CS4450

Computer Networks:
Architecture and Protocols

Lecture 19
BGP limitations
Switch Architecture

Rachit Agarwal

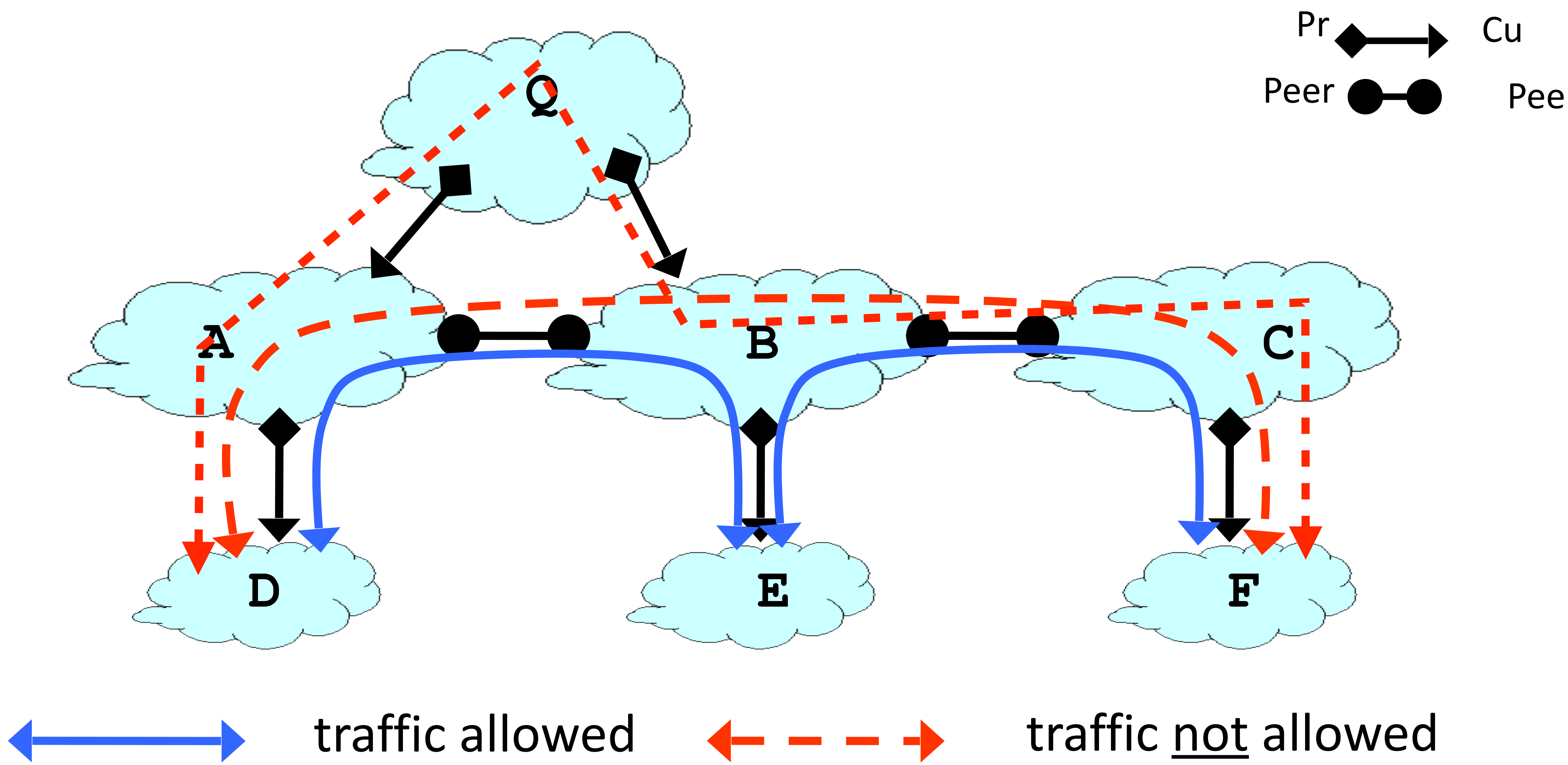
Announcements

- **Exam 2 grades released**
  - Please submit regrade requests only if your answer matches the rubric
- We will release our **first programming assignment** this week
  - Recall: not graded, but we will provide all the help

Goals for Today's Lecture

- Wrap up BGP
- Understand switch/router architecture

Recap: Inter-domain Routing Follows the Money



- ASes provide "transit" between their customers
- Peers do not provide transit between other peers

BGP is Inspired by Distance Vector

- Per-destination route advertisements
- No global sharing of network topology
- Iterative and distributed convergence on paths
- But, four key differences
  - BGP does not pick shortest paths
  - Each node announces one or multiple PATHs per destination
  - Selective Route advertisement: not all paths are announced
  - BGP may aggregate paths
    - may announce one path for multiple destinations

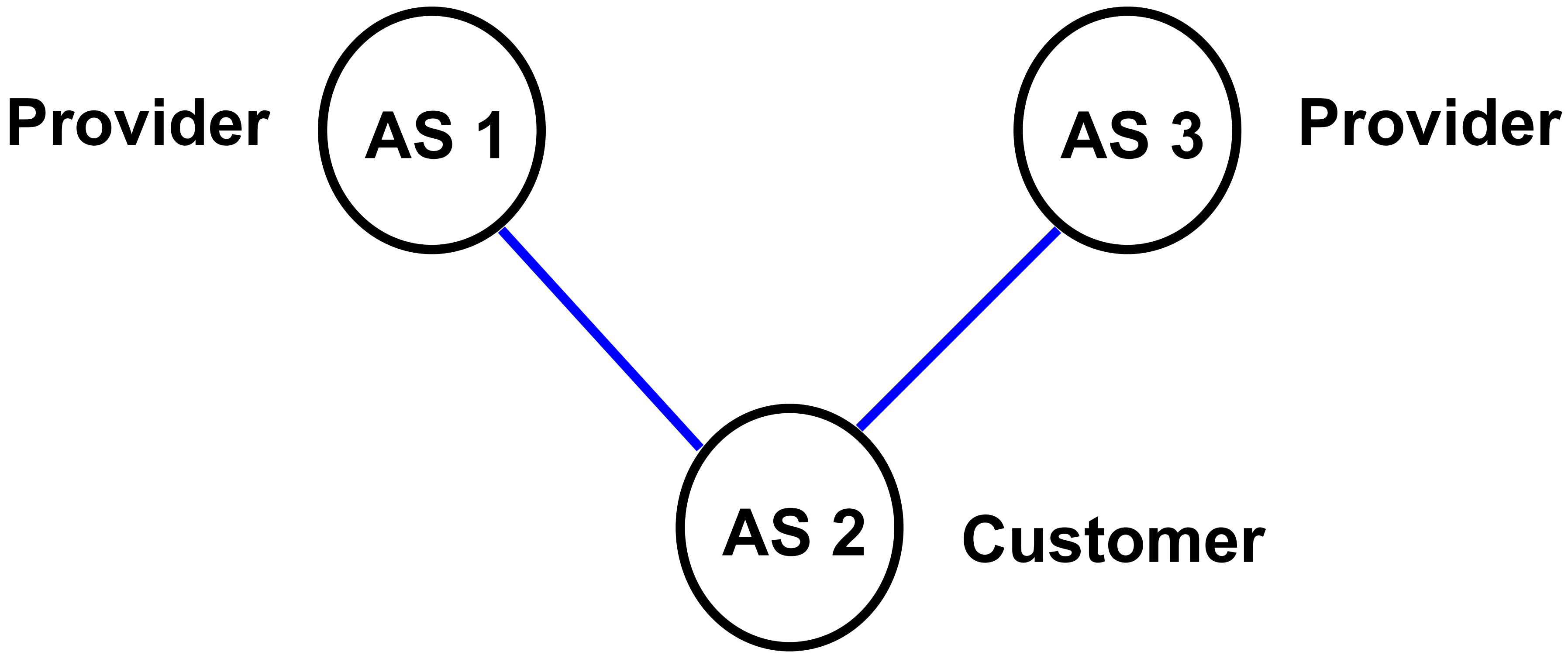
BGP Issues

BGP: Issues

- Reachability
- Security
- Convergence
- Performance
- Anomalies

Reachability

- In normal routing, if graph is connected then reachability is assured
- With policy routing, this doesn't always hold



Security

- An AS can claim to serve a prefix that they actually don't have a route to (blackholing traffic)
  - Problem not specific to policy or path vector
  - Important because of AS autonomy
  - *Fixable: make ASes prove they have a path*

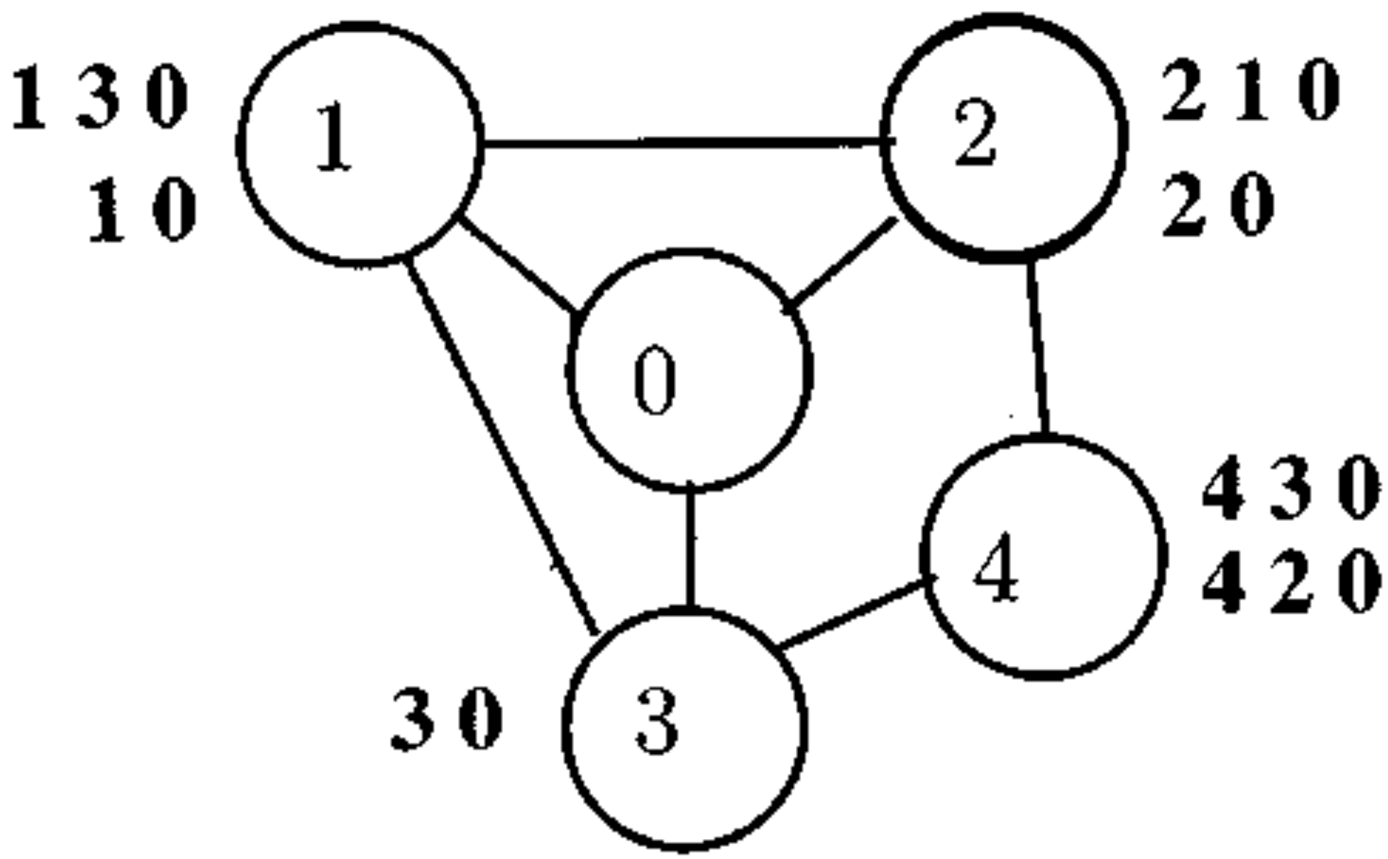
- But…
- AS may forward packets along a route different from what is advertised
  - Tell customers about a fictitious short path…
  - Much harder to fix!

Convergence

- If all AS policies follow Gao-Rexford rules,
  - Then BGP is guaranteed to converge (safety)

- For arbitrary policies, BGP may fail to converge!

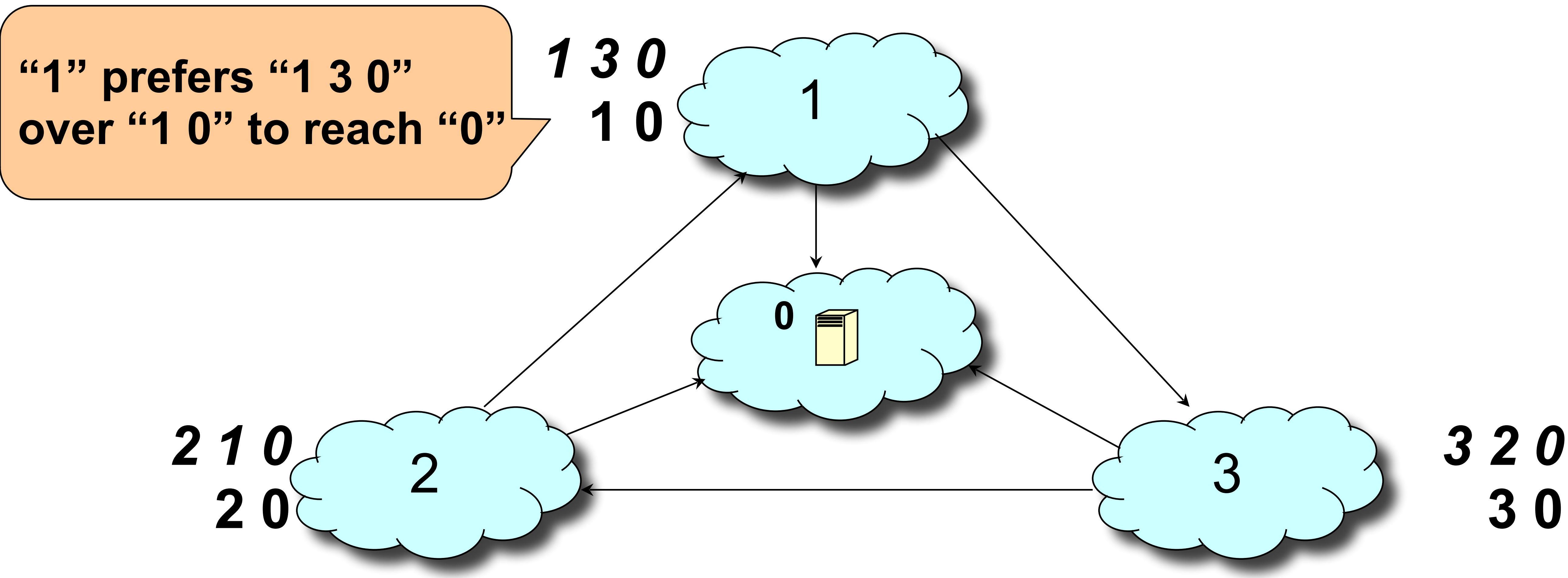
BGP Example (All good)



GOOD GADGET

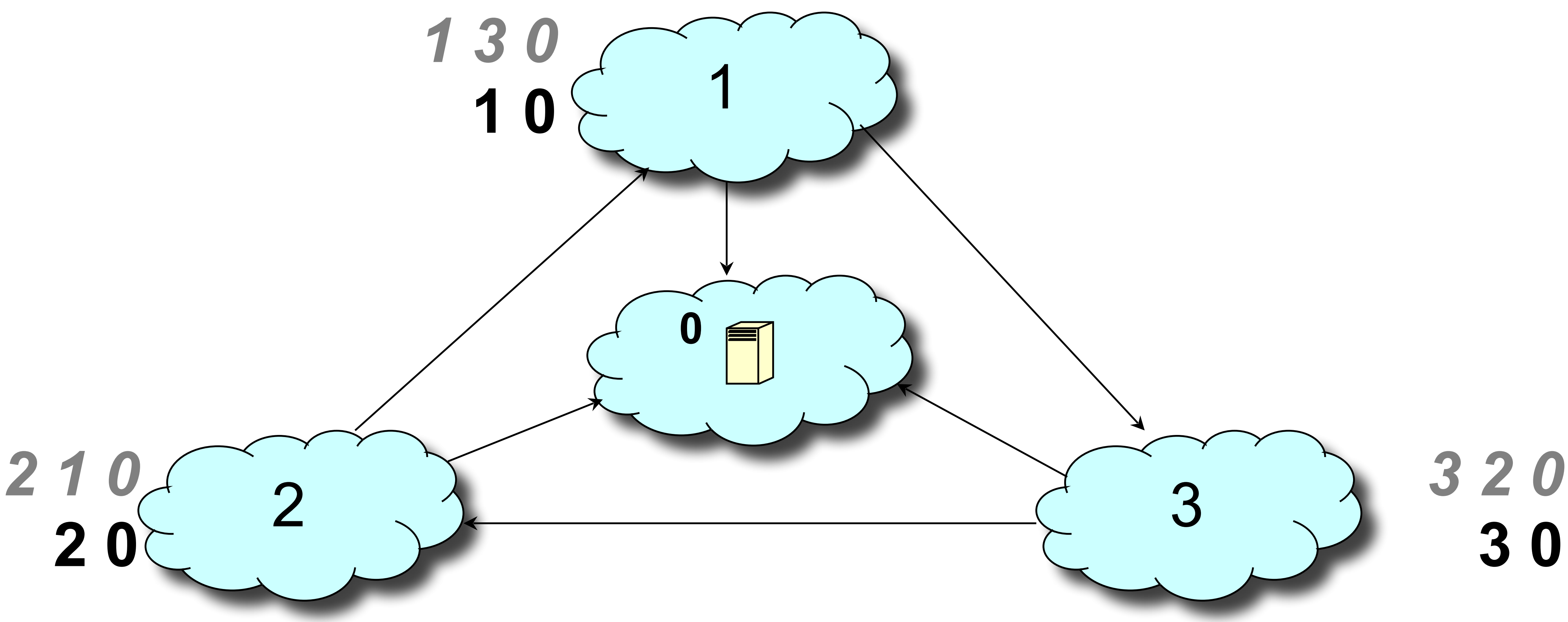
	1	2	3	4
R1	10	20	**30**	-
R2	10	20	**30**	430
R3	130	20	30	430

Example of Policy Oscillation



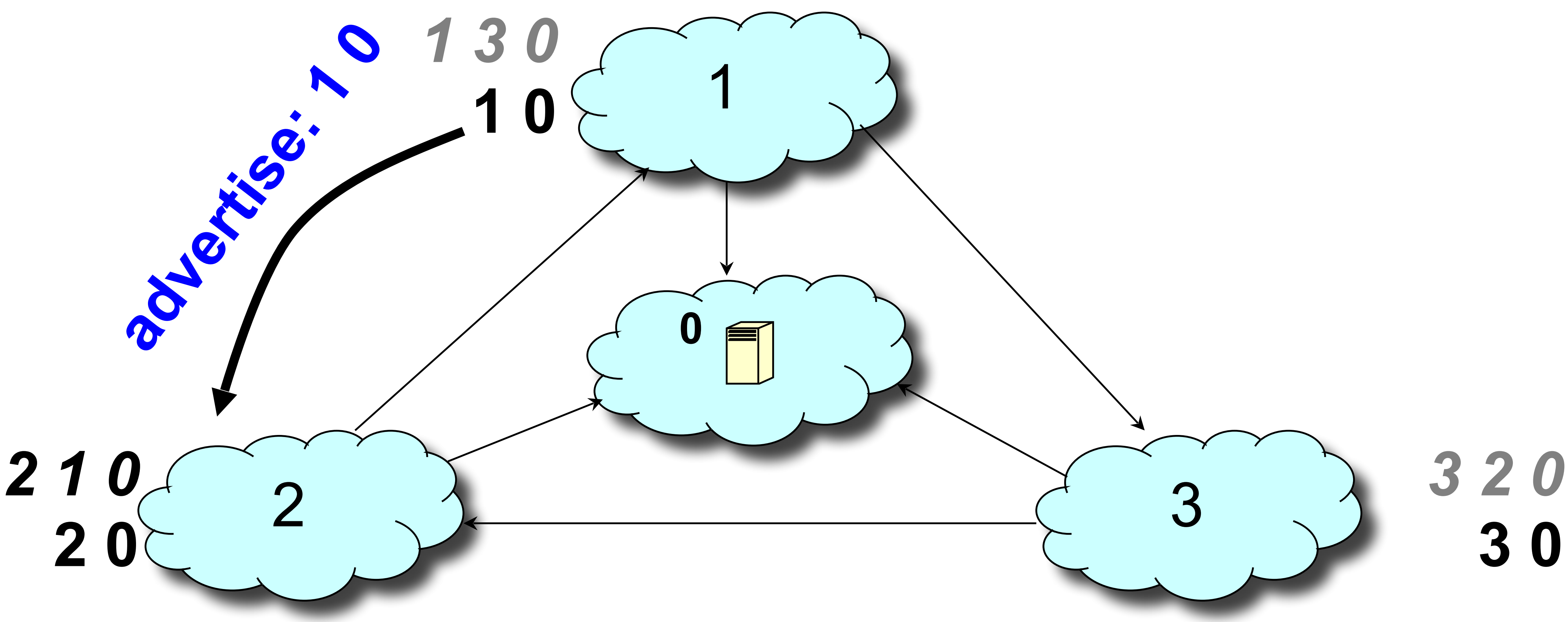
Step-by-step Policy Oscillation

Initially: nodes 1, 2, 3 know only shortest path to 0

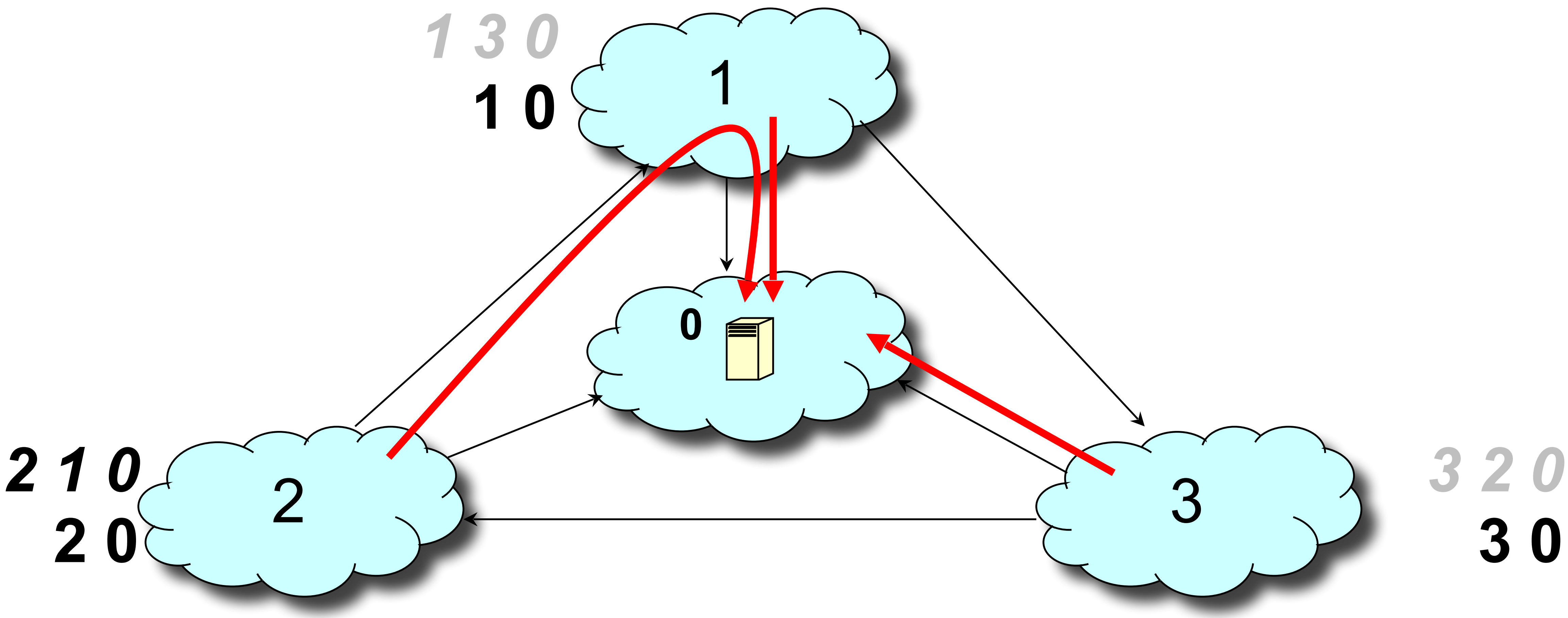


Step-by-step Policy Oscillation

1 **advertises** its path 1 0 to 2

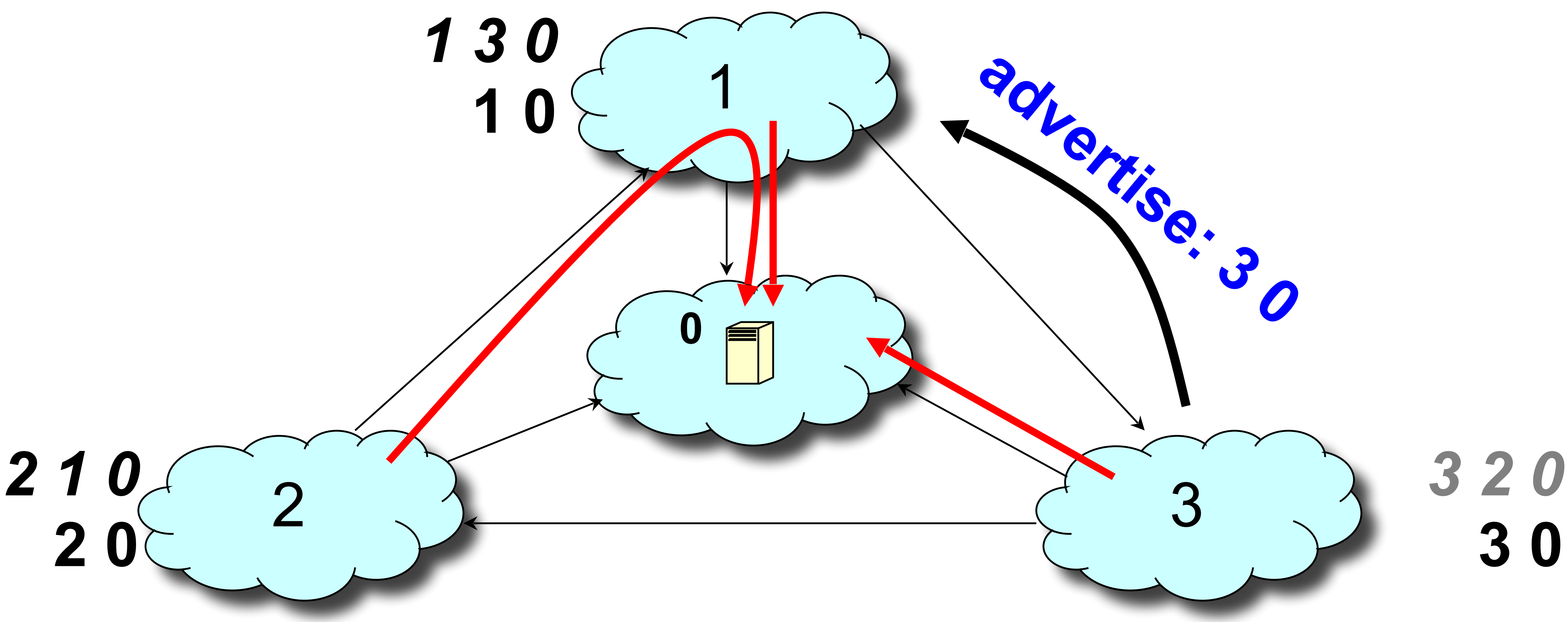


Step-by-step Policy Oscillation

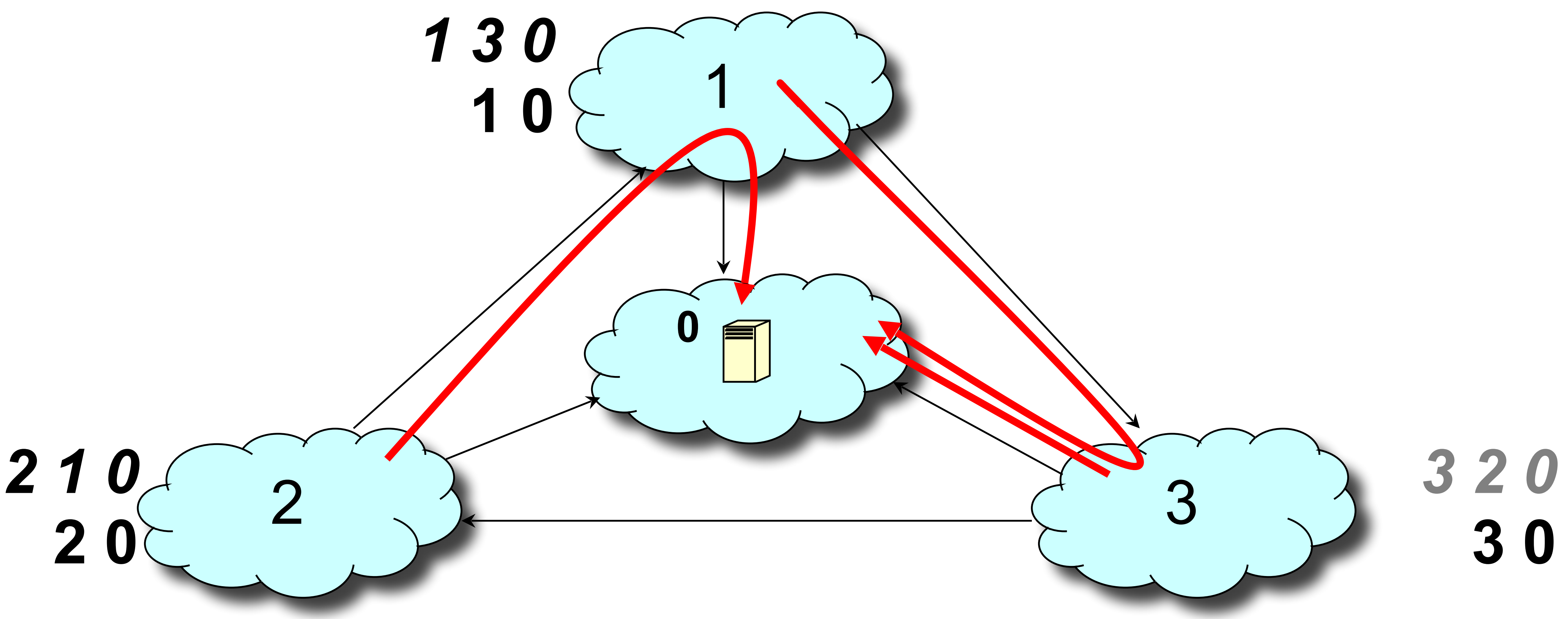


Step-by-step Policy Oscillation

3 **advertises** its path 3 0 to 1

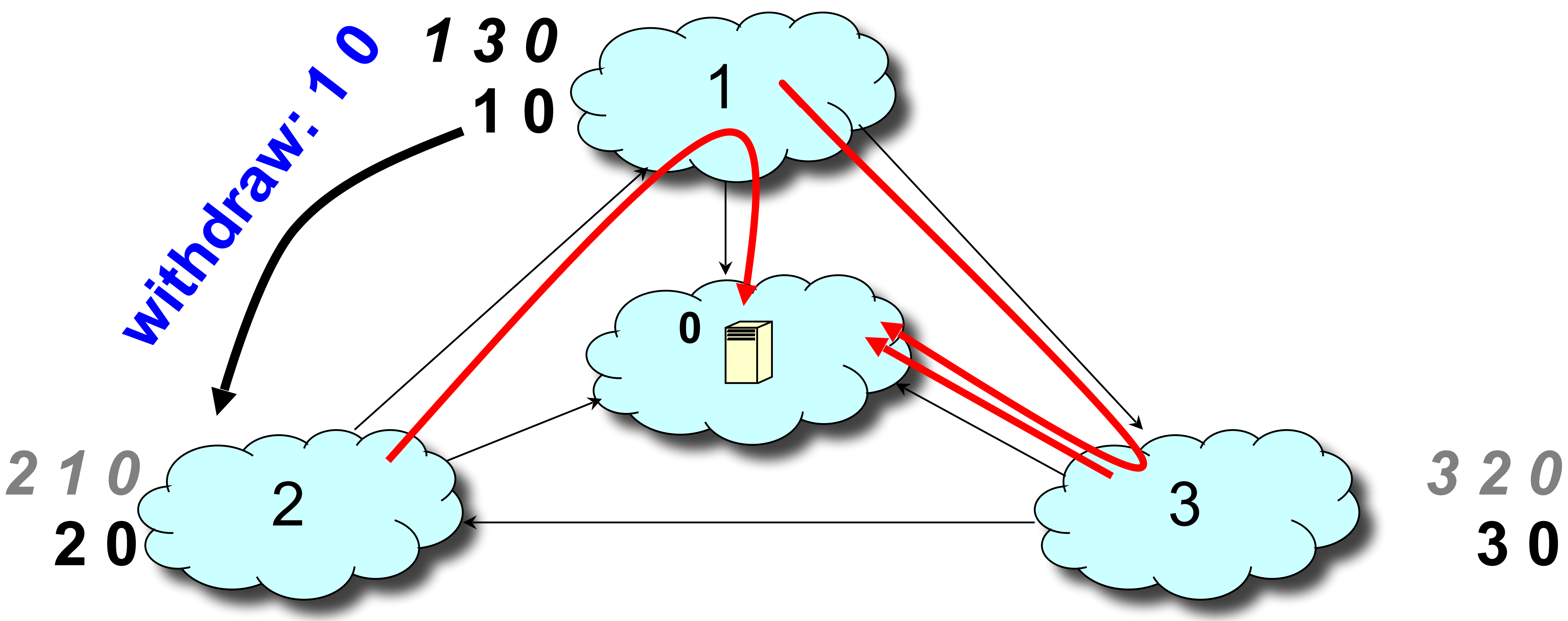


Step-by-step Policy Oscillation

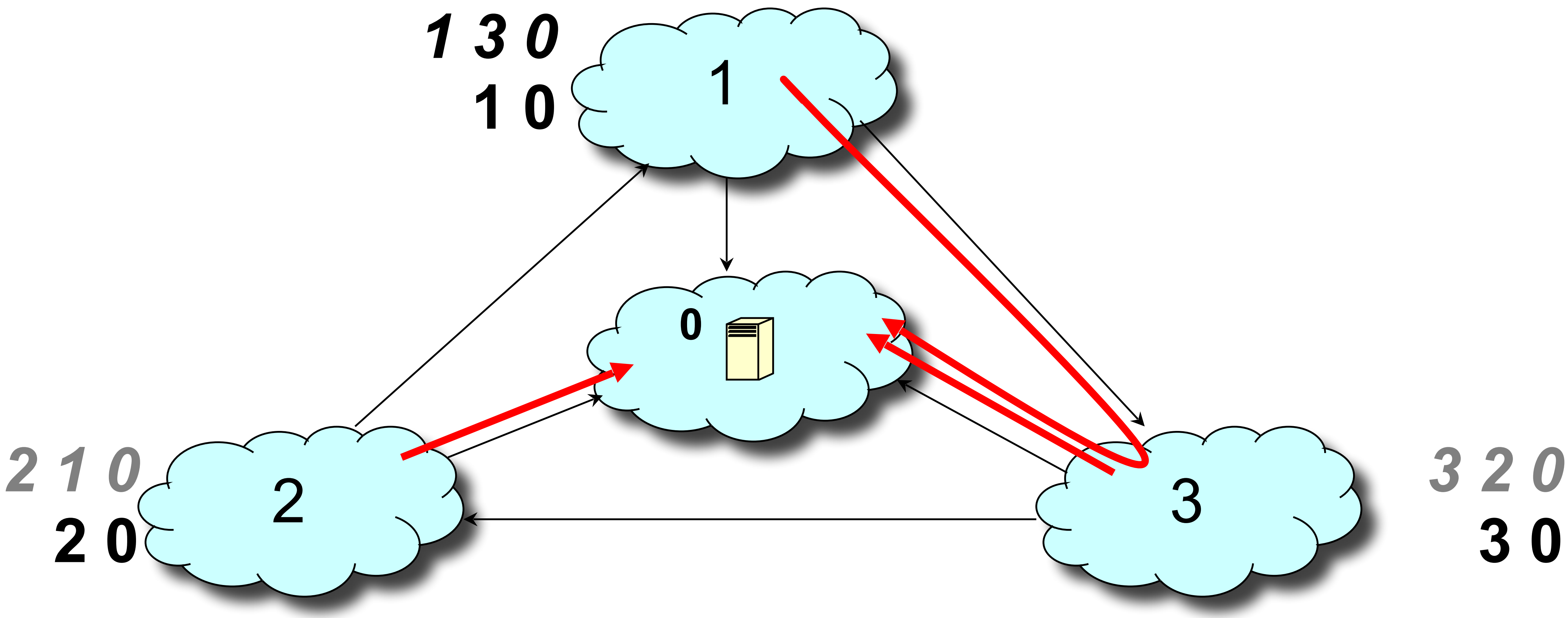


Step-by-step Policy Oscillation

1 **withdraws** its path 1 0 from 2

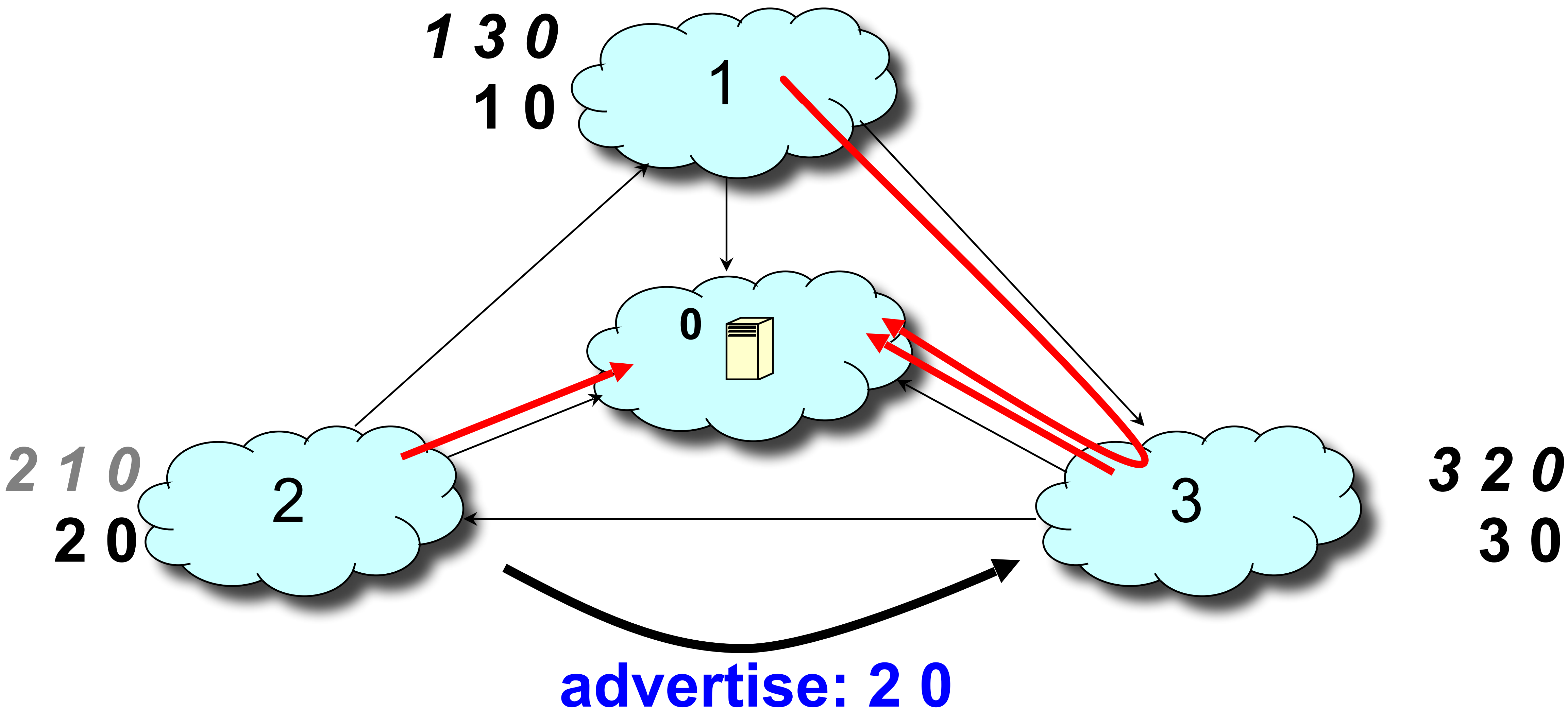


Step-by-step Policy Oscillation

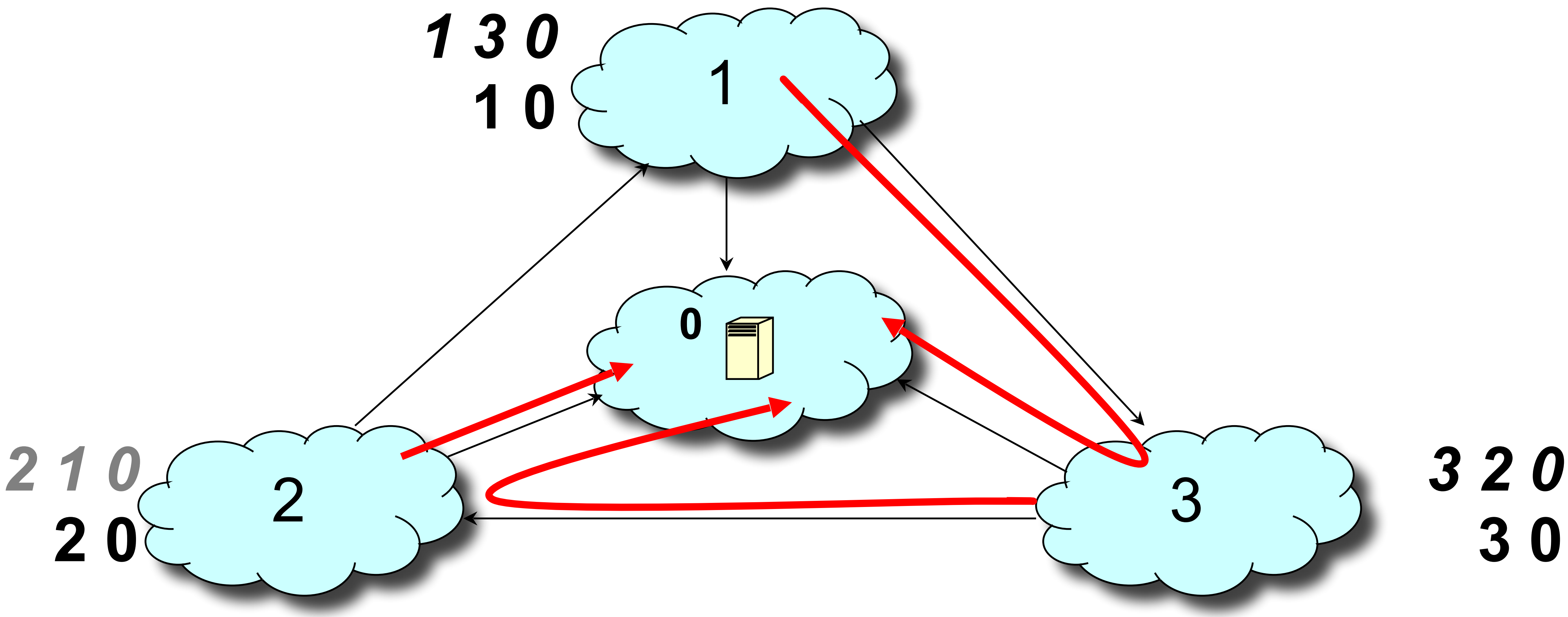


Step-by-step Policy Oscillation

2 **advertises** its path 2 0 to 3

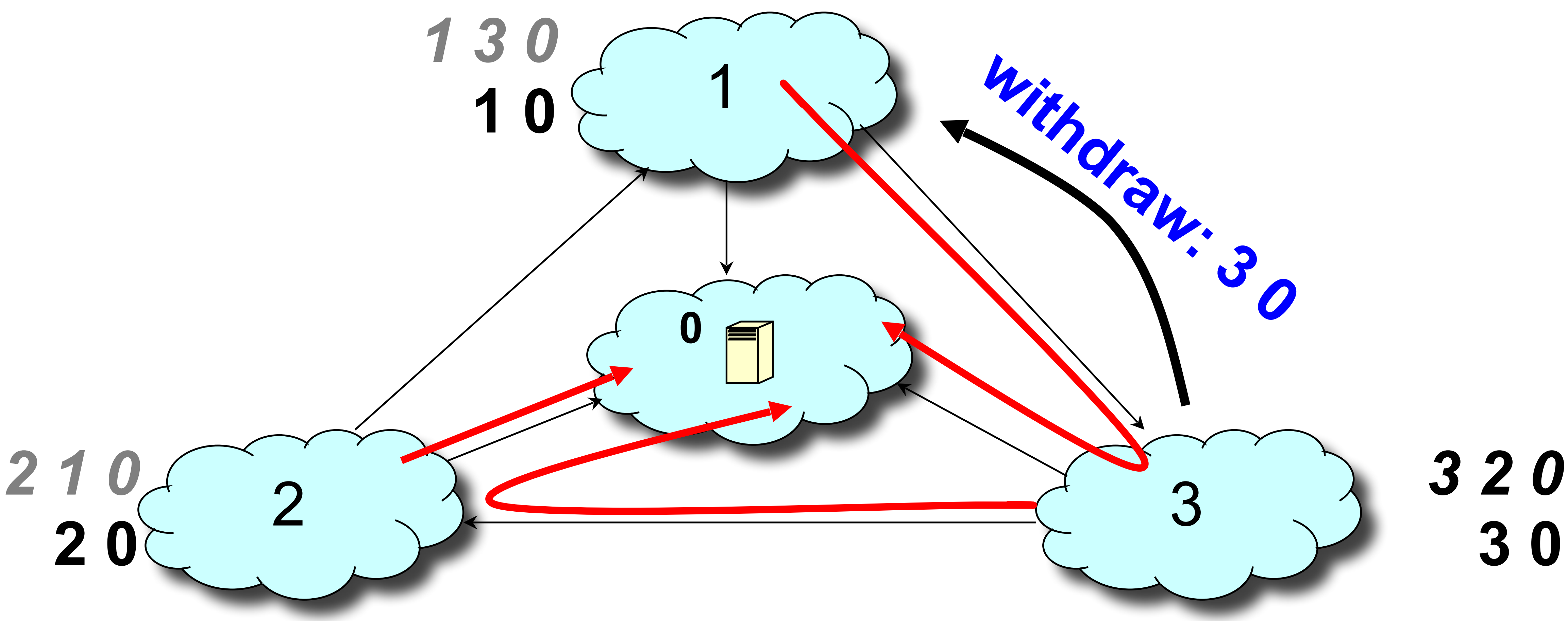


Step-by-step Policy Oscillation

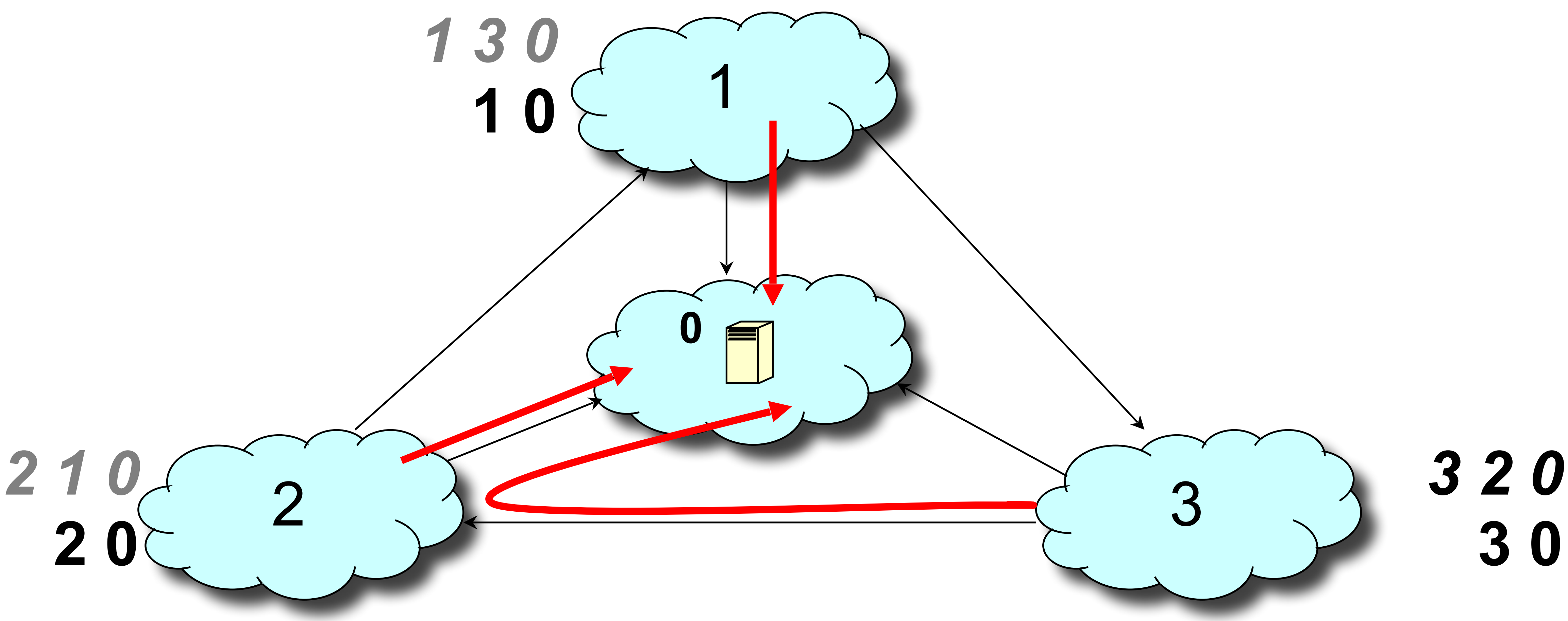


Step-by-step Policy Oscillation

3 **withdraws** its path 3 0 from 1

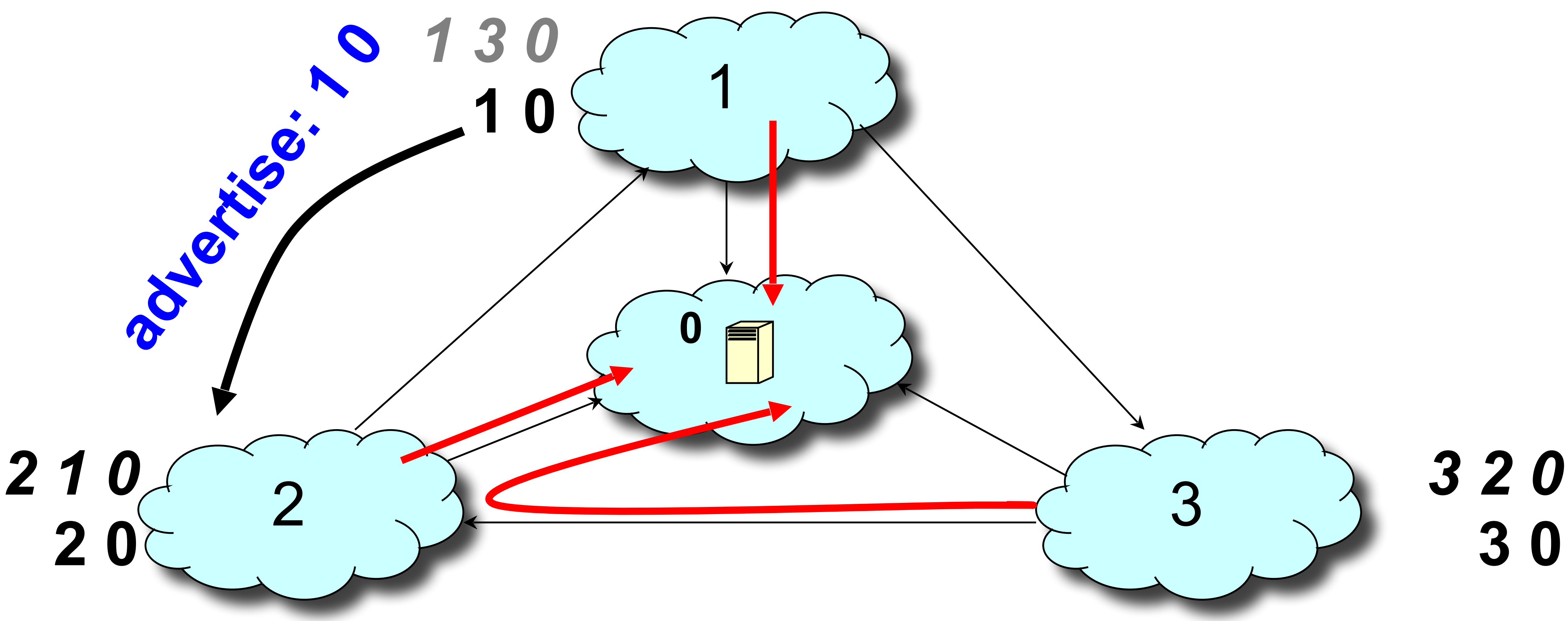


Step-by-step Policy Oscillation

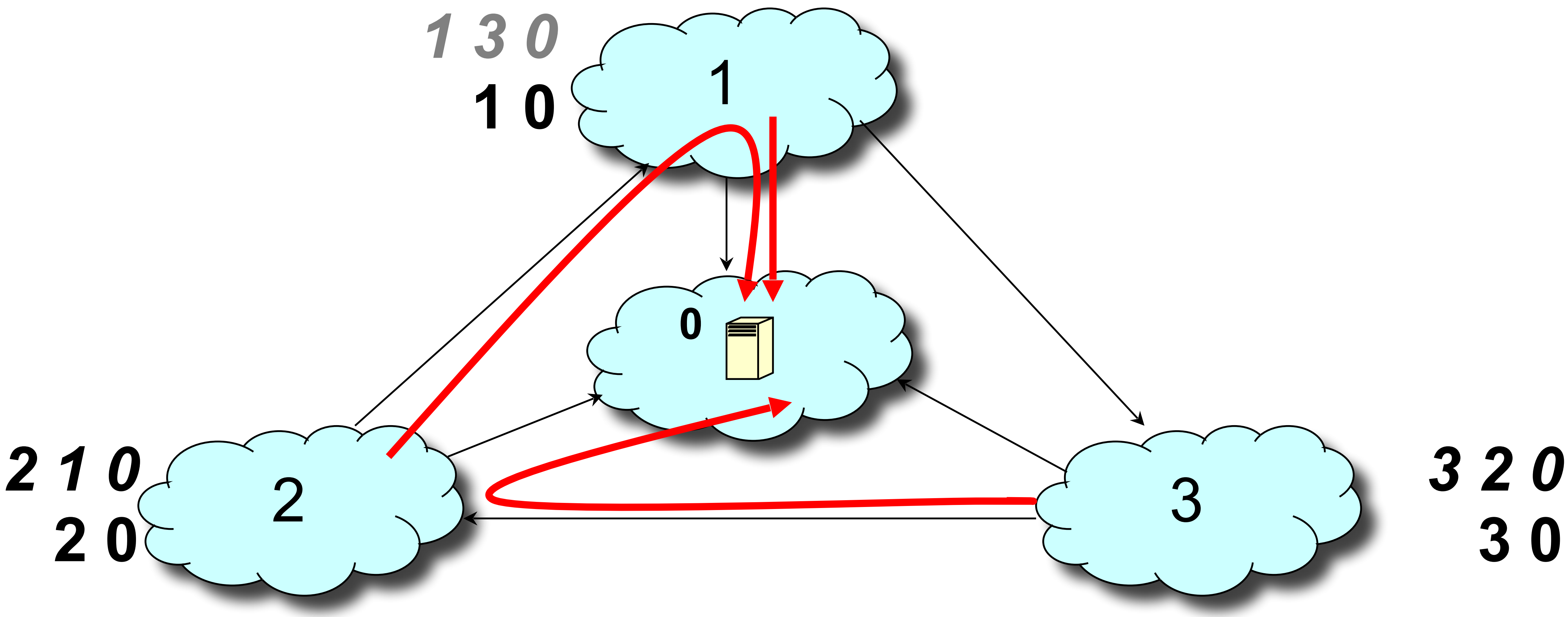


Step-by-step Policy Oscillation

1 **advertises** its path 1 0 to 2

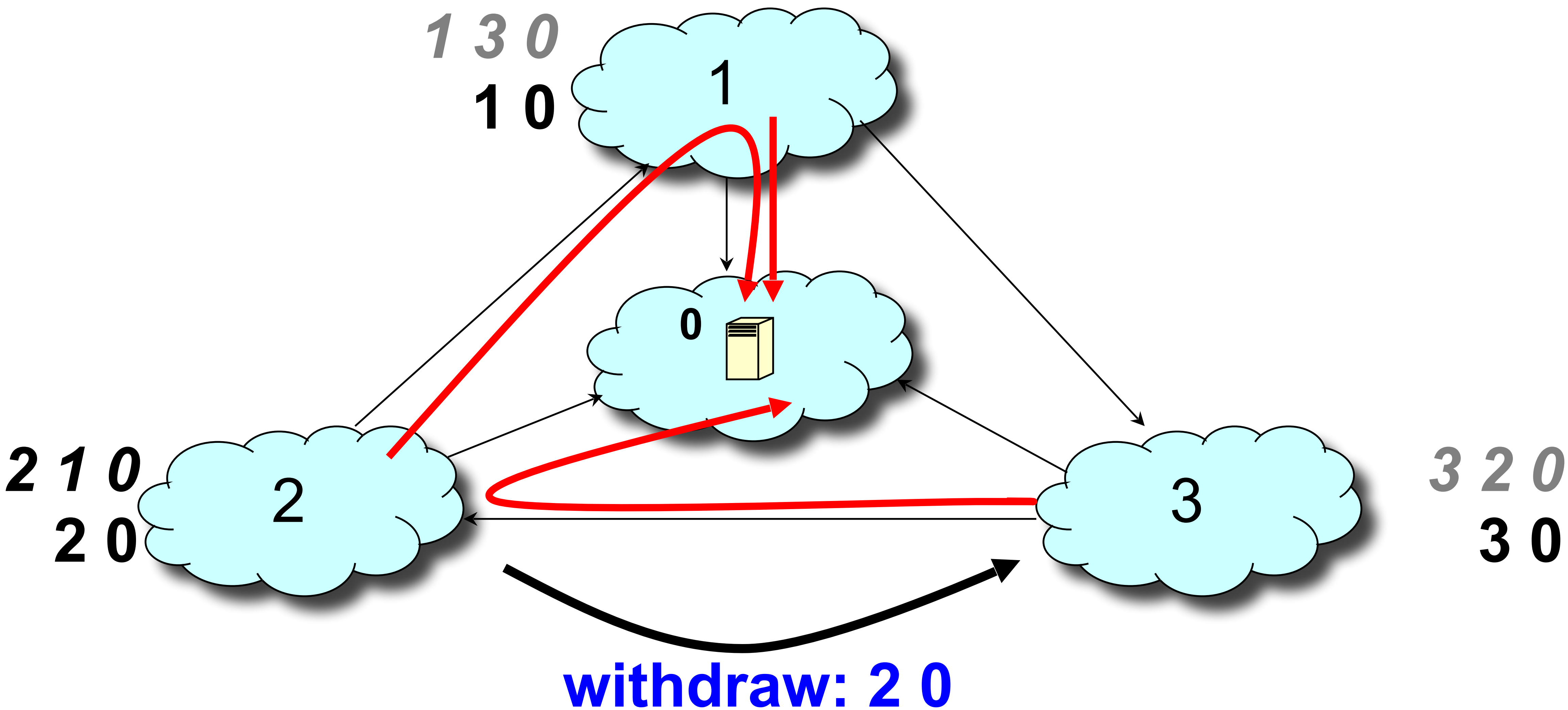


Step-by-step Policy Oscillation

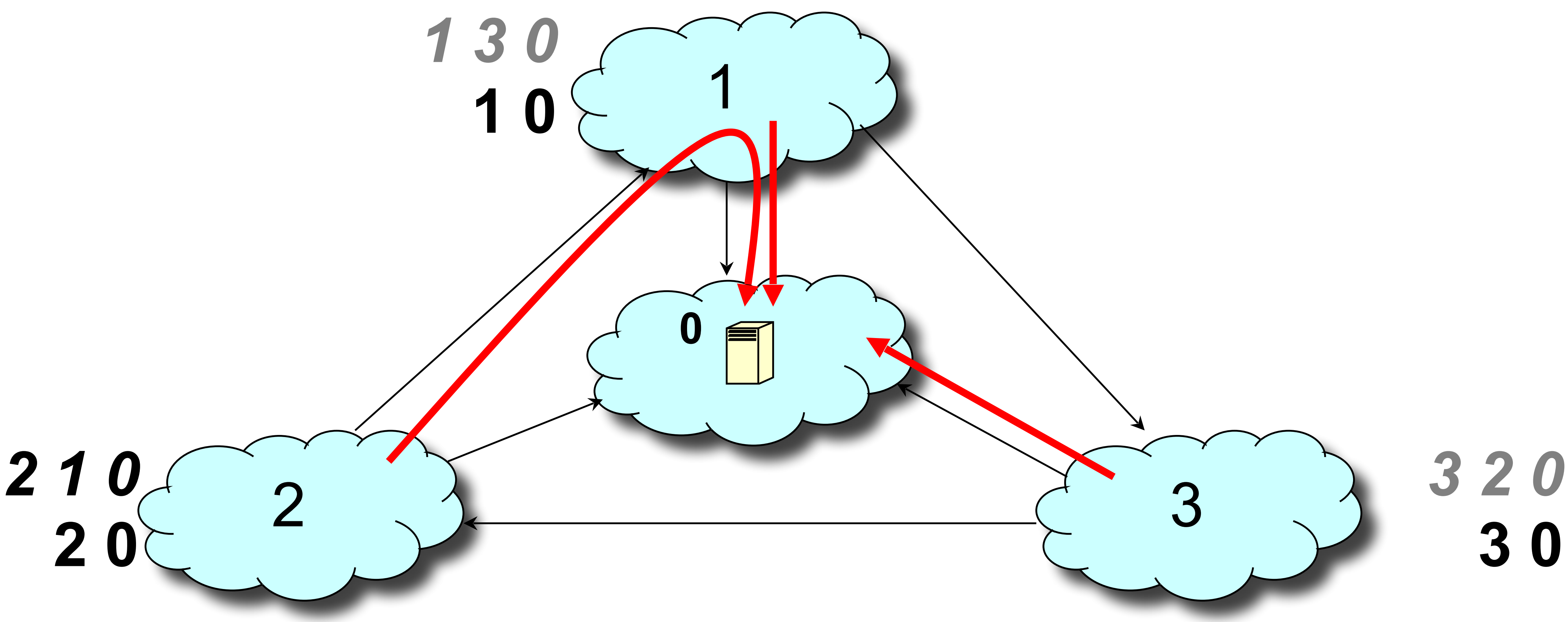


Step-by-step Policy Oscillation

2 **withdraws** its path 2 0 from 3

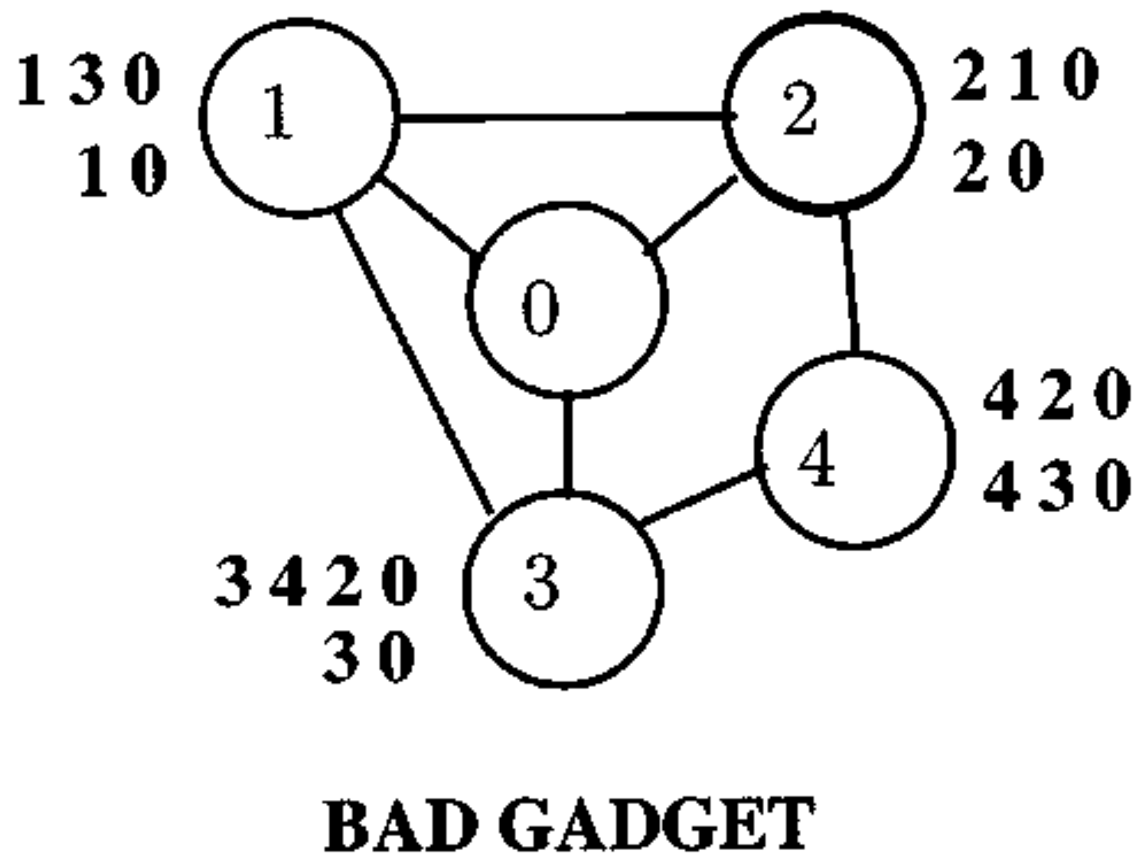


Step-by-step Policy Oscillation



We are back to where we started!

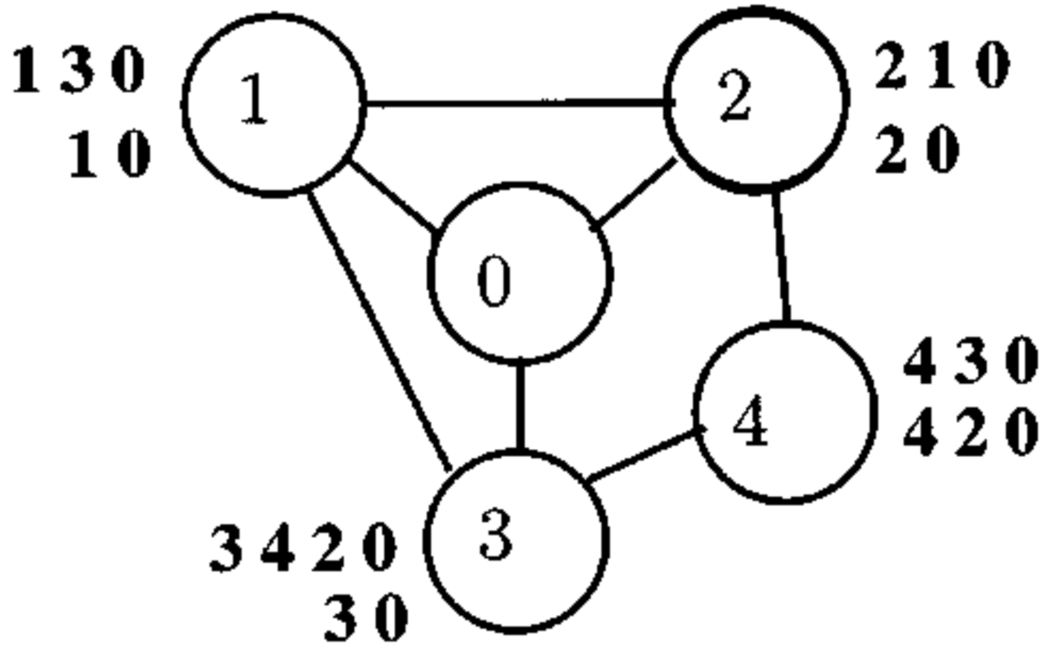
BGP Example (Persistent Loops)



	1	2	3	4
R1	10	**20**	30	-
R2	10	20	30	**420**
R3	**10**	20	3420	420
R4	10	**210**	3420	420
R5	10	210	3420	**-**
R6	10	210	**30**	-
R7	**130**	210	30	-
R8	130	**20**	30	-
R9	130	20	30	**420**
R10	130	20	**3420**	420
R11	10	20	3420	420

BGP Example (Bad bad bad)

	1	2	3	4
R1	10	20	**30**	-
R2	10	20	**30**	430
R3	130	20	30	430



NAUGHTY GADGET

	1	2	3	4
R1	10	**20**	30	-
R2	10	20	30	**420**
R3	**10**	20	3420	420
R4	10	**210**	3420	420
R5	10	210	3420	**-**
R6	10	210	**30**	-
R7	**130**	210	30	-
R8	130	**20**	30	-
R9	130	20	30	**420**
R10	130	20	**3420**	420
R11	10	20	3420	420

Convergence

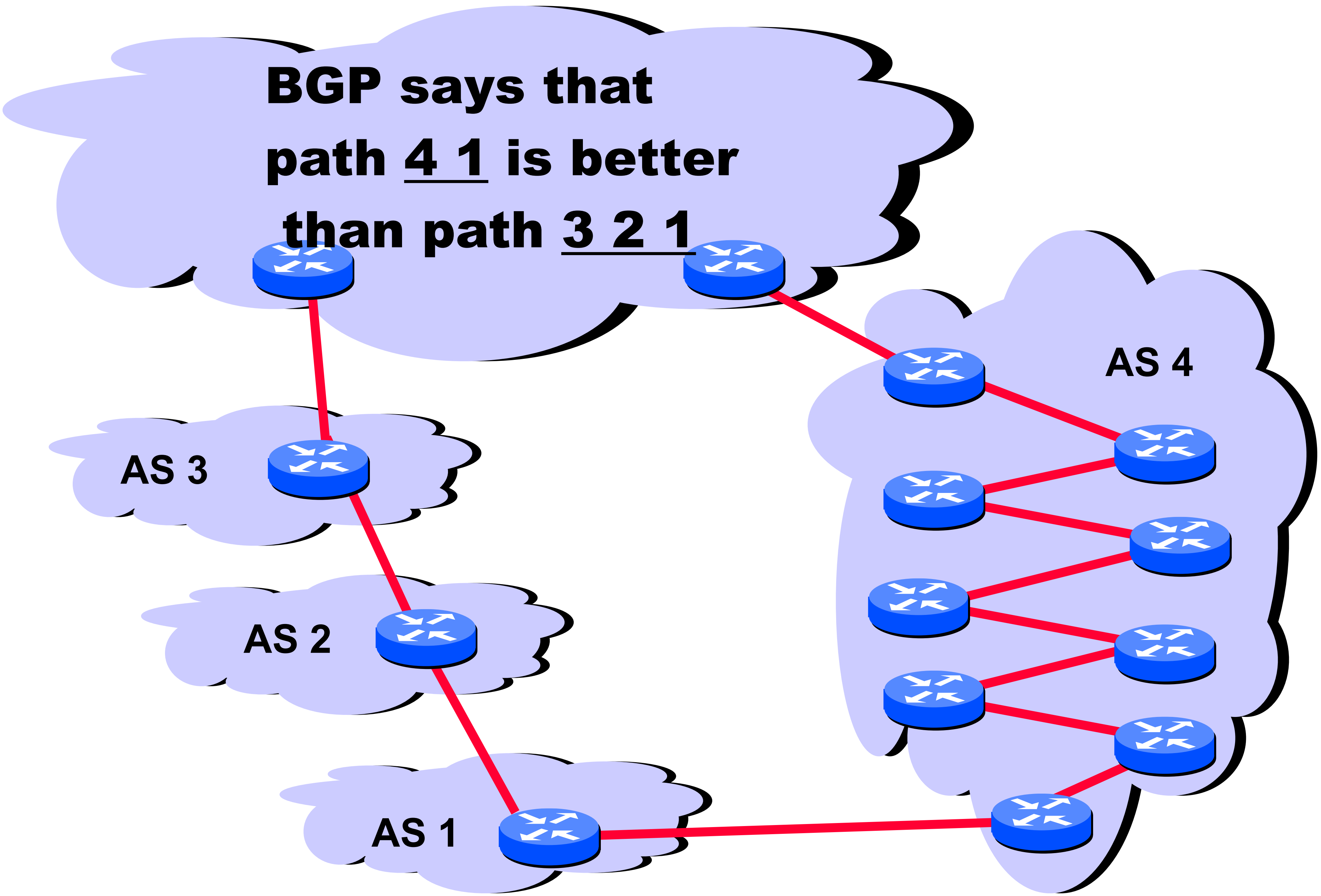
- If all AS policies follow Gao-Rexford rules,
  - Then BGP is guaranteed to converge (safety)
- For arbitrary policies, BGP may fail to converge!
- Why should this trouble us?

Performance Non-Issues

- Internal Routing
  - Domains typically use "hot potato" routing
  - Not always optimal, but economically expedient
- Policy not about performance
  - So policy-chosen paths aren't shortest
- AS path length can be misleading
  - 20% of paths inflated by at least 5 router hops

Performance (example)

- AS path length can be misleading
  - An AS may have many router-level hops

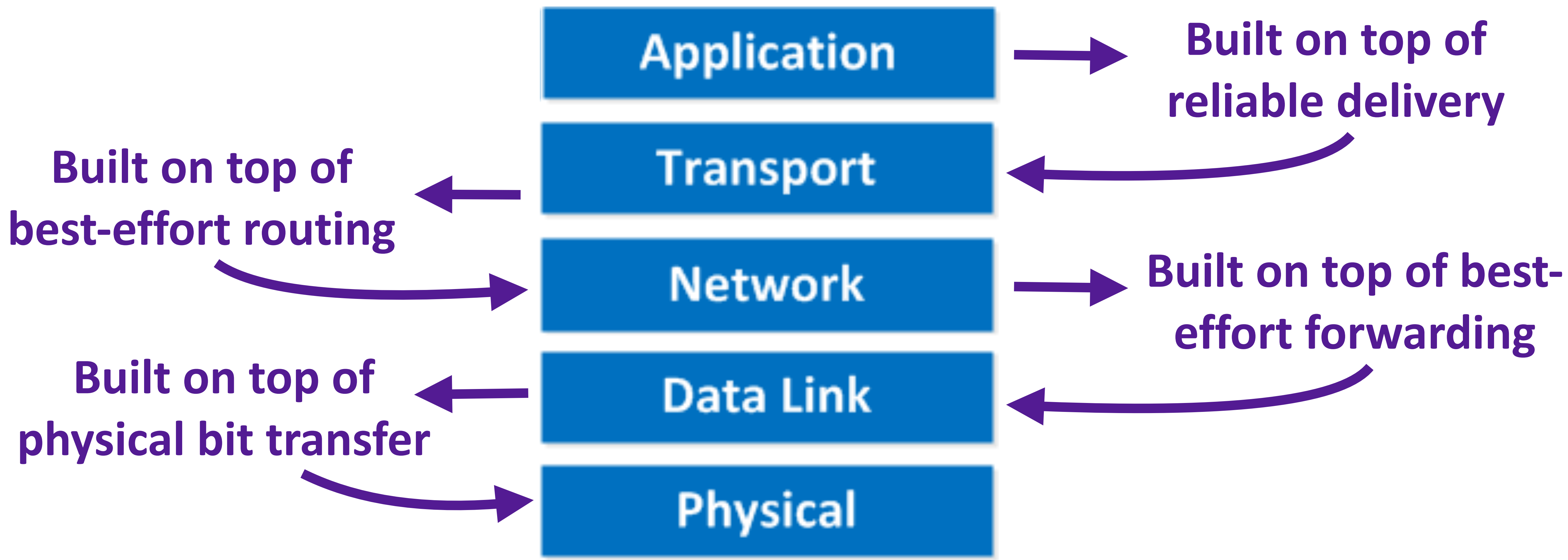


Performance: Real Issue

Slow Convergence

- BGP outages are biggest source of Internet problems
- Labovitz et al. *SIGCOMM'97*
  - 10% of routes available less than 95% of the time
  - Less than 35% of routes available 99.99% of the time
- Labovitz et al. *SIGCOMM 2000*
  - 40% of path outages take 30+ minutes to repair
- But most popular paths are very stable

Where are we?



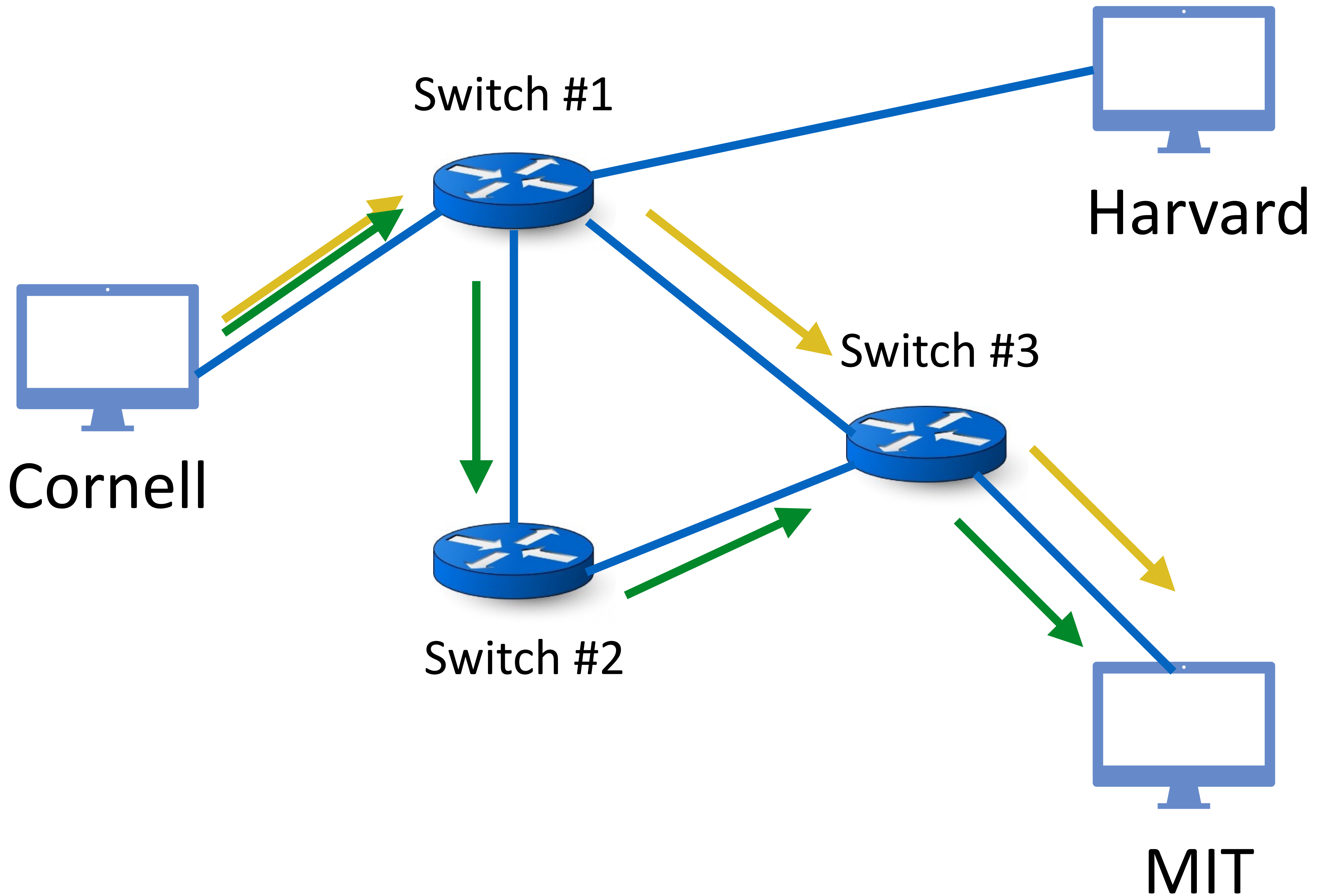
Application
Transport
Network
Data Link
Physical
Built on top of reliable delivery
Built on top of best-effort routing
Built on top of best-effort forwarding
Built on top of physical bit transfer

Switch/Router Architecture

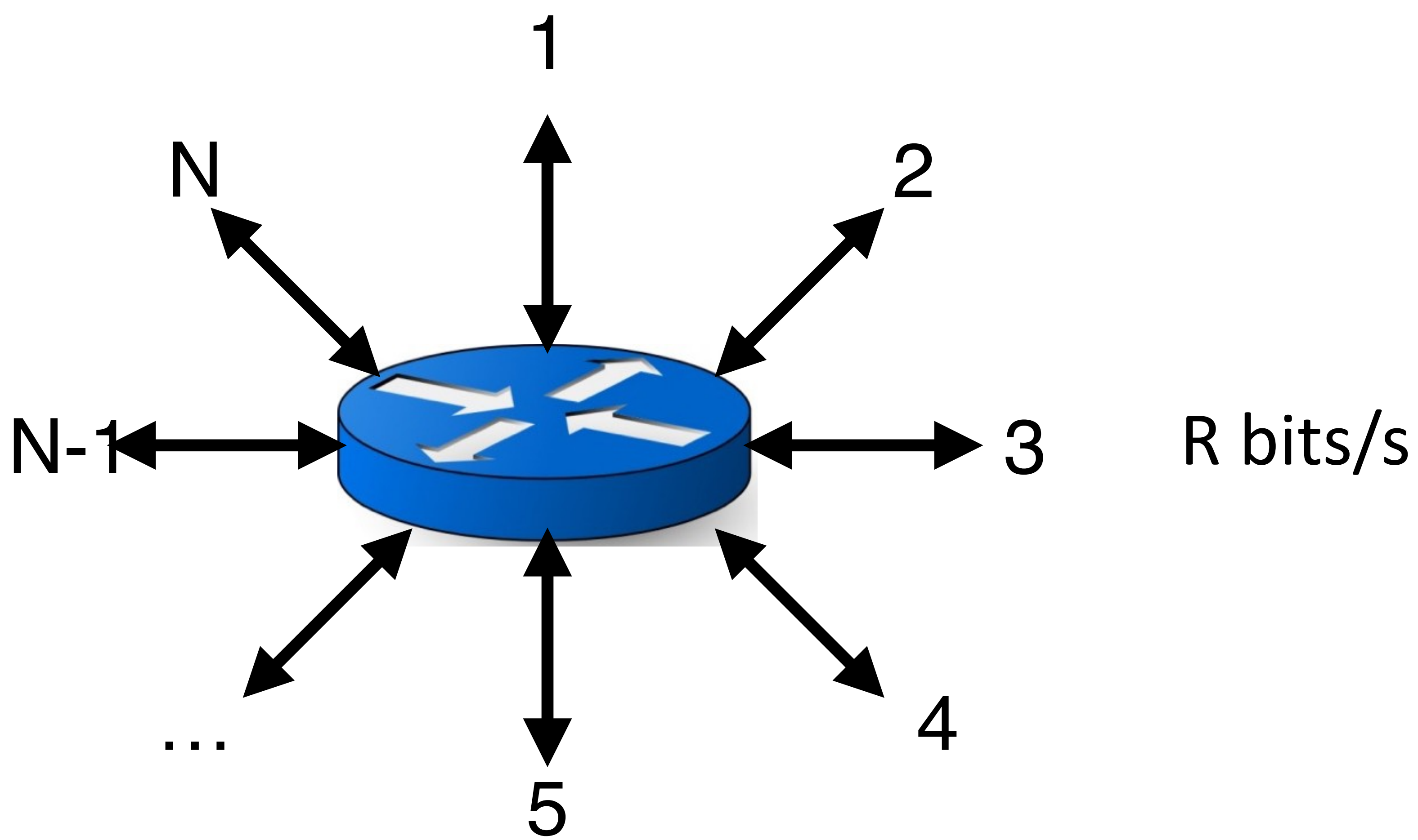
IP Routers and Switches (used interchangeably today)

- Core building block of Internet infrastructure
- $120B+ industry
- Vendors: Cisco, Huawei, Juniper, Alcatel-Lucent (account for >90%)

Recap: Routers Forward Packets

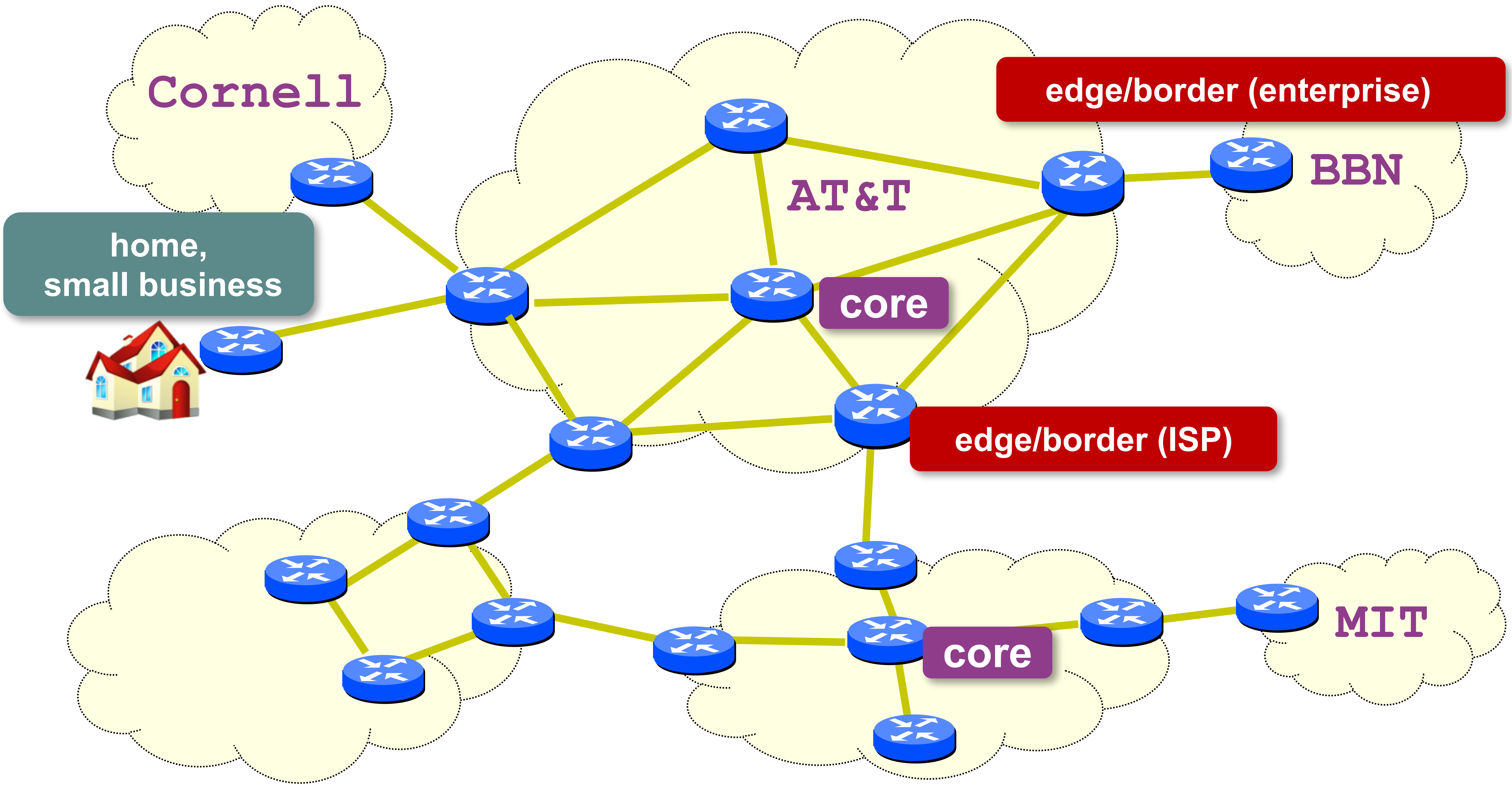


Router Definitions

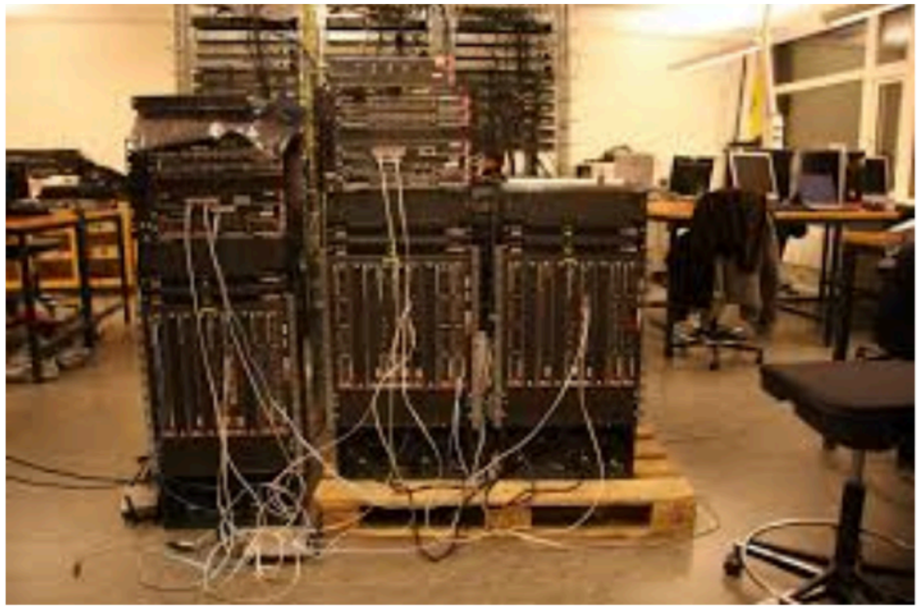


- N = No. Of external router ports
- R = bandwidth (“line rate”) of a port
- Router capacity = NxR

Networks and Routers



Examples of Routers (core)

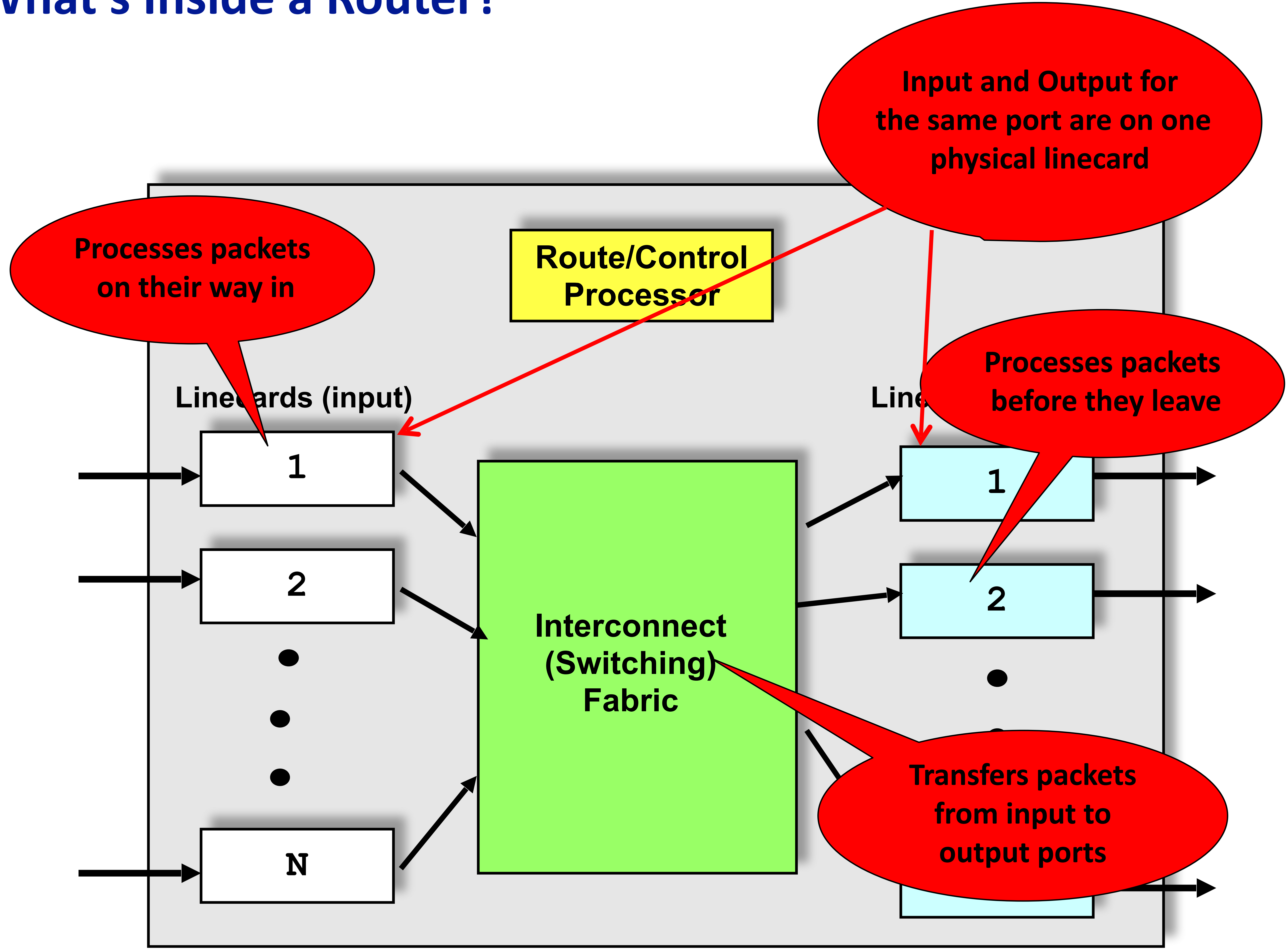


- **Core:** Cisco CRS
  - R = 10/40/100 Gbps
  - NR = 922 Tbps
  - Netflix: 0.7 GB/hr (1.5Mb/s)
  - ~600 million concurrent Netflix users

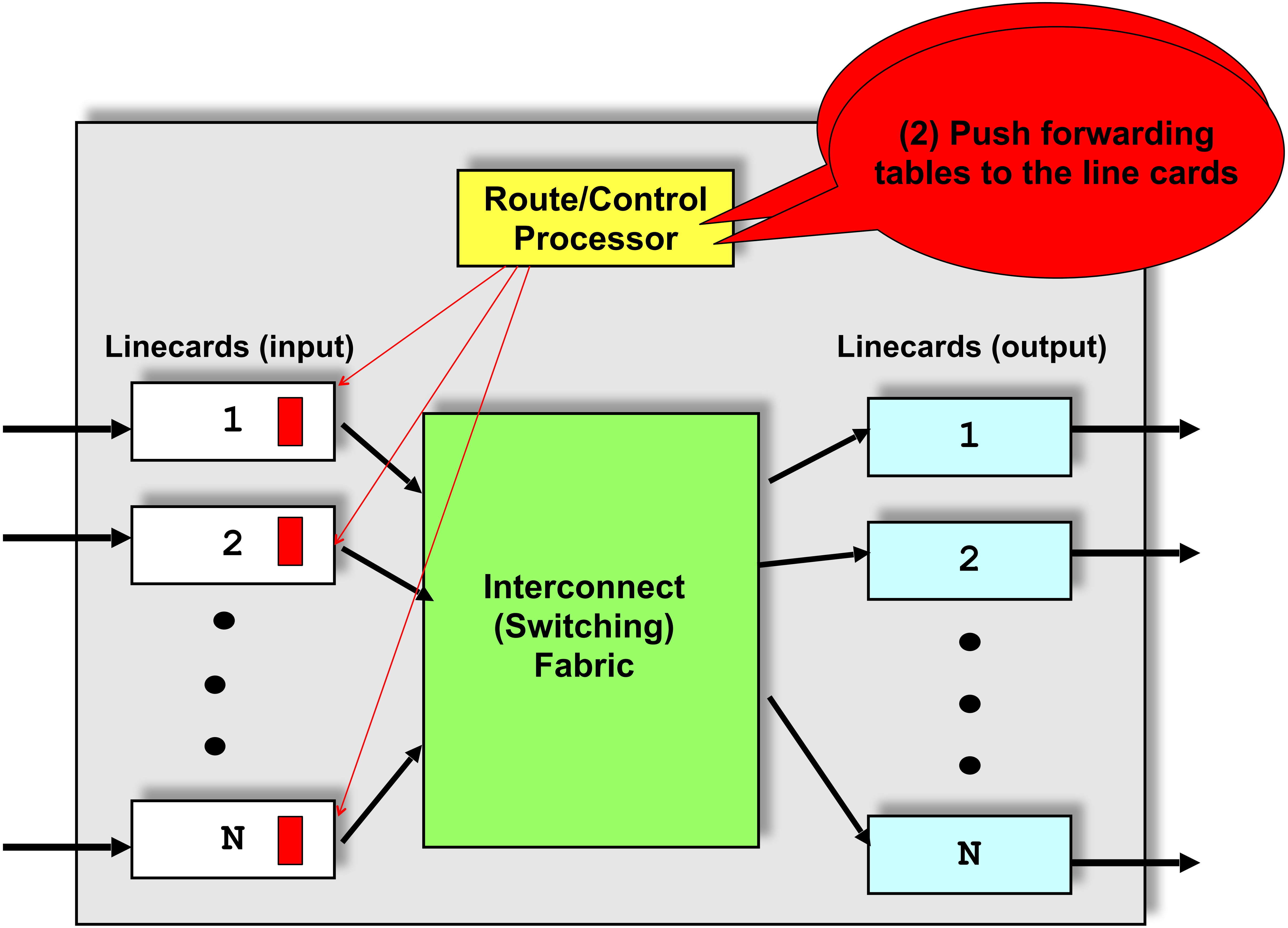
- **Edge (ISP):** Cisco ASR
  - R = 1/10/40 Gbps
  - NR = 120 Gbps

- **Edge (enterprise):** Cisco 3945E
  - R = 10/100/1000 Mbps
  - NR < 10 Gbps

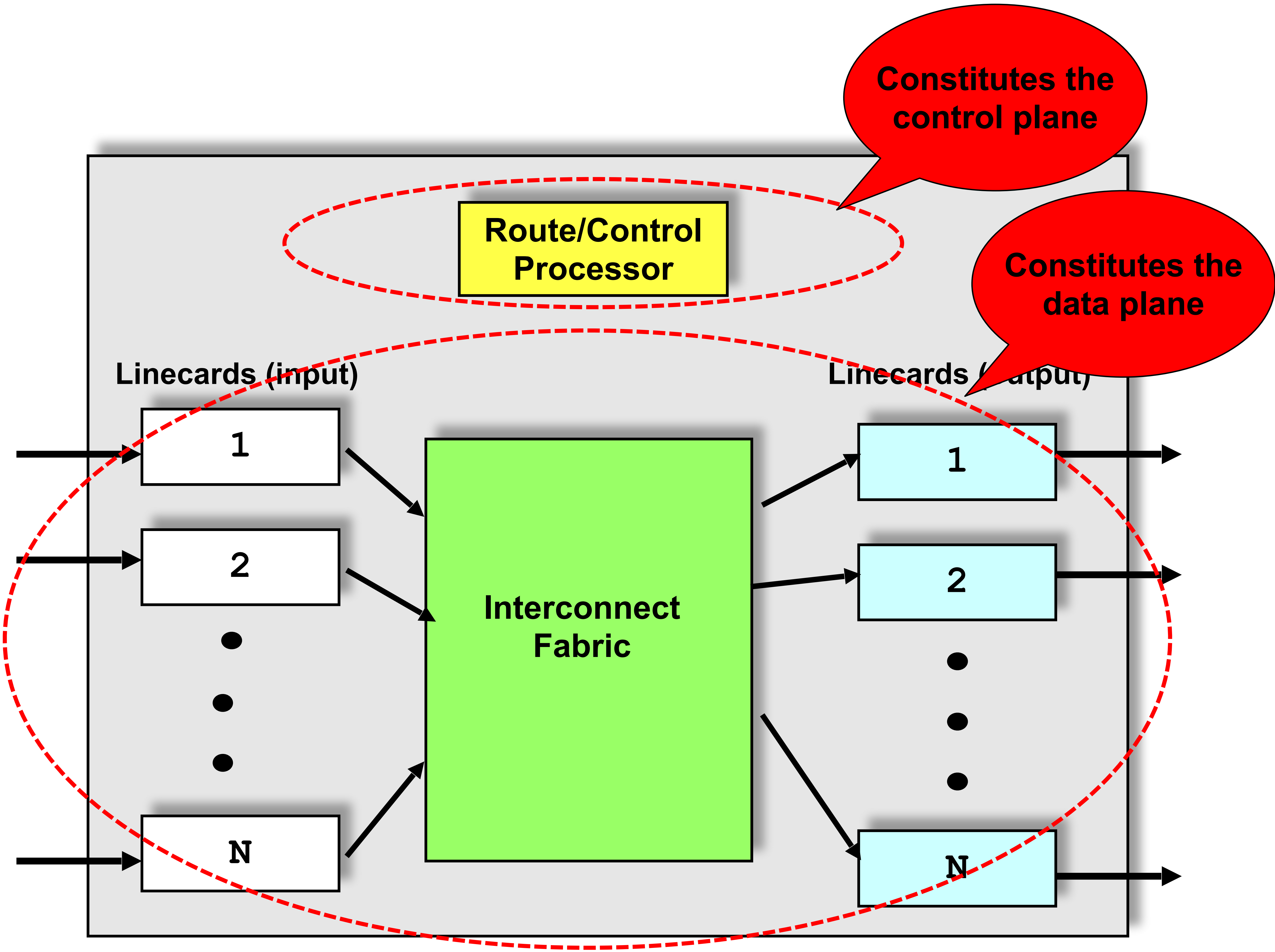
What's Inside a Router?



What's Inside a Router?



What’s Inside a Router?



Input Line Cards: Tasks

- Receive incoming packets (physical layer stuff)
- Update the IP header
    - TTL, Checksum (maybe some other fields)
- Lookup the output port for the destination IP address
- Queue the packet at the switch fabric

Challenge: Speed!

- 100B packets @ 40Gbps => packet every 20 nano secs!

- Typically implemented with specialized hardware
    - ASICs, specialized “network processors”

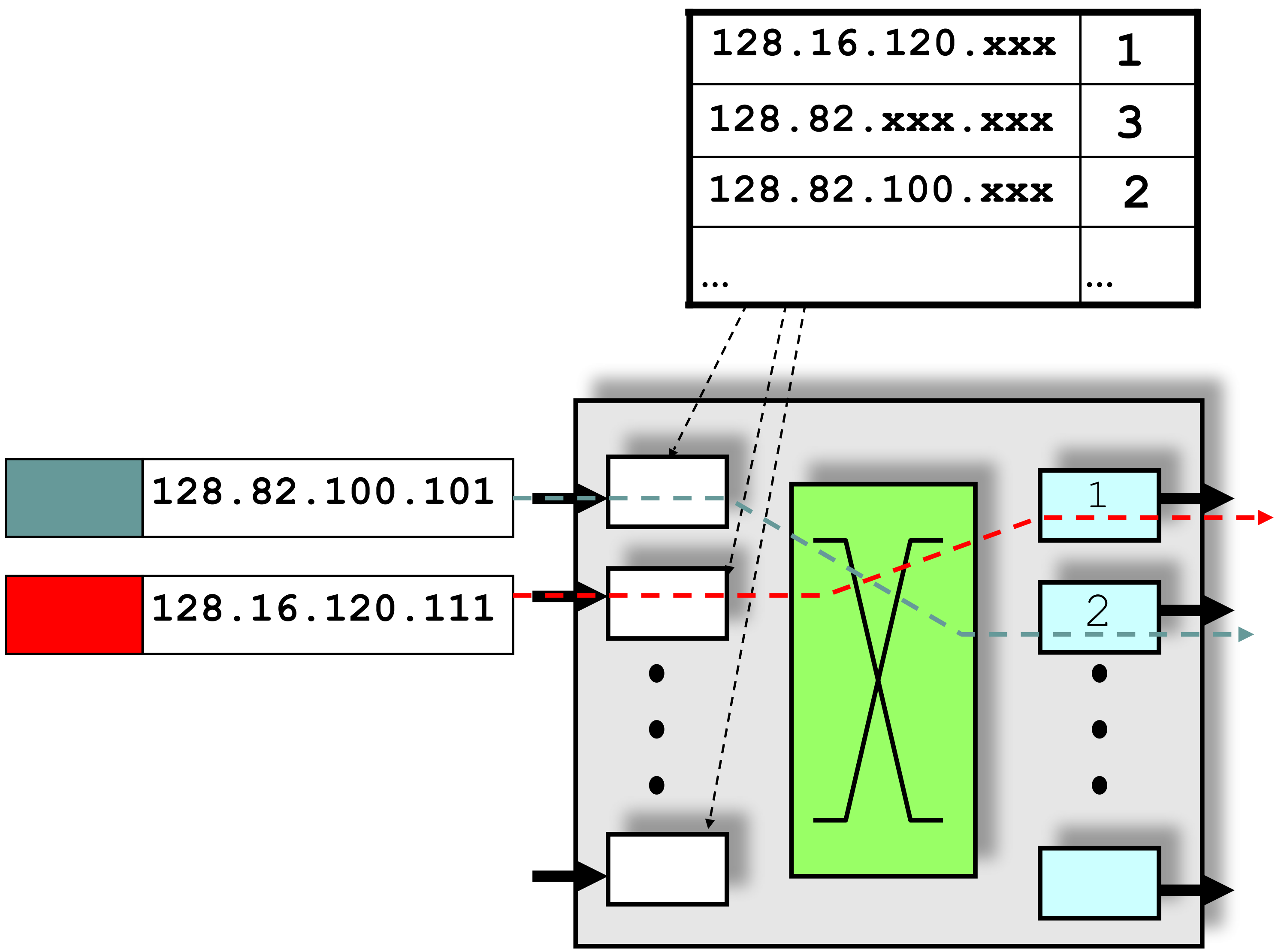
Looking up the Output Port

- Upon receiving a packet
  - Inspect the destination IP address in the header
  - Index into the routing/forwarding table
  - If no match, select the **default route**
  - Forward packet out appropriate interface

- **Default route**
  - Configured to cover cases where no matches
  - Allows small tables at edge (w/o routing algorithms)
    - **if it isn't on my subnet, send it to my ISP**

Scaling the Lookup

- Recall: For scalability, addresses are **aggregated**
- **Longest Prefix match**
  - Find the entry with matching "longest prefix" with destination address



Finding a Match

- Incoming packet destination: 201.143.7.0

Prefix	Port
201.143.0.0/22	Port 1
201.143.4.0.0/24	Port 2
201.143.5.0.0/24	Port 3
201.143.6.0/23	Port 4

Finding a Match: Covert to Binary

- Incoming packet destination: 201.143.7.0

11001001	10001111	00000111	11010010

Routing Table

201.143.0.0/22

11001001	10001111	000000 - -	- - - - - - - -

201.143.4.0/24

11001001	10001111	00000100	- - - - - - - -

201.143.5.0/24

11001001	10001111	00000101	- - - - - - - -

201.143.6.0/23

11001001	10001111	0000011-	- - - - - - - -

Finding a Match: Covert to Binary

- Incoming packet destination: 201.143.7.0

11001001	10001111	00000111	11010010

Routing Table

201.143.0.0/22

11001001	10001111	000000 - -	- - - - - - - -

201.143.4.0/24

11001001	10001111	00000100	- - - - - - - -

201.143.5.0/24

11001001	10001111	00000101	- - - - - - - -

201.143.6.0/23

11001001	10001111	0000011-	- - - - - - - -

Finding a Match: Covert to Binary

- Incoming packet destination: 201.143.7.0

11001001	10001111	00000111	11010010

Routing Table

201.143.0.0/22

~~11001001~~	~~10001111~~	~~000000~~	

201.143.4.0/24

11001001	10001111	00000100	- - - - - - - -

201.143.5.0/24

11001001	10001111	00000101	- - - - - - - -

201.143.6.0/23

11001001	10001111	0000011-	- - - - - - - -

Longest Prefix Match

- Incoming packet destination: 201.143.7.0

11001001	10001111	00000**1**11	11010010

Routing Table

201.143.0.0/22

~~11001001~~	~~10001111~~	~~00000**0**--~~	~~- - - - - - - -~~

201.143.4.0/24

~~11001001~~	~~10001111~~	~~00000**1**00~~	~~- - - - - - - -~~

201.143.5.0/24

~~11001001~~	~~10001111~~	~~00000**1**01~~	~~- - - - - - - -~~

201.143.6.0/23

11001001	10001111	00000**1**1-	- - - - - - - -

Check an address against all destination prefixes and select the prefix it matches with on the most bits

Finding the Match Efficiently

- Testing each entry to find a match scales poorly
  - Roughly (number of entries) × (number of bits)

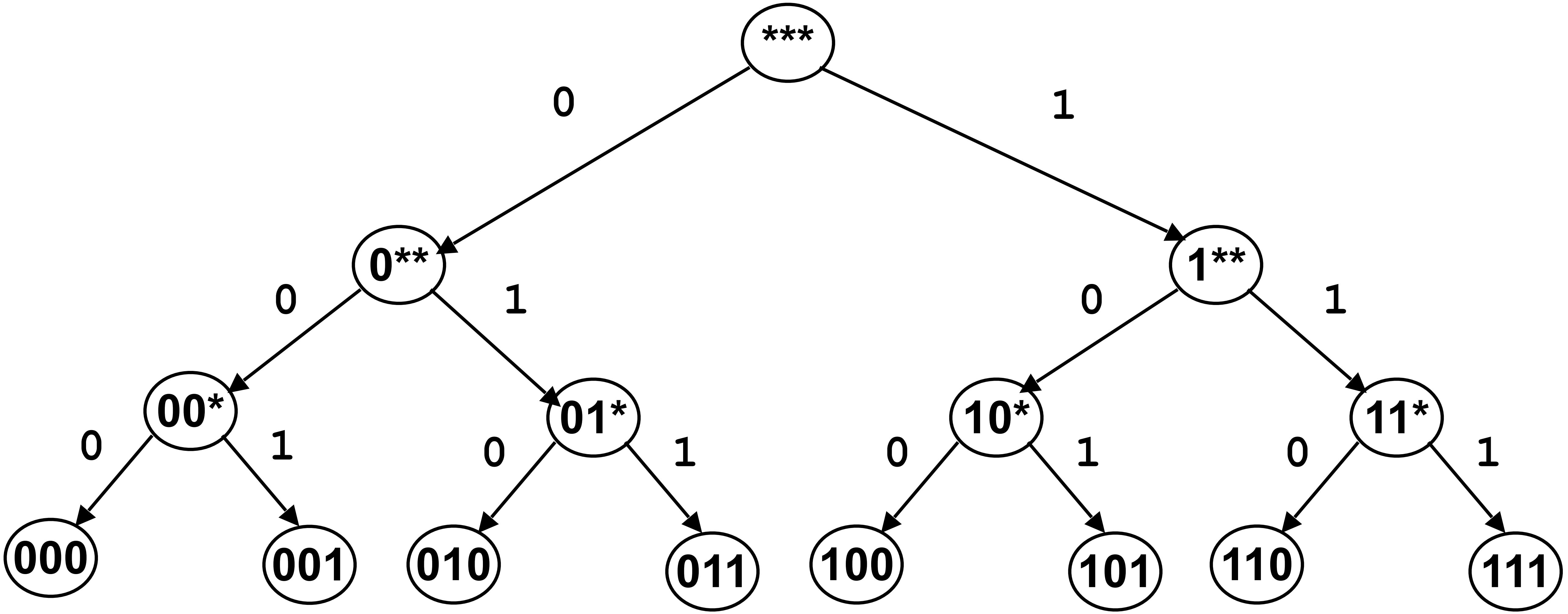
- Must leverage tree structure of binary strings
  - Set up tree-like data structure
  - Called a **TRIE**
    - We will briefly discuss it; more details in text
    - In case you are interested ….

Consider Four 3-Bit Prefixes

- Just focusing on the bits where all the action is....

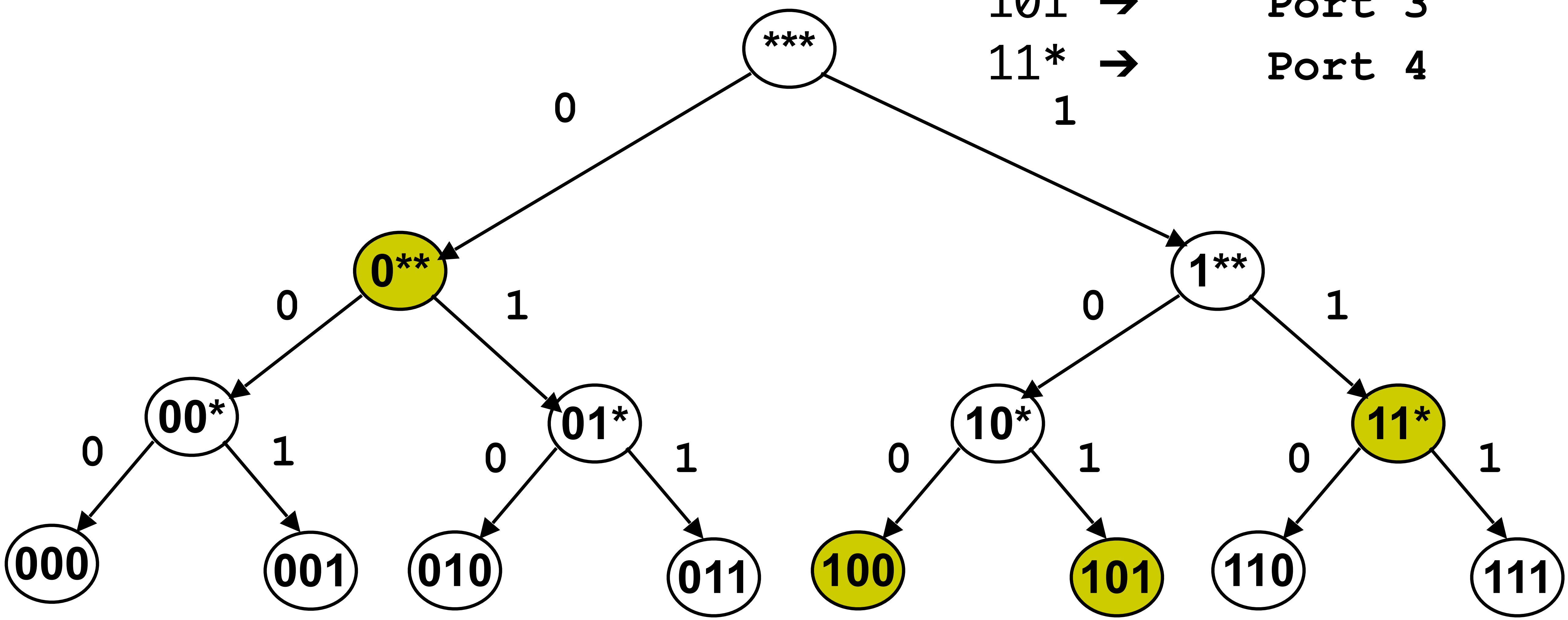
- 0** → Port 1
- 100 → Port 2
- 101 → Port 3
- 11* → Port 4

Tree Structure



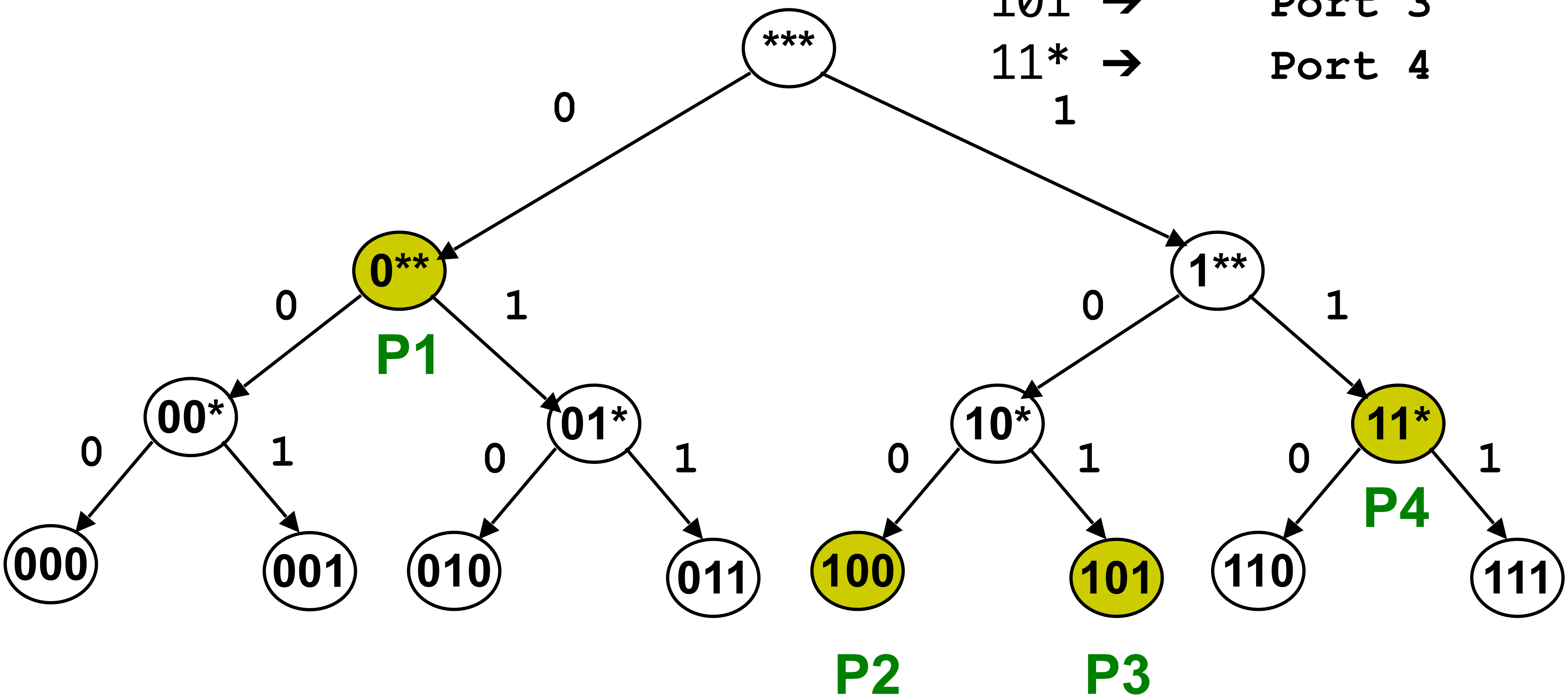
Walk Tree: Stop at Prefix Entries

0** → Port 1
100 → Port 2
101 → Port 3
11* → Port 4



Walk Tree: Stop at Prefix Entries

0**	➔	Port 1
100	➔	Port 2
101	➔	Port 3
11*	➔	Port 4



walking trees takes O(#bits)

Longest Prefix Match in Real Routers

- Real routers use far more advanced/complex solutions
    - But what we discussed is the starting point
- With many heuristics and optimizations that leverage real-world patterns
    - Some destinations more popular than others
    - Some ports lead to more destinations
    - Typical fix granularities

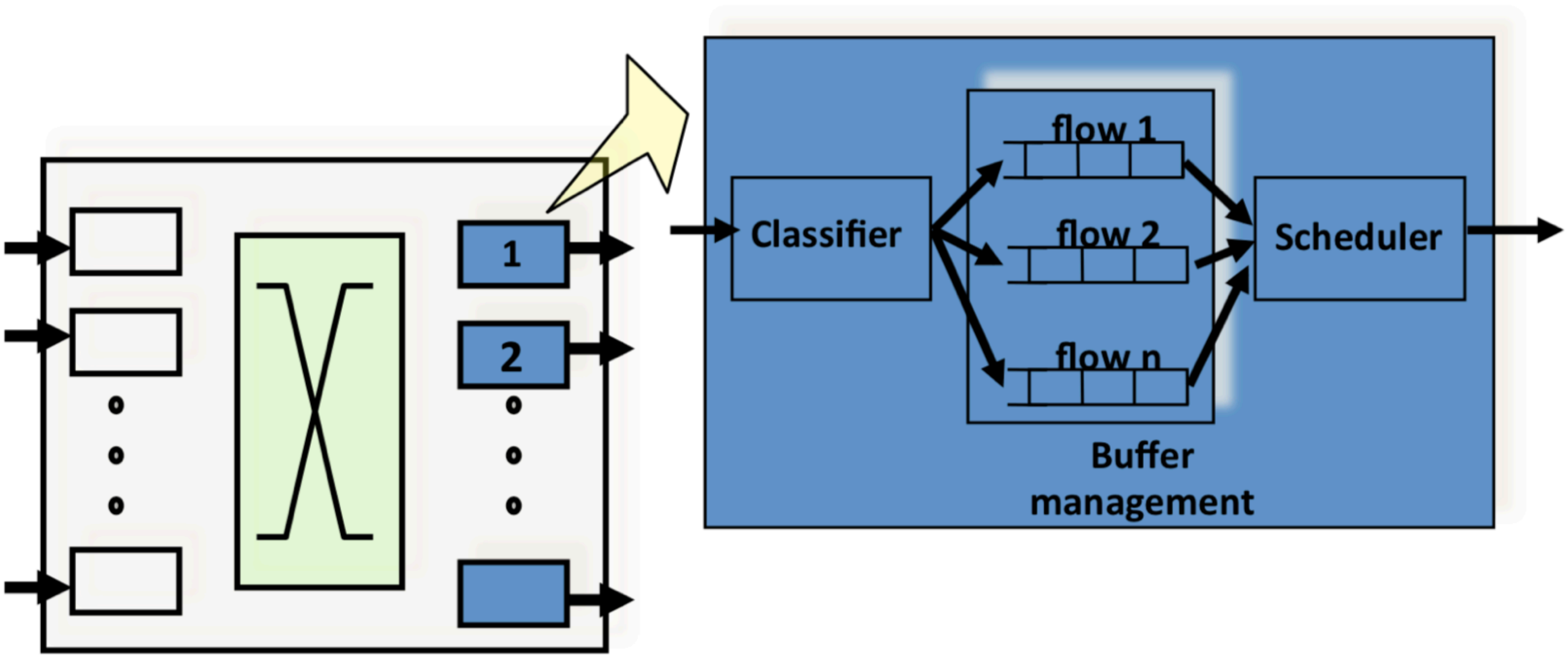
Recap: Input Linecards

- Main challenge is processing speed
    - But what we discussed is the starting point

- Tasks involved
    - Update packet header (easy)
    - Longest prefix match lookup on destinations address (harder)

- Mostly implemented with specialized hardware

Output Linecard



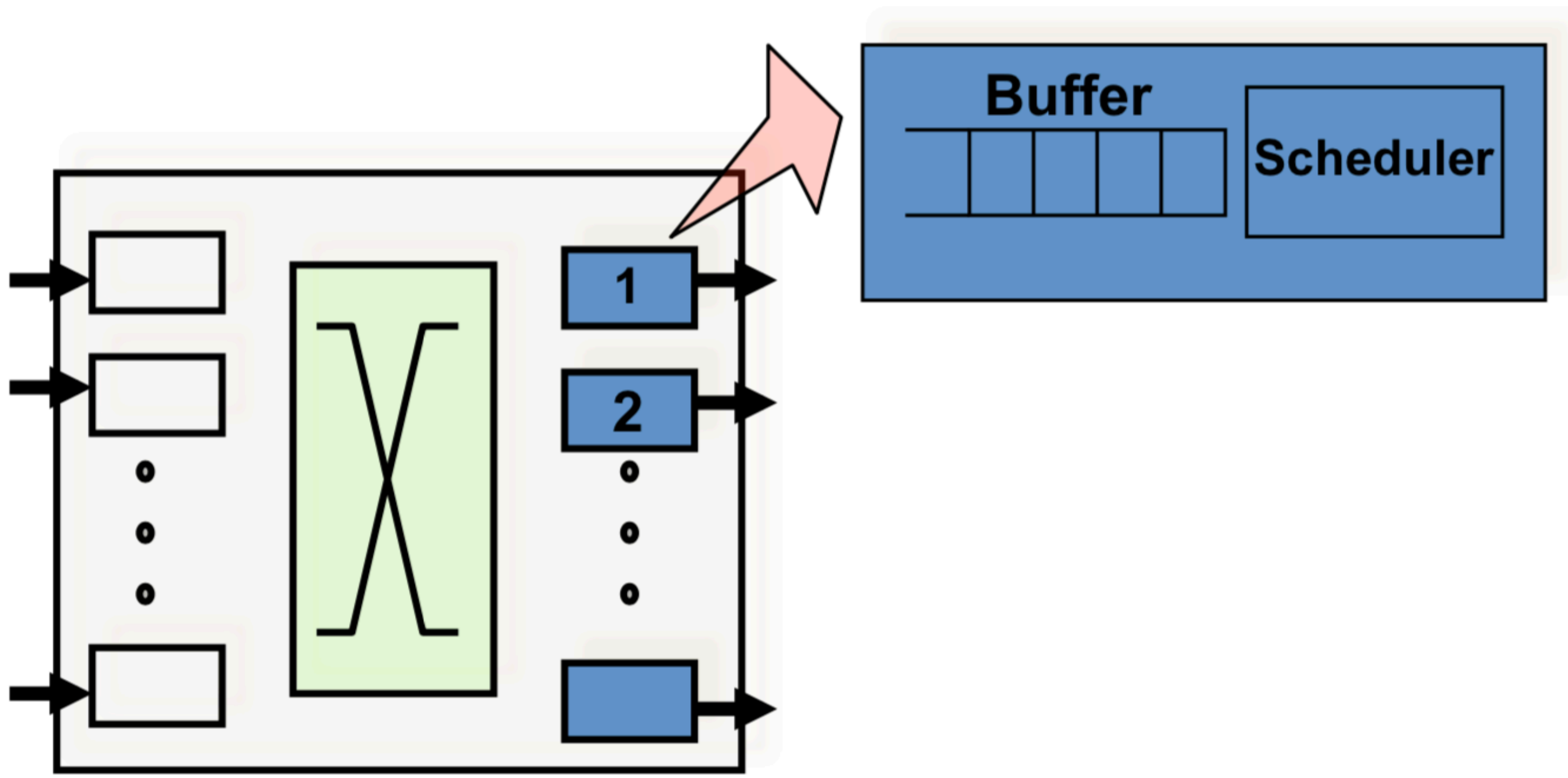
- **Packet Classification**: map each packet to a "flow"
    - Flow (for now): set of packets between two particular endpoints
- **Buffer Management**: decide when and which packet to drop
- **Scheduler**: decide when and which packet to transmit

Output Linecard

- **Packet Classification**: map each packet to a “flow”
  - Flow (for now): set of packets between two particular endpoints
- **Buffer Management**: decide when and which packet to drop
- **Scheduler**: decide when and which packet to transmit
- Used to implement various forms of policy
  - Deny all e-mail traffic from ISP X to Y (**access control**)
  - Route IP telephony traffic from X to Y via PHY_CIRCUIT (**policy**)
  - Ensure that no more than 50 Mbps are injected from ISP-X (**QoS**)

Simplest FIFO Router

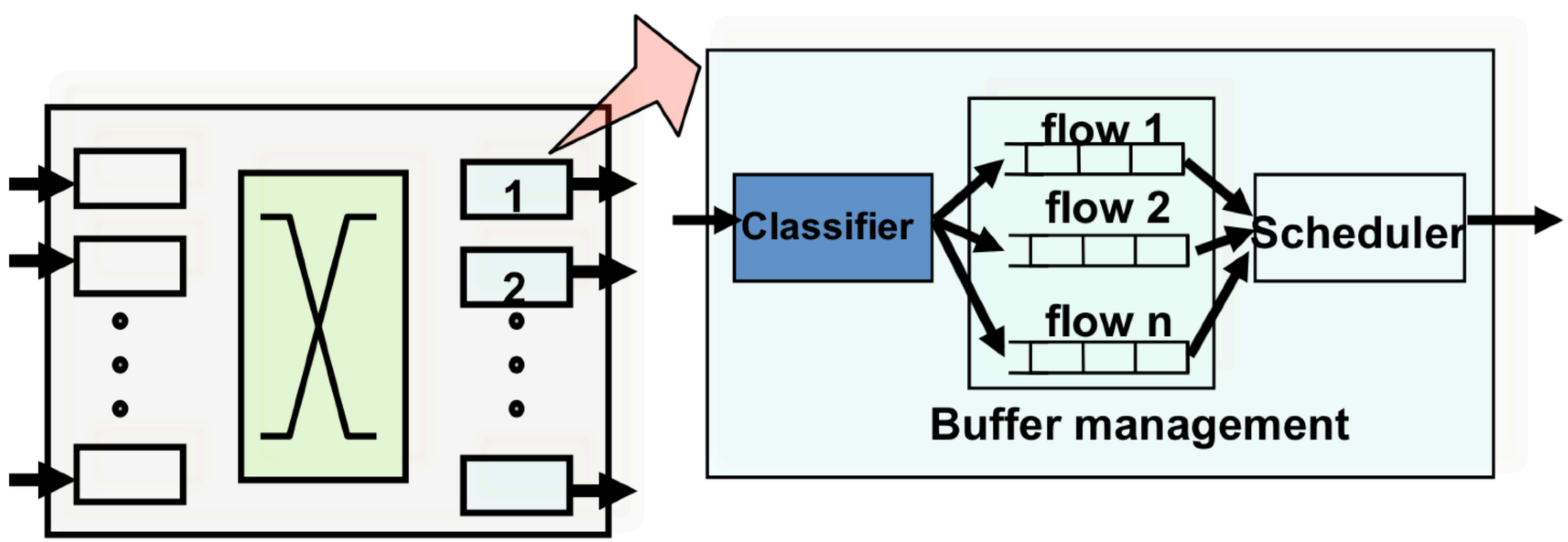
- **No classification**
- **Drop tail buffer management:** when buffer is full drop incoming packet
- **First In First Out (FIFO) Scheduling:** schedule packets in order of arrival



Packet Classification

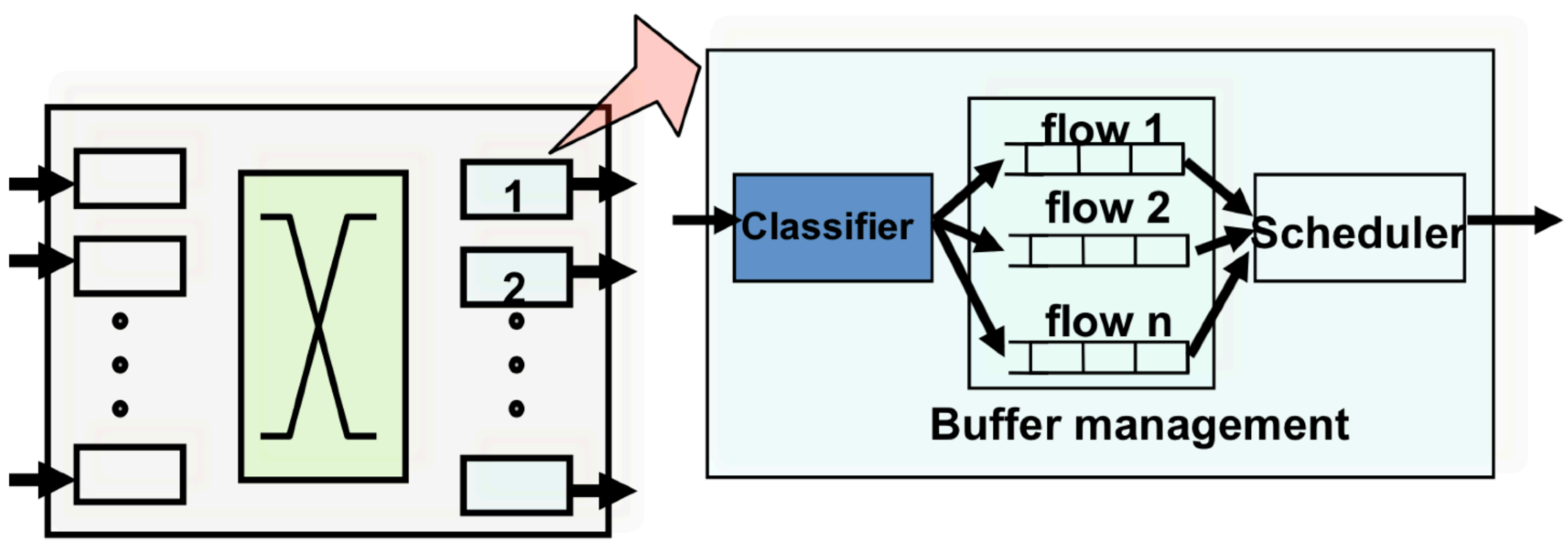
- Classify an IP packet based on the number of fields in the packet header
  - Source/destination IP address (32 bits)
  - Source/destination TCP port number (16 bits)
  - Type of Service (TOS) byte (8 bits)
  - Type of Protocol (8 bits)

- In general fields are specified by range
  - Classification requires a multi-dimensional range search



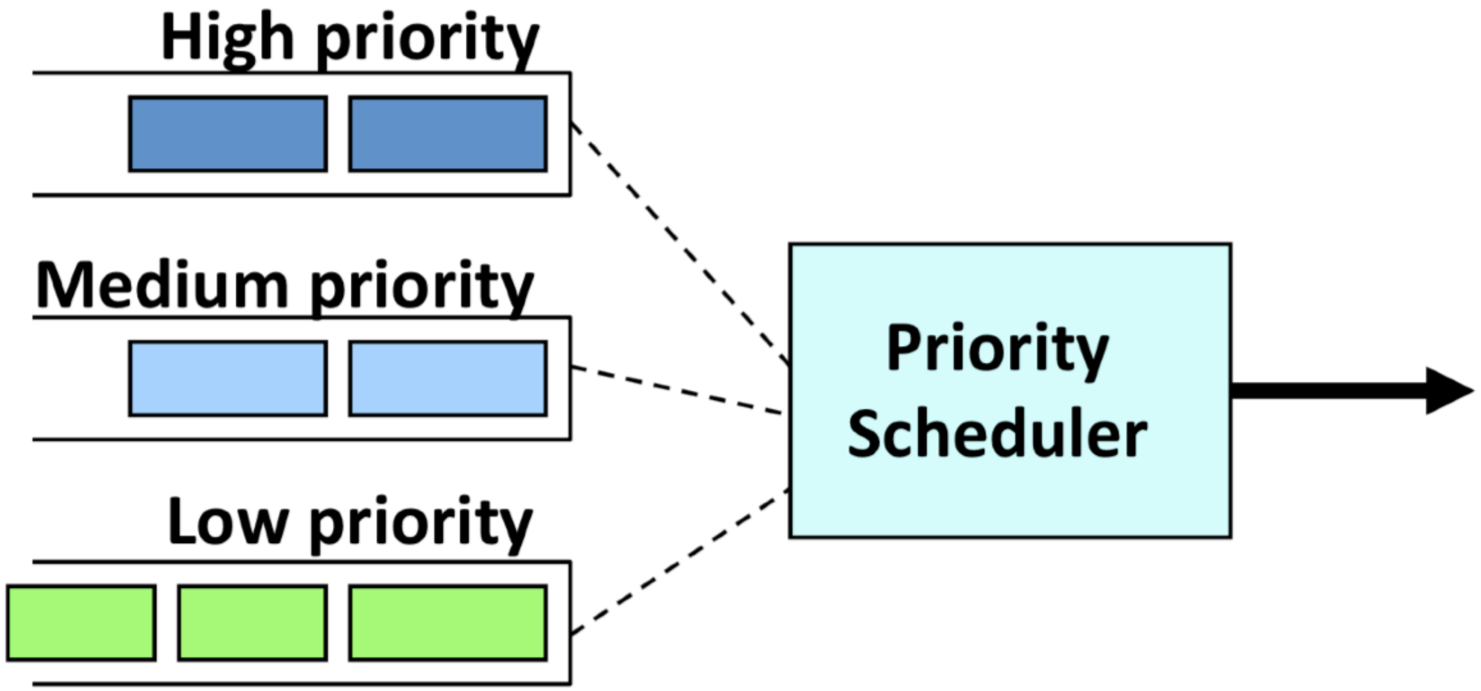
Scheduler

- One queue per flow
- Scheduler decides from which queue to send a packet
- Goals of scheduling algorithm
  - Fast!
  - Depends on the policy being implemented (fairness, priority, etc.)



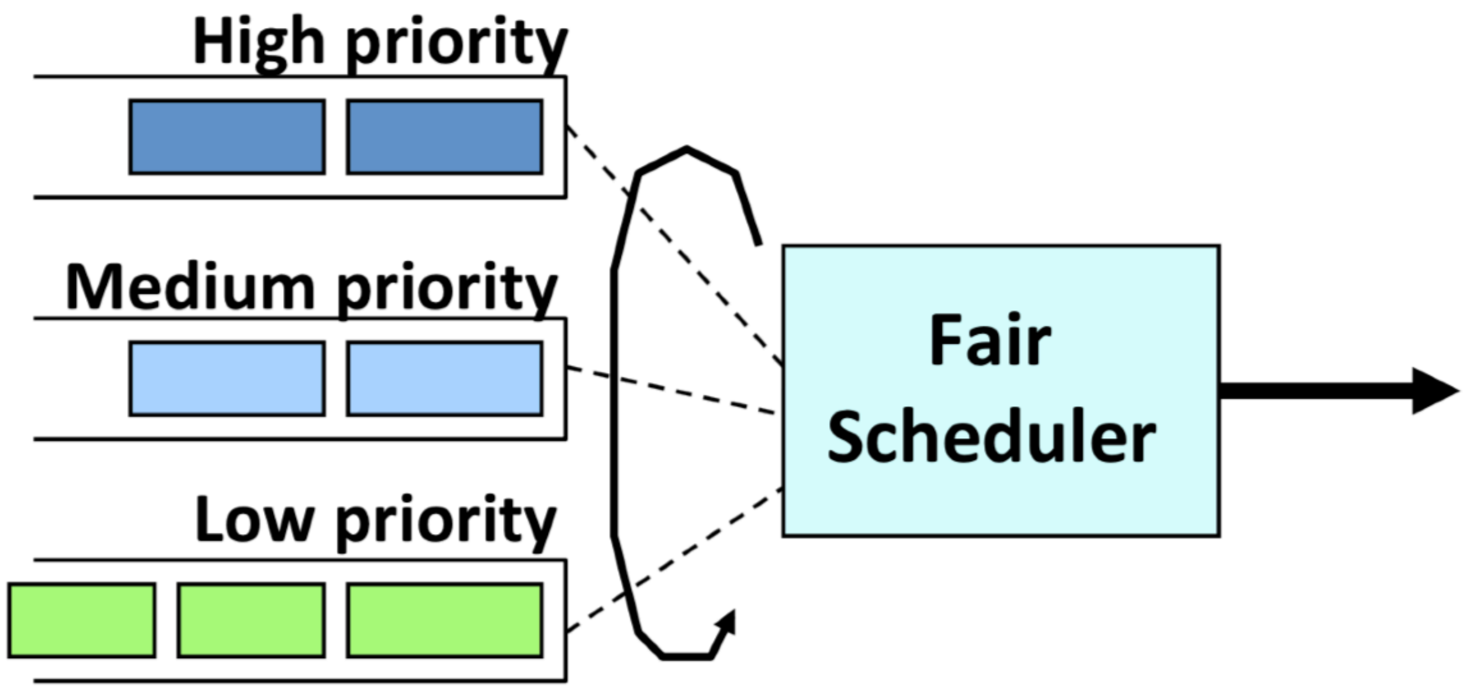
Example: Priority Scheduler

- Packets in the highest priority queue are always served before the packets in the lower priority queues



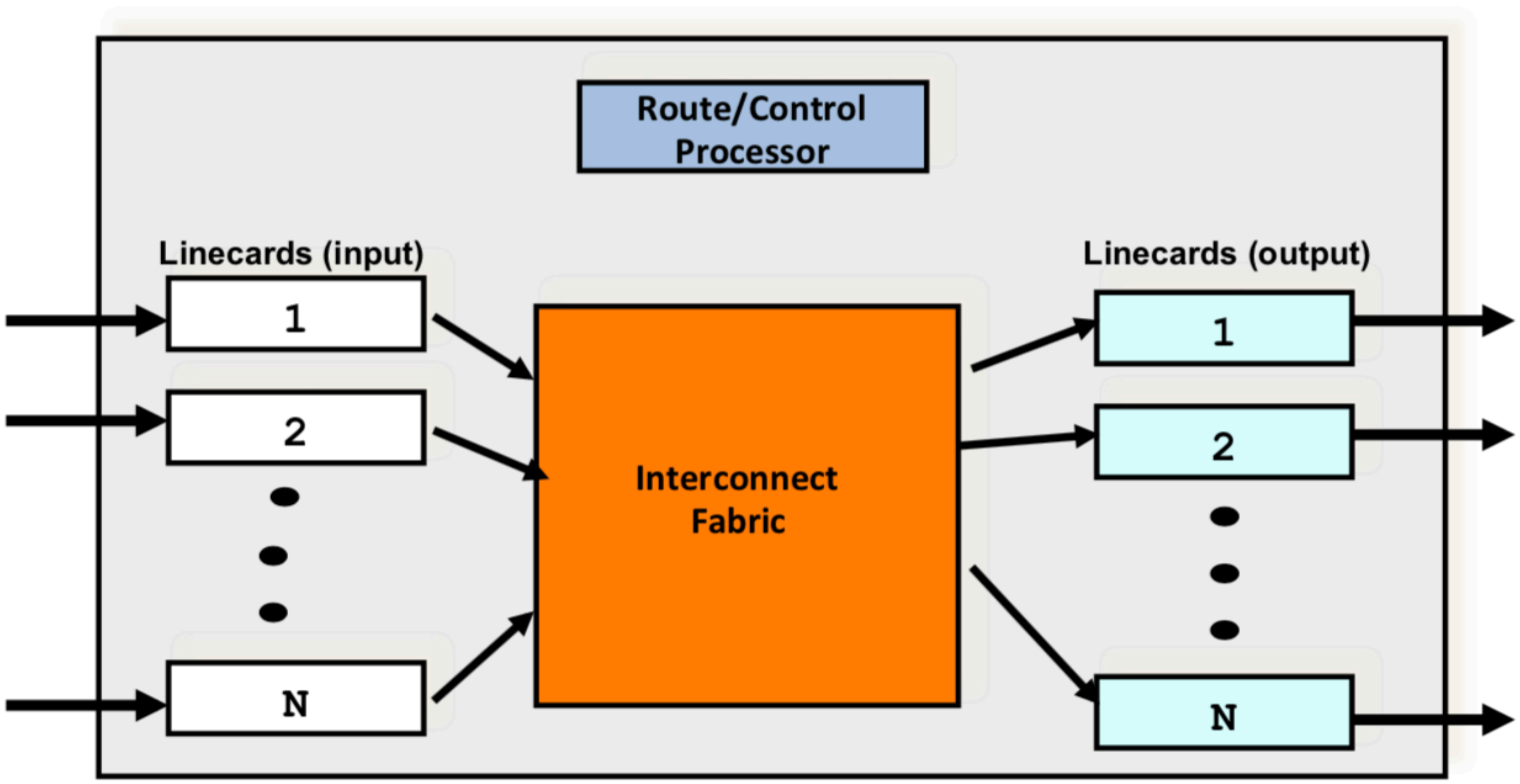
Example: Round Robin Scheduler

- Packets are served from each queue in turn

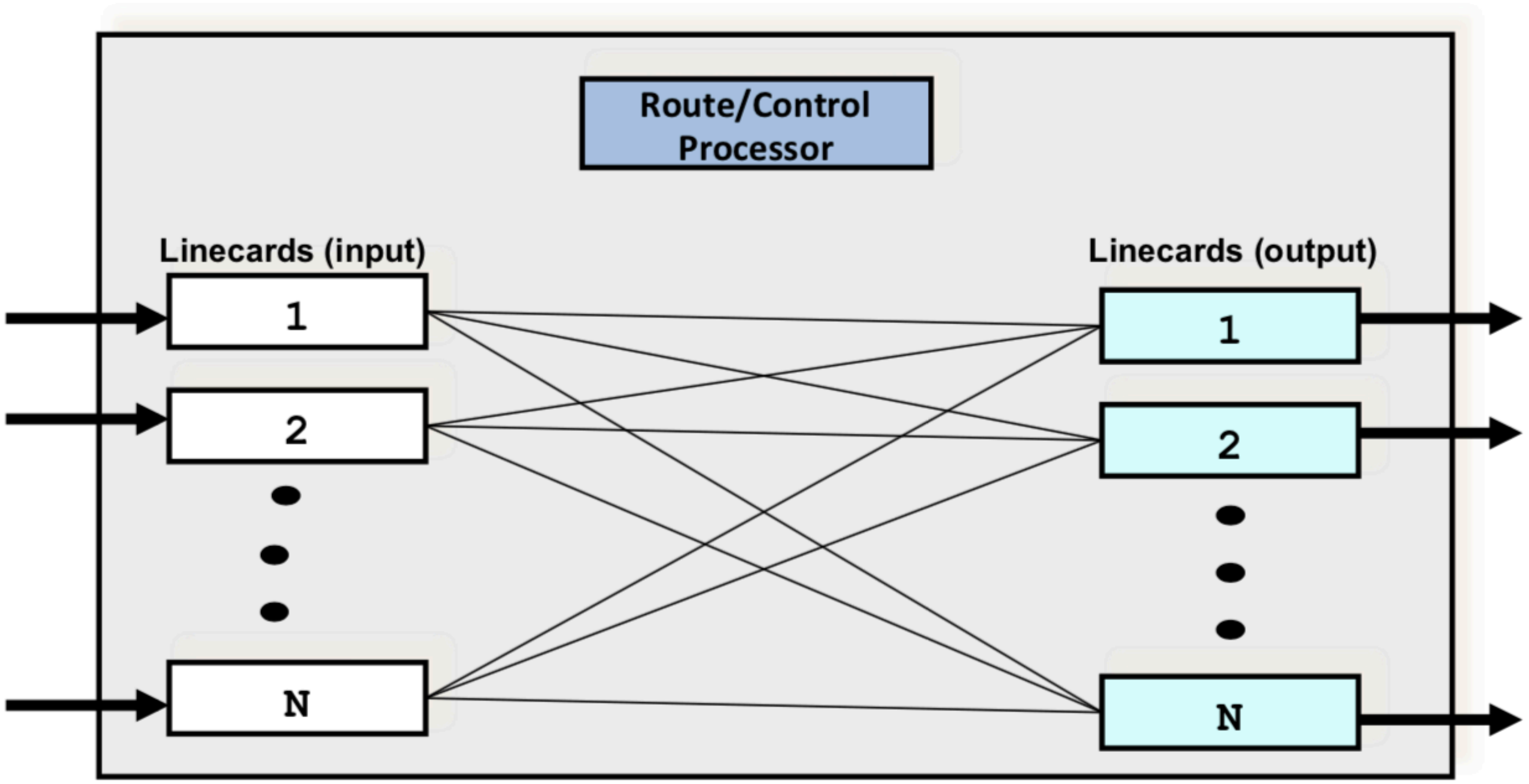


Connecting Input to Output: Switch Fabric

- Priority Scheduler: packets are served from each queue in turn



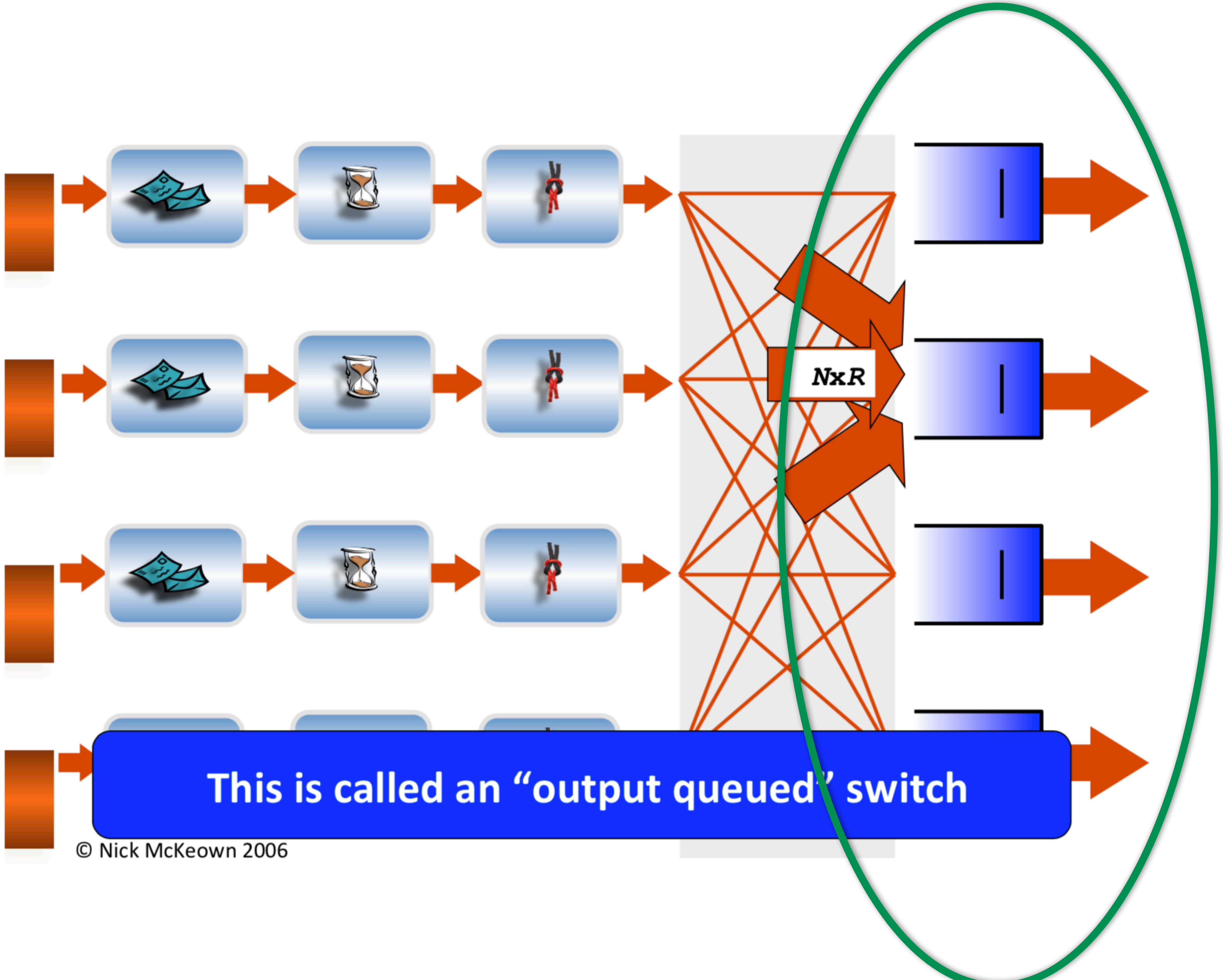
Today's Switch Fabrics: Mini Network!



What's Hard About the Switch Fabric?

Queueing!

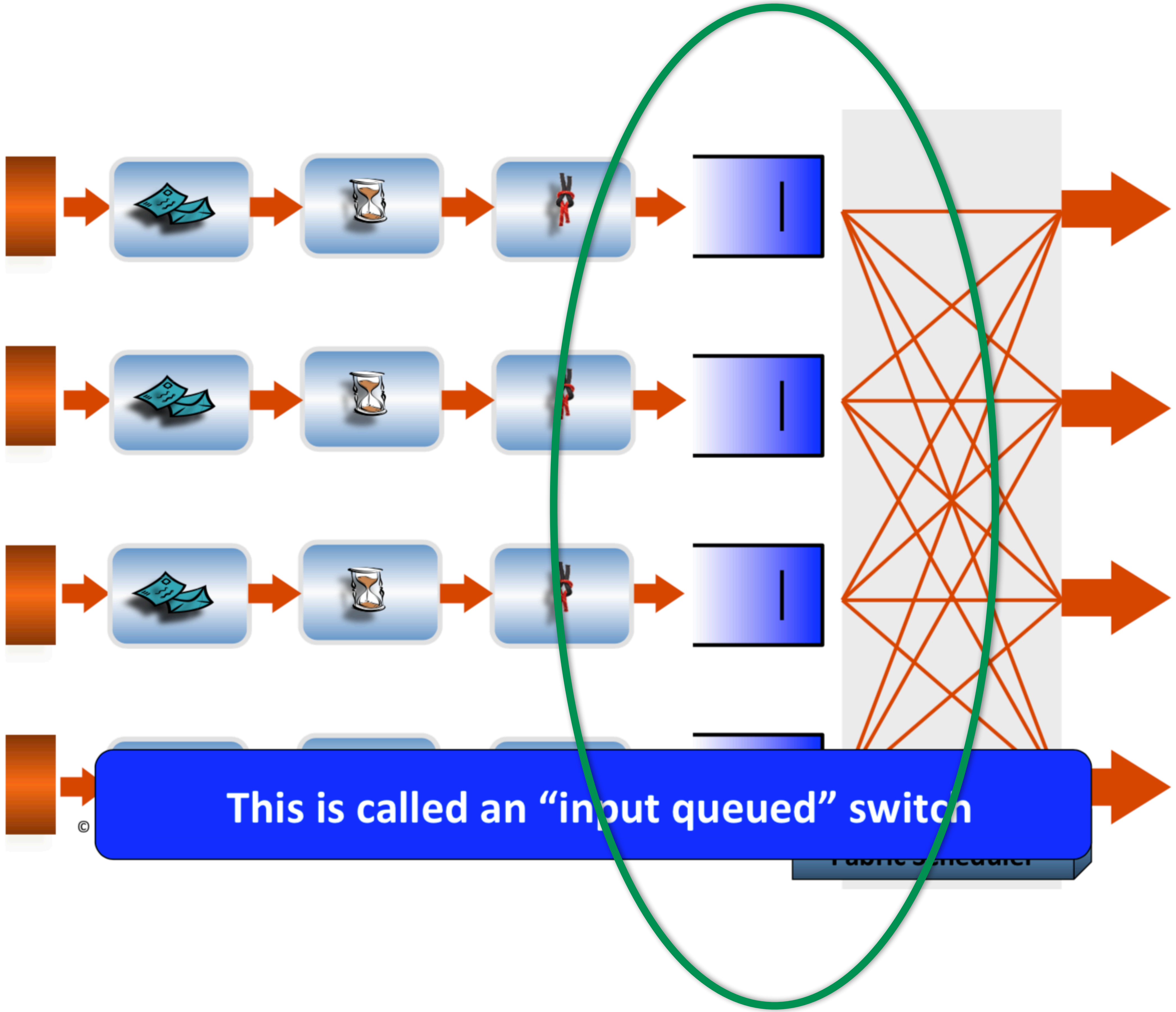
Third Generation Router: Switched Interconnects



This is called an "output queued" switch

© Nick McKeown 2006

Third Generation Router: Switched Interconnects



Reality is More Complicated

- Commercial high-speed routers use
  - Combination of input and output queueing
  - Complex multi-stage “topologies”
  - Distributed multi-stage schedulers (for scalability)

IP Routers Recap

- Core building block of Internet infrastructure
- Scalable Routing -> Longest Prefix Matching
- Need fast implementations for
    - Longest prefix matching
    - Switch fabric scheduling

What do we know so far [1] …

- **Network performance metrics**
  - Transmission delay, propagation delay, queueing delay, bandwidth
- **Sharing networks**
  - Circuit switching, packet switching, and associated tradeoffs
  - **Why** is Internet packet switched?
- **Architectural principles and design goals**
  - Layering principle, End-to-end principle, Fate sharing principle
  - Many important design goals from David Clark's paper
    - And many important missing goals
- **Addressing**
  - **Link layer MAC names,** and scalability challenges at the Internet
  - **Network layer IP addresses**: three requirements, aggregation, CIDR

What do we know so far [2] …

- **Link Layer**
  - Sharing a Broadcast medium, associated challenges, CSMA/CD
  - Link layer addressing: MAC names
  - **Why** Frames? **Why** Switched Ethernet?
  - The Spanning Tree Protocol (STP)

- **Network Layer**
  - **Why Network Layer? Why not just use STP across the Internet?**
  - **Routing Tables:** A collection of spanning trees, one per destination
  - **Generating Valid Routing tables (within a domain):**
    - Global view (Link-State Protocol), and limitations
    - Local view (Distance-vector Protocol)
  - **Generating Valid Routing tables (across domains):**
    - Border Gateway Protocol, Internet structure, routing policies

Network Layer

- THE functionality: **delivering the data**
- **THE protocol: Internet Protocol (IP)**
- **Achieves its functionality (delivering the data), using three ideas:**
    - **Addressing** (IP addressing)
    - **Routing** (using a variety of protocols)
    - **Packet header as an interface** (Encapsulating data into packets)