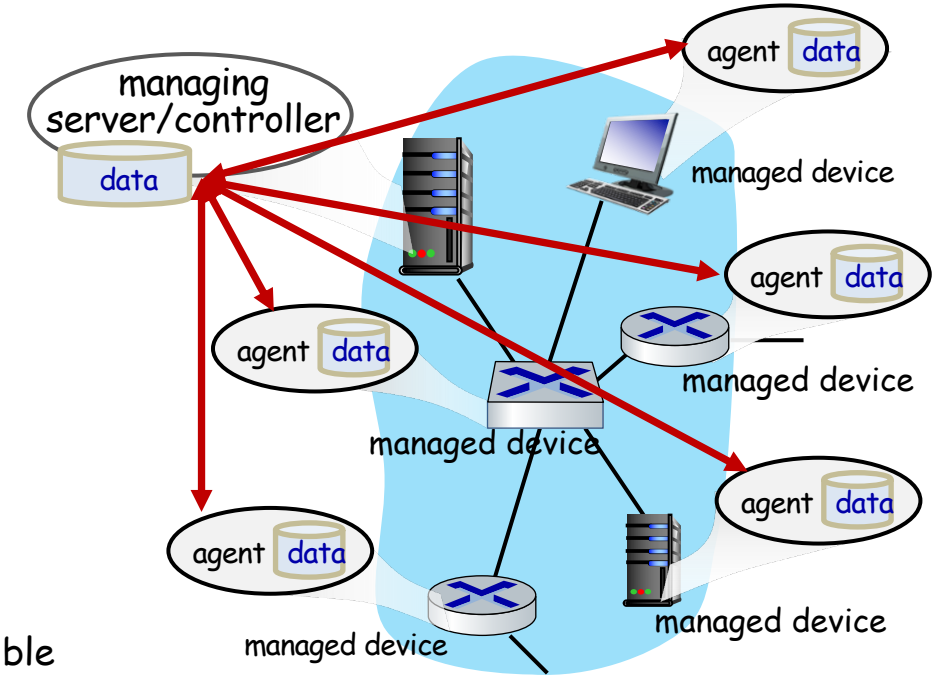
- operator issues (types, scripts) direct to individual devices (e.g., vis ssh)

SNMP/MIB

- operator queries/sets devices data (MIB) using Simple Network Management Protocol (SNMP)

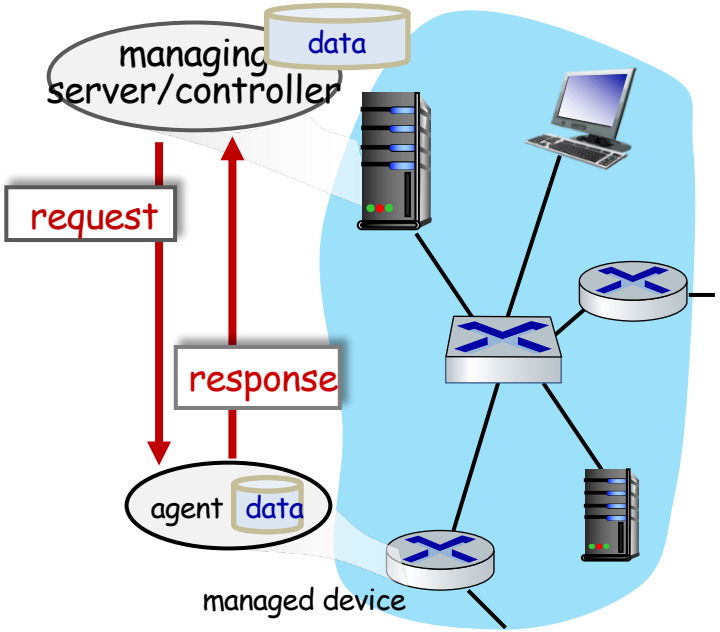
NETCONF/YANG

- more abstract, network-wide, holistic
- emphasis on multi-device configuration management.
- YANG: data modeling language
- NETCONF: communicate YANG-compatible actions/data to/from/among remote devices

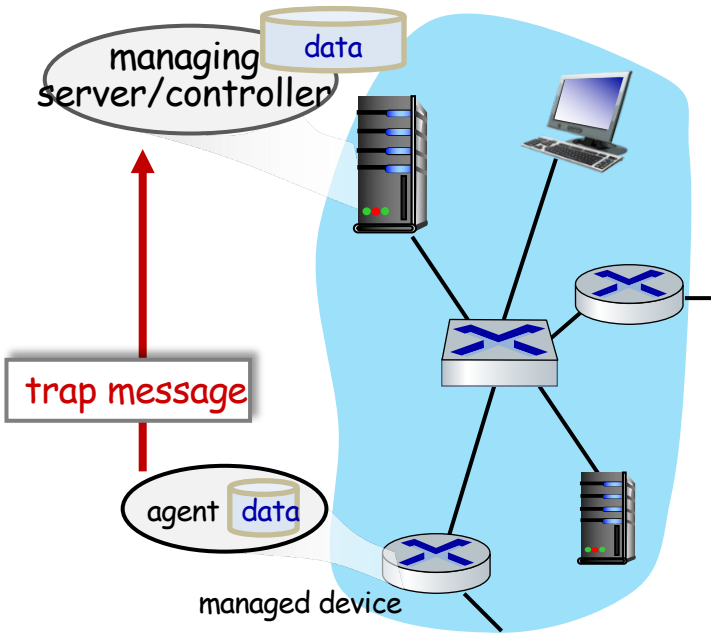


SNMP protocol

Two ways to convey MIB info, commands:



request/response mode

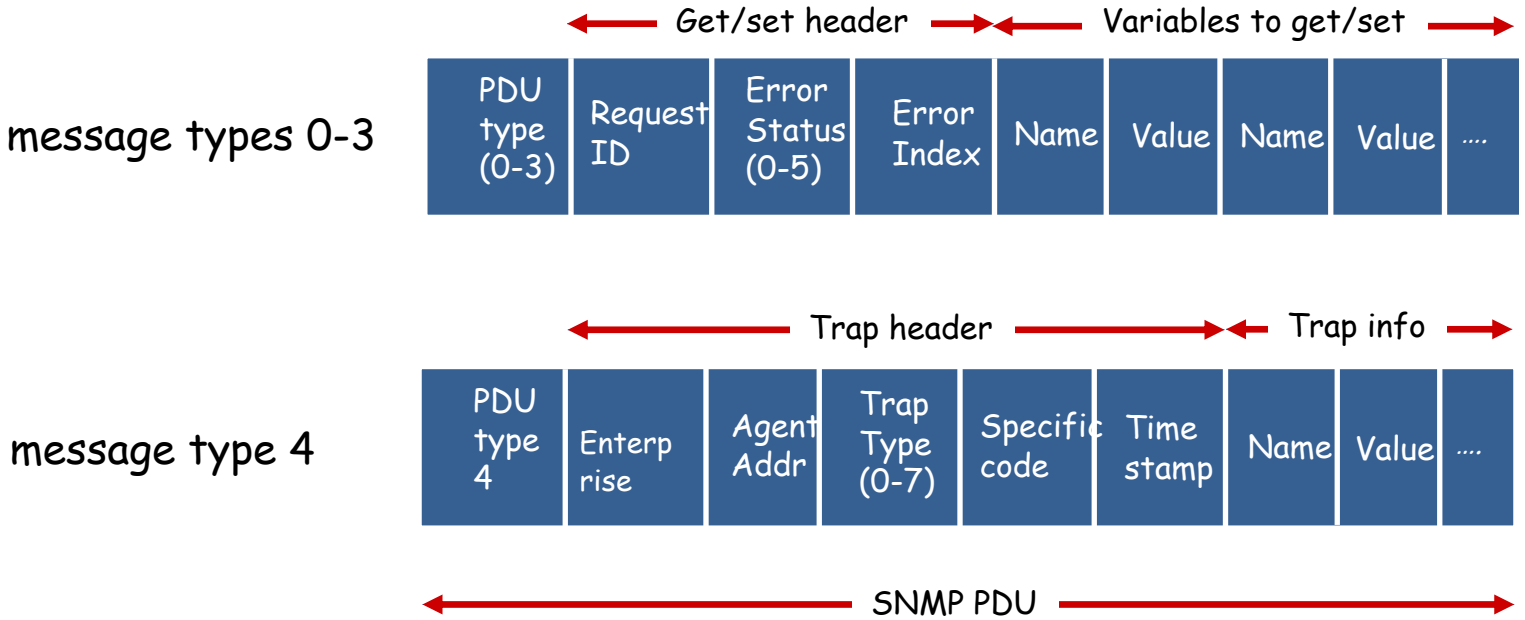


trap mode

SNMP protocol: message types

Message type	Function
GetRequest GetNextRequest GetBulkRequest	manager-to-agent: "get me data" (data instance, next data in list, block of data).
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



SNMP: Management Information Base (MIB)

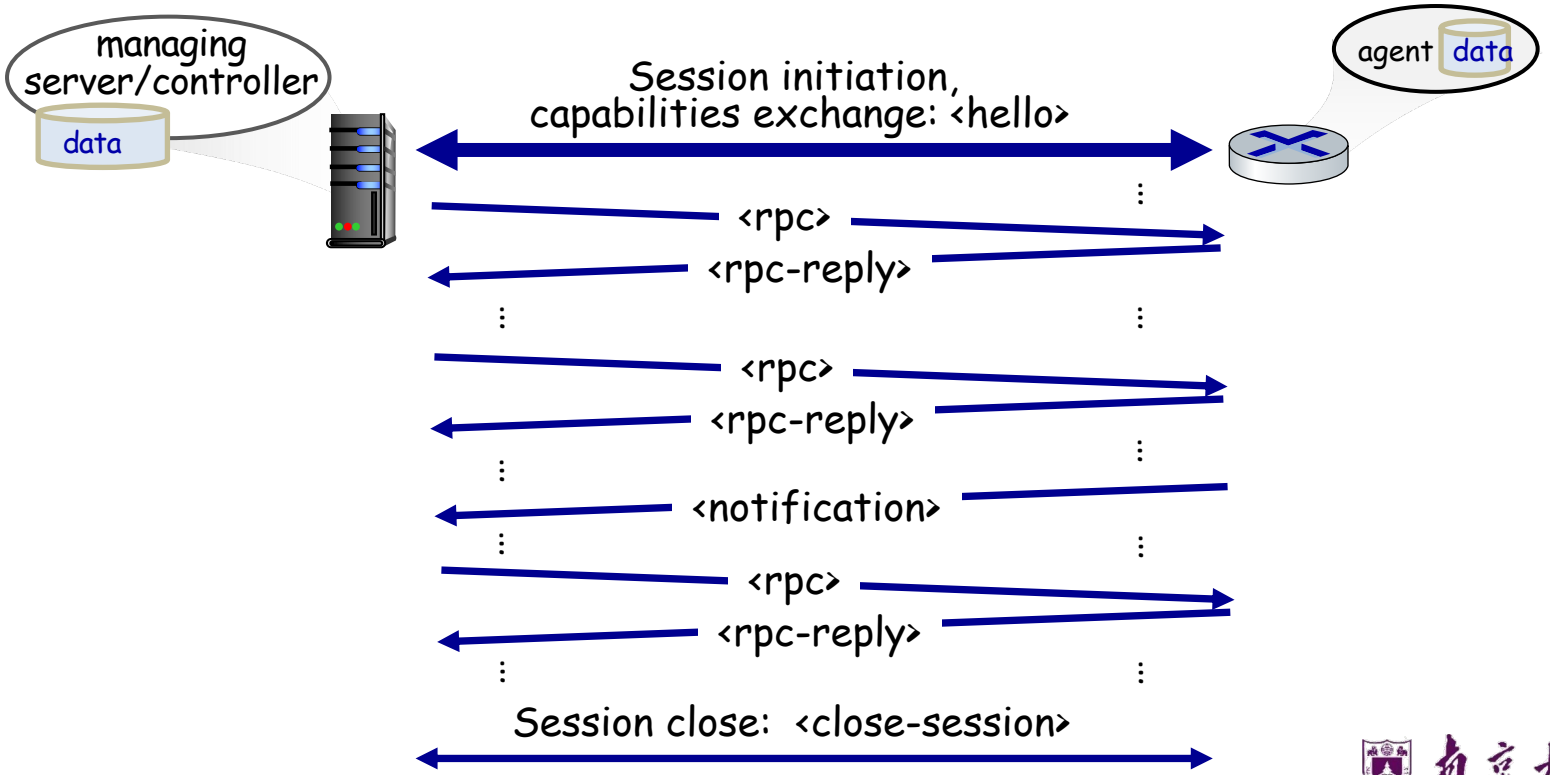
- managed device's operational (and some configuration) data
- gathered into device MIB module
  - 400 MIB modules defined in RFC's; many more vendor-specific MIBs
- Structure of Management Information (SMI): data definition language
- example MIB variables for UDP protocol:

Object ID	Name	Type	Comments
1.3.6.1.2.1.7.1	UDPInDatagrams	32-bit counter	total # datagrams delivered
1.3.6.1.2.1.7.2	UDPNoPorts	32-bit counter	# undeliverable datagrams (no application at port)
1.3.6.1.2.1.7.3	UDInErrors	32-bit counter	# undeliverable datagrams (all other reasons)
1.3.6.1.2.1.7.4	UDPOutDatagrams	32-bit counter	total # datagrams sent
1.3.6.1.2.1.7.5	udpTable	SEQUENCE	one entry for each port currently in use

NETCONF overview

- goal: actively manage/configure devices network-wide
- operates between managing server and managed network devices
  - actions: retrieve, set, modify, activate configurations
  - atomic-commit actions over multiple devices
  - query operational data and statistics
  - subscribe to notifications from devices
- remote procedure call (RPC) paradigm
  - NETCONF protocol messages encoded in XML
  - exchanged over secure, reliable transport (e.g., TLS) protocol

NETCONF initialization, exchange, close



Selected NETCONF Operations

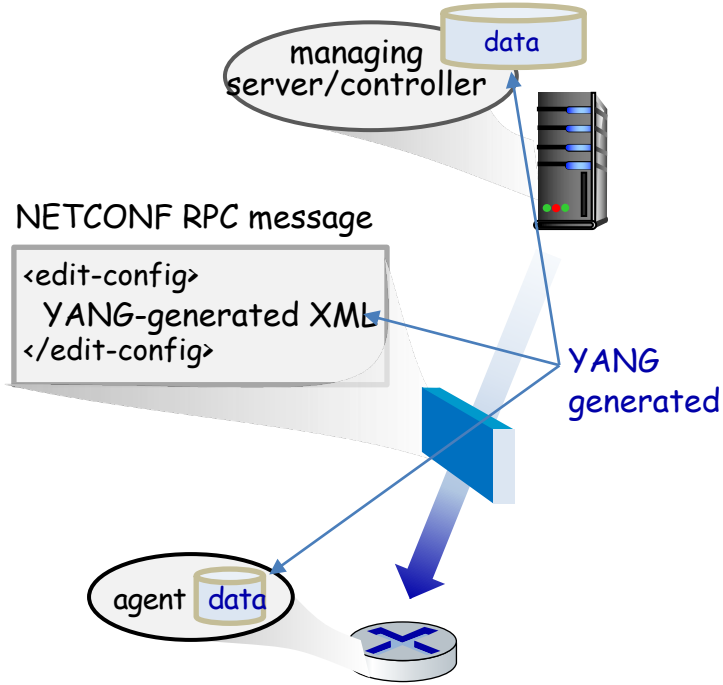
NETCONF	Operation Description
<get-config>	Retrieve all or part of a given configuration. A device may have multiple configurations.
<get>	Retrieve all or part of both configuration state and operational state data.
<edit-config>	Change specified (possibly running) configuration at managed device. Managed device <rpc-reply> contains <ok> or <rpcerror> with rollback.
<lock>, <unlock>	Lock (unlock) configuration datastore at managed device (to lock out NETCONF, SNMP, or CLIs commands from other sources).
<create-subscription>, <notification>	Enable event notification subscription from managed device

Sample NETCONF RPC message

```
01 <?xml version="1.0" encoding="UTF-8"?>
02 <rpc message-id="101"                    note message id
03   xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
04   <edit-config>                         change a configuration
05     <target>
06       <running/>                        change the running configuration
07     </target>
08     <config>
09       <top xmlns="http://example.com/schema/
         1.2/config">
10         <interface>
11           <name>Ethernet0/0</name>      change MTU of Ethernet 0/0 interface to 1500
12           <mtu>1500</mtu>
13         </interface>
14       </top>
15     </config>
16   </edit-config>
17 </rpc>
```

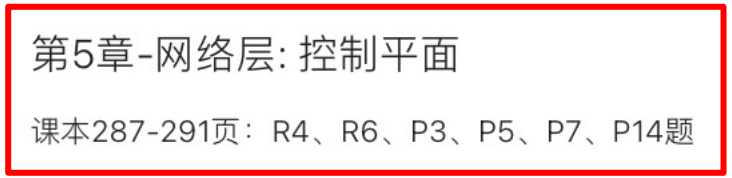
YANG

- data modeling language used to specify structure, syntax, semantics of NETCONF network management data
  - built-in data types, like SMI
- XML document describing device, capabilities can be generated from YANG description
- can express constraints among data that must be satisfied by a valid NETCONF configuration
  - ensure NETCONF configurations satisfy correctness, consistency constraints



课程习题（作业）——截止日期：4月29日晚23:59

- **课本287-291页**：R4、R6、P3、P5、P7、P14题
- 提交方式：https://selearning.nju.edu.cn/（教学支持系统）



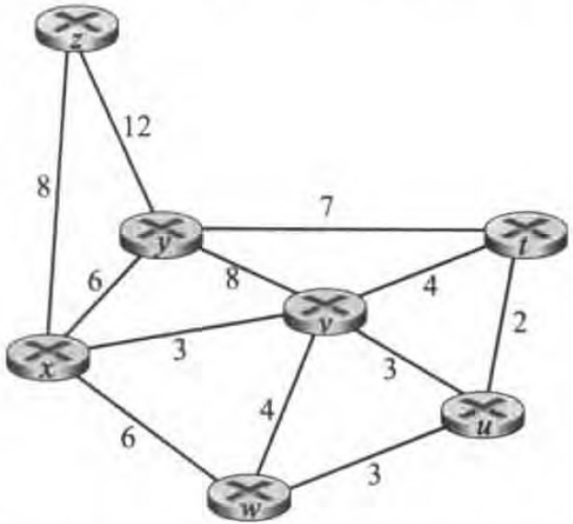
- 命名：学号+姓名+第*章。
- 若提交遇到问题请及时发邮件或在下一次上课时反馈。

课程习题（作业）——截止日期：4月29日晚23:59

R4. 比较和对照链路状态和距离矢量这两种路由选择算法。

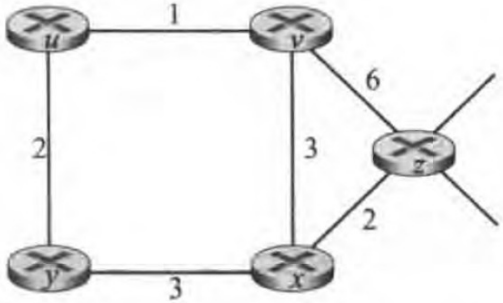
R6. 每个自治系统使用相同的 AS 内部路由选择算法是必要的吗？说明其原因。

P3. 考虑下面的网络。对于标明的链路开销，用 Dijkstra 的最短路算法计算出从 x 到所有网络节点的最短路径。通过计算一个类似于表 5-1 的表，说明该算法是如何工作的。



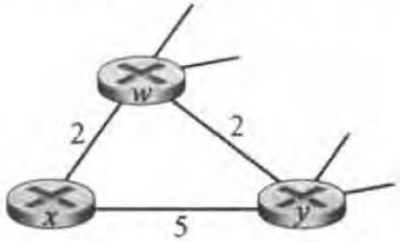
课程习题（作业）——截止日期：4月29日晚23:59

P5. 考虑下图所示的网络，假设每个节点初始时知道到它的每个邻居的开销。考虑距离向量算法，并显示在节点 z 中的距离表表项。



课程习题（作业）——截止日期：4月29日晚23:59

P7. 考虑下图所示的网络段。x 只有两个相连邻居 w 与 y。w 有一条通向目的地 u（没有显示）的最低开销路径，其值为5，y 有一条通向目的地 u 的最低开销路径，其值为6。从 w 与 y 到 u（以及 w 与 y 之间）的完整路径未显示出来。网络中所有链路开销皆为正整数值。



a. 给出 x 对目的地 w、y 和 u 的距离向量。

b. 给出对于 $c(x, w)$ 或 $c(x, y)$ 的链路开销的变化，使得执行了距离向量算法后，x 将通知其邻居有一条通向 u 的新最低开销路径。

c. 给出对 $c(x, w)$ 或 $c(x, y)$ 的链路开销的变化，使得执行了距离向量算法后，x 将不通知其邻居有一条通向 x 的新最低开销路径。

课程习题（作业）——截止日期：4月29日晚23:59

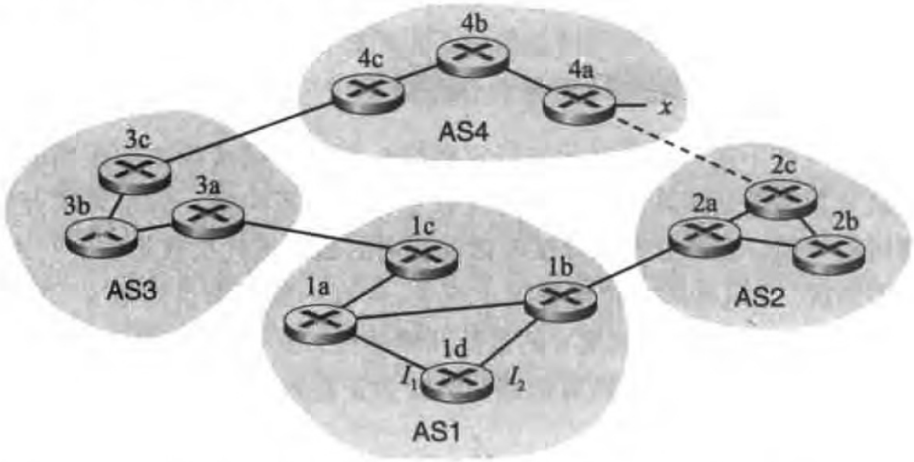
P14. 考虑下图所示的网络。假定 AS3 和 AS2 正在运行 OSPF 作为其 AS 内部路由选择协议。假定 AS1 和 AS4 正在运行 RIP 作为其 AS 内部路由选择协议。假定 AS 间路由选择协议使用的是 eBGP 和 iBGP。假定最初在 AS2 和 AS4 之间不存在物理链路。

a. 路由器 3c 从哪个路由选择协议学习到了前缀 x?

b. 路由器 3a 从哪个路由选择协议学习到了前缀 x?

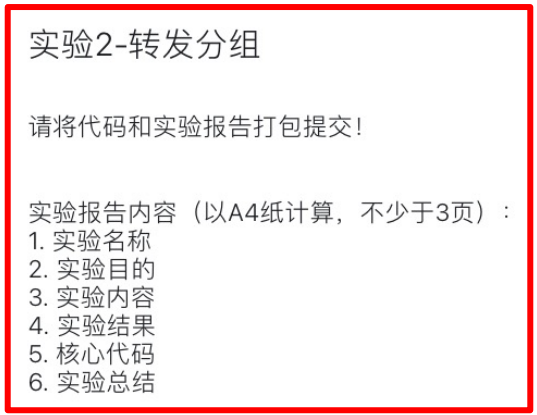
c. 路由器 1c 从哪个路由选择协议学习到了前缀 x?

d. 路由器 1d 从哪个路由选择协议学习到了前缀 x?



实验2——截止日期：5月6日晚23:59

- 实验2：转发分组
- 提交方式：https://selearning.nju.edu.cn/（教学支持系统）



- 命名：学号+姓名+实验*。
- 若提交遇到问题请及时发邮件或在下一次上课时反馈。

实验2——截止日期：5月6日晚23:59

Lab 2: Forwarding Packets

Overview

This is the second exercise for creating the "brains" of an IPv4 router. The basic functions of an Internet router are to:

1. Respond to ARP (address resolution protocol) requests for addresses that are assigned to interfaces on the router. (Remember that the purpose of ARP is to obtain the Ethernet MAC address associated with an IP address so that an Ethernet frame can be sent to another host over the link layer.)
2. Receive and forward packets that arrive on links and are destined for other hosts. Part of the forwarding process is to perform address lookups ("longest prefix match" lookups) in the forwarding table. We will just use "static" routing in our router rather than a dynamic routing protocol like RIP or OSPF.
3. Make ARP requests for IP addresses that have no known Ethernet MAC address. A router will often have to send packets to other hosts, and needs Ethernet MAC addresses to do so.
4. Respond to ICMP messages like echo requests ("pings").
5. Generate ICMP error messages when necessary, such as when an IP packet's TTL (time to live) value has been decremented to zero.

The goal of this stage is to accomplish #2 and #3 above.

实验2——截止日期：5月6日晚23:59

Lab 2: Forwarding Packets

Task 1: Preparation
Initiate your project with our template.
Start the task here

Task 2: IP Forwarding Table Lookup
Build a forwarding table and match the destination addresses.
Start the task here

Task 3: Forwarding the Packet and ARP
Send an ARP query to create a new Ethernet header and forward the packet.
Start the task here

提问

Q & A

殷亚凤
智能软件与工程学院
苏州校区南雍楼东区225
yafeng@nju.edu.cn，https://yafengnju.github.io/

南京大学 NANJING UNIVERSITY