Network Virtualization

Lecture 7

Srinivas Narayana

http://www.cs.rutgers.edu/~sn624/553-S23

RUTGERS
UNIVERSITY | NEW BRUNSWICK

How to virtualize networking across a shared compute cluster?

A detour.

Software/hardware layering at hosts

Communication functions broken up and “stacked”

Each layer depends on the one below it.

Each layer supports the one above it.

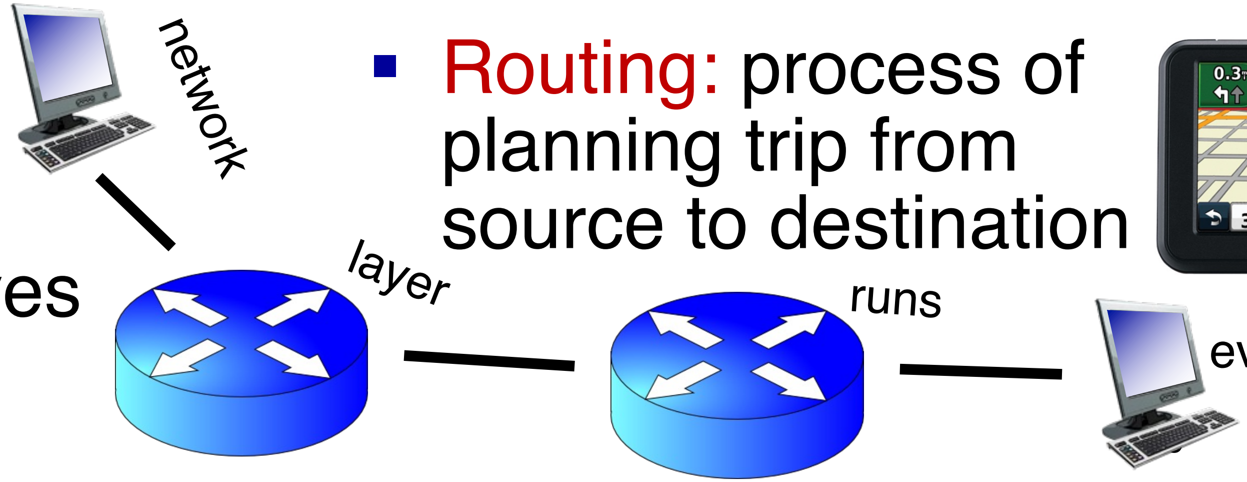
The interfaces between layers are well-defined and standardized.

Routing

Norwegian Sea
Iceland
Sweden
Finland
Norway
Russia
Denmark
Ireland
Poland
Belarus
Ukraine
Kazakhstan
Austria
Romania
Spain
Portugal
Turkey
Turkmenistan
China
Sea of Japan
Japan
Sea of Okhotsk
Tunisia
Syria
Iraq
Iran
Afghanistan
Pakistan
Nepal
India
Morocco
Algeria
Libya
Egypt
Saudi Arabia
Oman
Western Sahara
Mauritania
Mali
Niger
Chad
Sudan
Yemen
Gulf of Aden
Arabian Sea
Bay of Bengal
Myanmar (Burma)
Philippines
Burkina Faso
Guinea
Nigeria
Ghana
South Sudan
Ethiopia
Laccadive Sea
Gulf of Guinea
Somalia
Kenya
Gabon
DR Congo
Tanzania
Indonesia
Banda Sea
Arafura Sea
Papua New Guinea

Two key network-layer functions

- Forwarding: move packets from router's input to appropriate router output
- Routing: determine route taken by packets from source to destination
  - routing algorithms
- The network layer solves the routing problem.



Analogy: taking a road trip

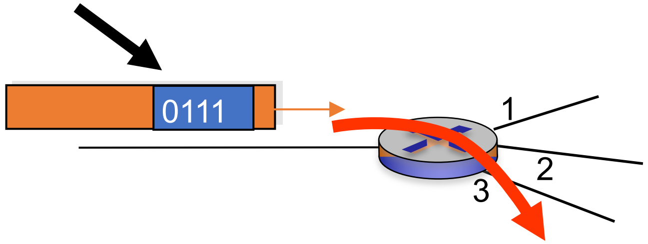
- Forwarding: process of getting through single exit
- Routing: process of planning trip from source to destination

Control/Data Planes

Data plane = Forwarding

- local, per-router function
- determines how datagram arriving on router input port is forwarded to router output port

values in arriving packet header



Control plane = Routing

- network-wide logic
- determines how datagram is routed along end-to-end path from source to destination endpoint
- two control-plane approaches:
  - Distributed routing algorithm running on each router
  - Centralized routing algorithm running on a (logically) centralized machine

Distributed Routing

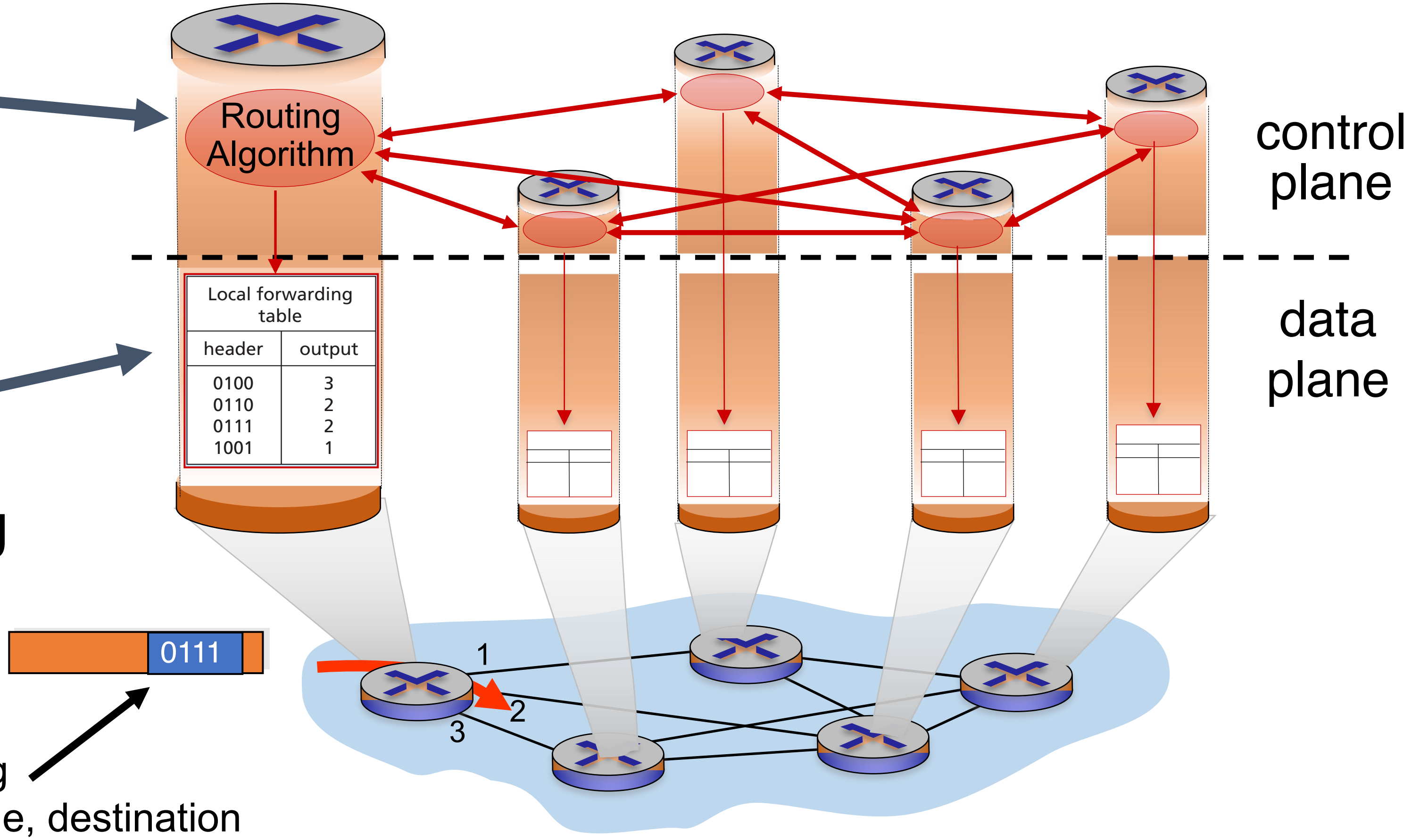
Distributed routing

Control plane

Traditional routing protocols: per route-change processing (~ a few tens of seconds)

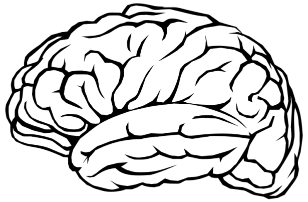
Data plane

per-packet processing (~ tens of nanoseconds)

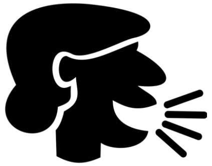


values in arriving packet header, i.e, destination IP address

What is the goal of routing?



- Efficiency: find "good" paths
  - Low latency, low cost, high bandwidth, etc.
  - Often translates to shortest path on a suitably modeled graph!
  - Edges: link metrics. Nodes: routers.

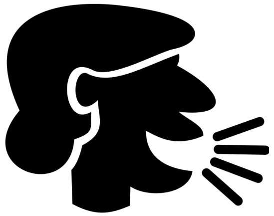


- Internet rationale: distribute intelligence: avoid failures; scale

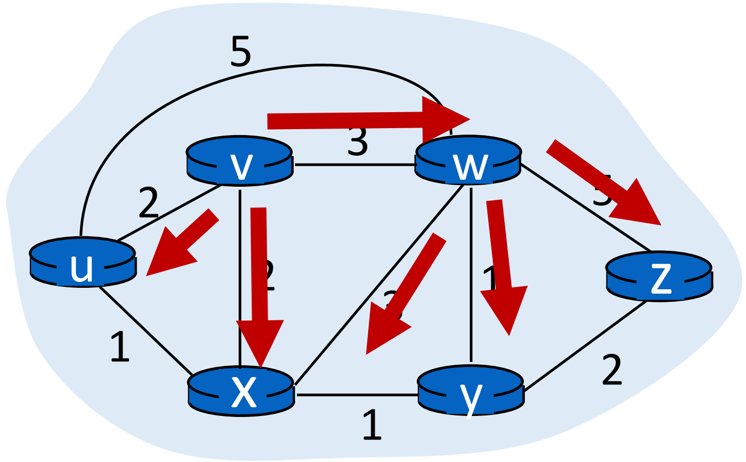
- Two questions: (1) what messages? (2) what algorithm?
  - Link state and distance vector protocols
  - Applicable when

Example 1: link state routing protocol

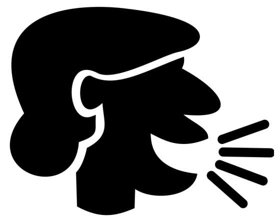
Q1: Information exchange



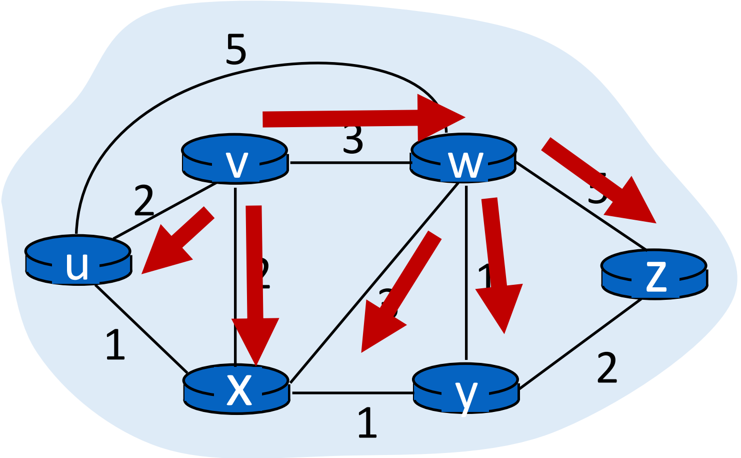
- Link state flooding: the process by which neighborhood information of each network router is transmitted to all other routers
- Each router sends a link state advertisement (LSA) to each of its neighbors
- LSA contains the router ID, the IP prefix owned by the router, the router's neighbors, and link cost to those neighbors
- Upon receiving an LSA, a router forwards it to each of its neighbors: flooding



Q1: Information exchange



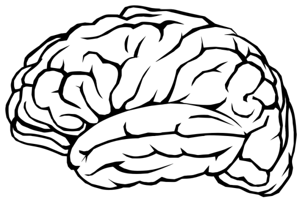
- Eventually, the entire network receives LSAs originated by each router
- LSAs put into a link state database
- LSAs occur periodically and whenever the graph changes
  - Example: if a link fails, or new router added
- The routing algorithm running at each router can use the entire network's graph to compute least cost paths



Destination IP prefix	Output Port on Router

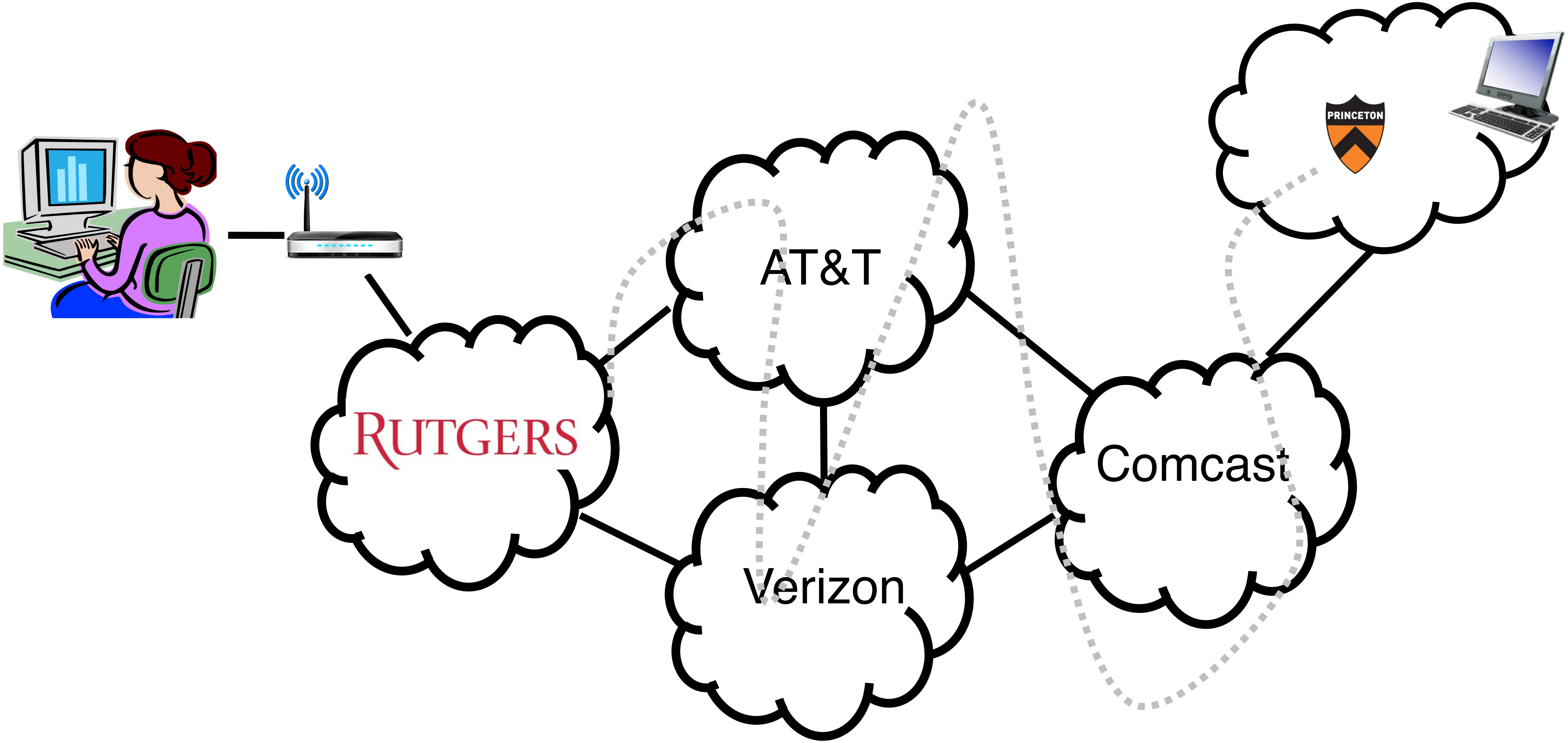
Forwarding table

Q2: Algorithm: Dijkstra's shortest paths



Example 2: Internet Routing

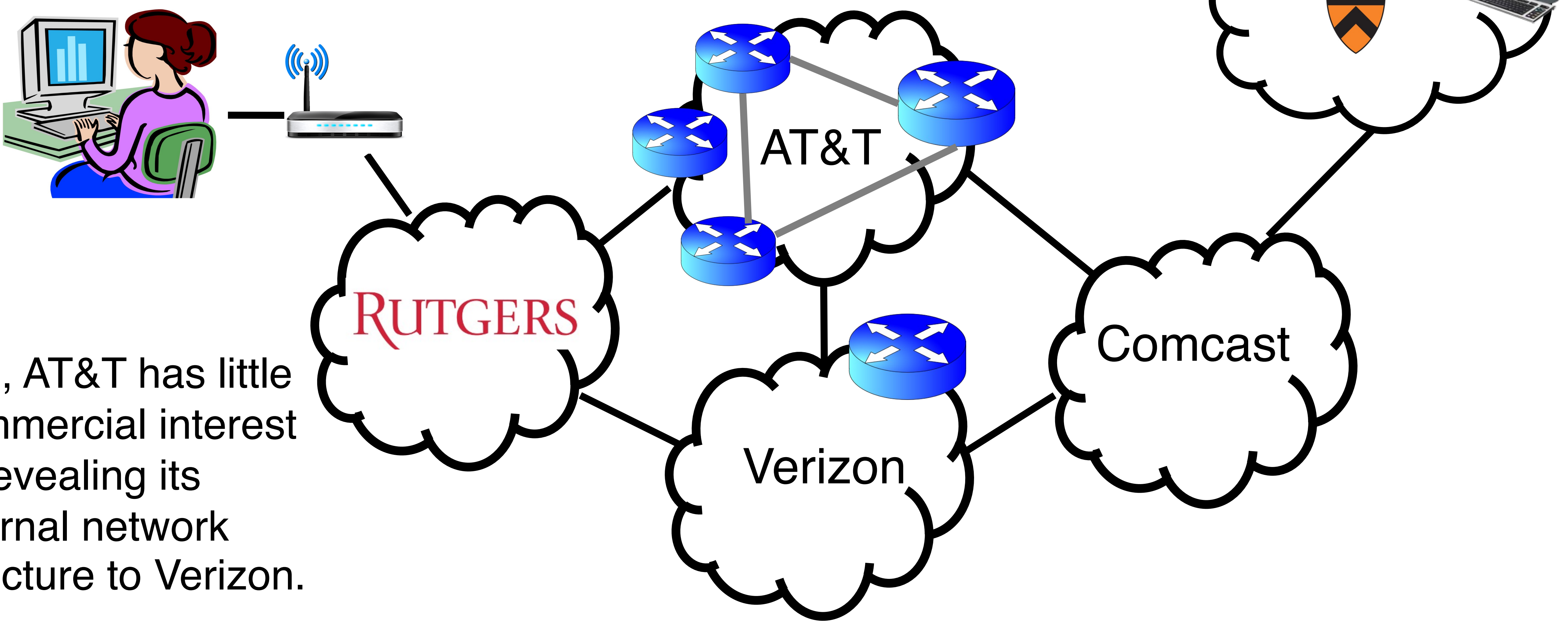
The Internet is a large federated network



The Internet is a large federated network

Several autonomously run organizations (AS'es): No one "boss"

Organizations cooperate, but also compete

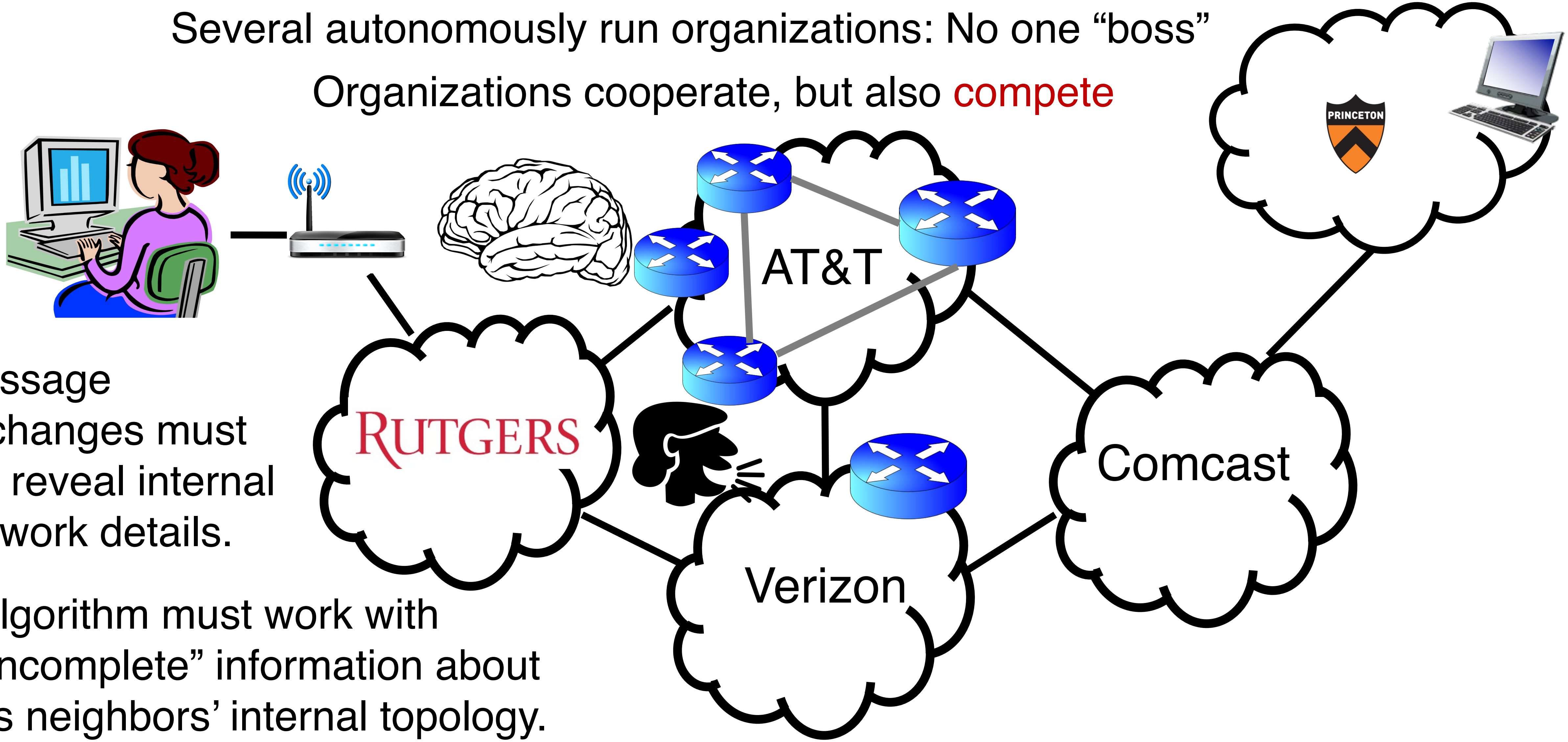


e.g., AT&T has little commercial interest in revealing its internal network structure to Verizon.

The Internet is a large federated network

Several autonomously run organizations: No one "boss"

Organizations cooperate, but also compete



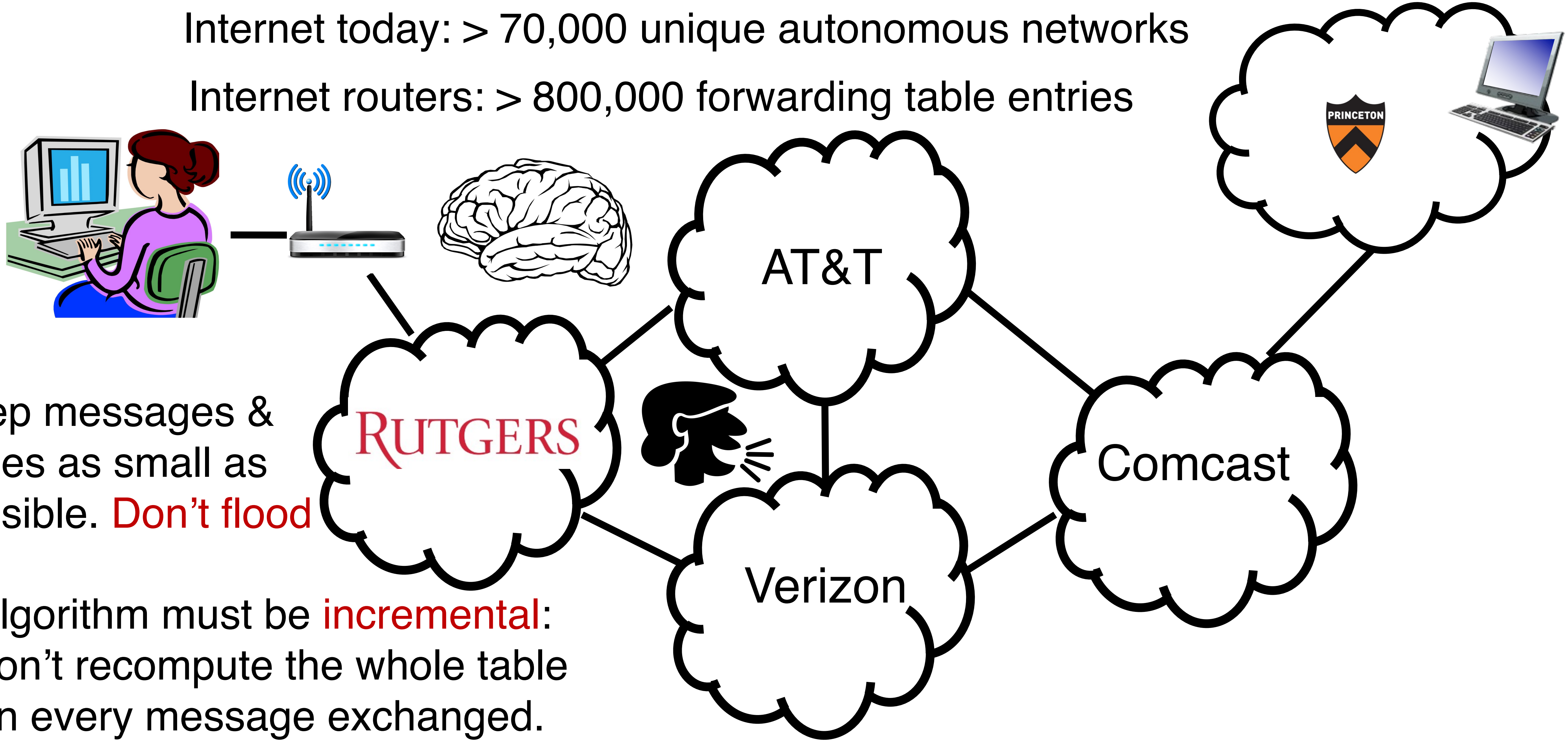
Message exchanges must not reveal internal network details.

Algorithm must work with "incomplete" information about its neighbors' internal topology.

The Internet is a large federated network

Internet today: > 70,000 unique autonomous networks

Internet routers: > 800,000 forwarding table entries

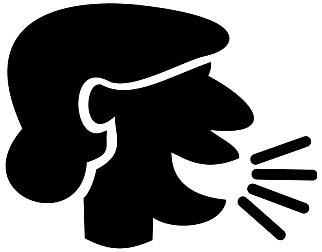


Keep messages & tables as small as possible. Don't flood

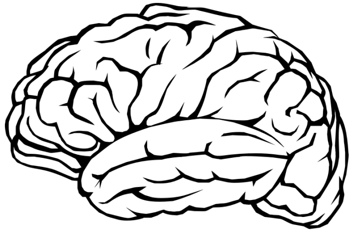
Algorithm must be incremental: don't recompute the whole table on every message exchanged.

Inter-domain Routing

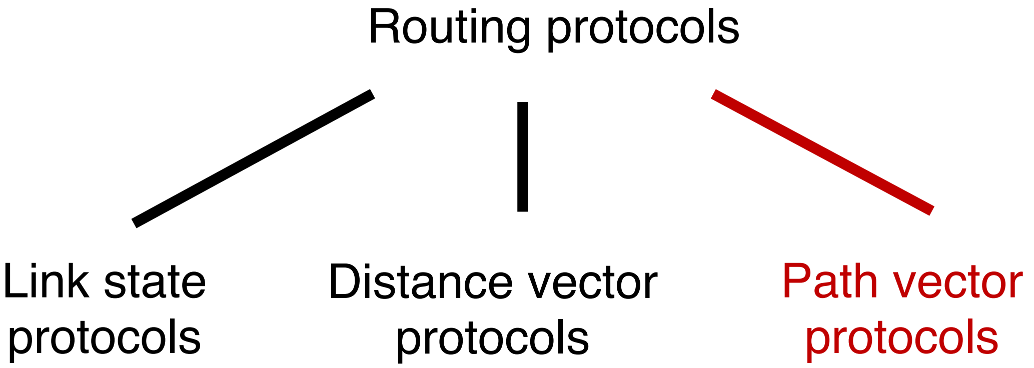
- The Internet uses Border Gateway Protocol (BGP)
- All AS'es speak BGP. It is the glue that holds the Internet together
- BGP is a path vector protocol



Messages?

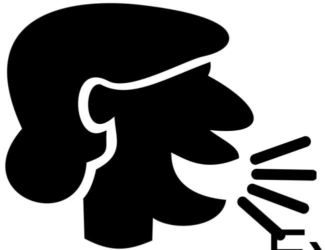


Algorithm?



Applicable within a single AS

(1) BGP Messages

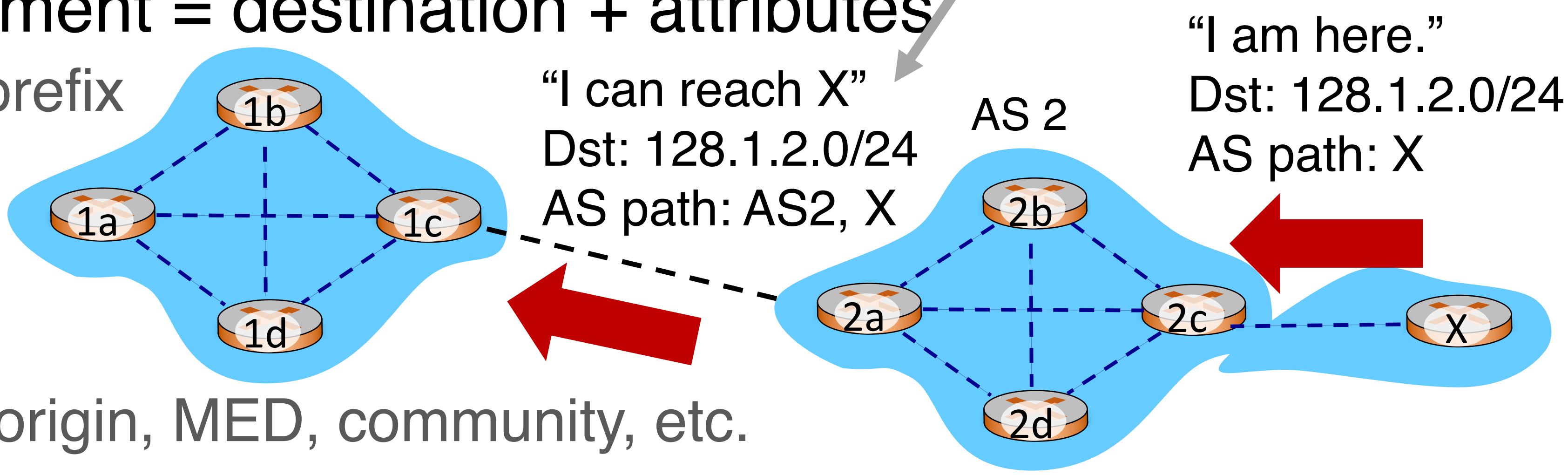


Loop detection is easy (no “count to infinity”)

Exchange paths: path vector

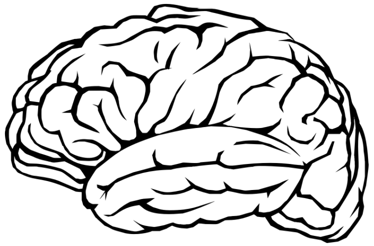
No link metrics, distances!

- Routing Announcements or Advertisements
  - “I am here” or “I can reach here”
  - Occur over a TCP connection (BGP session) between routers
- Route announcement = destination + attributes
  - Destination: IP prefix
- Route Attributes:
  - AS-level path
  - Next hop
  - Several others: origin, MED, community, etc.



- An AS promises to use advertised path to reach destination
- Only route changes are advertised after BGP session established

(2) BGP algorithm



- A BGP router does *not* consider every routing advertisement it receives by default to make routing decisions!
  - An import policy determines whether a route is even considered a candidate
- Once imported, the router performs route selection
- A BGP router does *not* propagate its chosen path to a destination to all other AS'es by default!
  - An export policy determines whether a (chosen) path can be advertised to other AS'es and routers

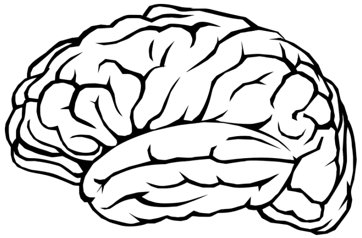
Programmed by network operator

Business policy considerations drive BGP.
Not necessarily efficient outcomes!

Policy arises from business relationships

- Customer-provider relationships:
  - E.g., Rutgers is a customer of AT&T
- Peer-peer relationships:
  - E.g., Verizon is a peer of AT&T
- Business relationships depend on where connectivity occurs
  - “Where”, also called a “point of presence” (PoP)
  - e.g., customers at one PoP but peers at another
  - Internet-eXchange Points (IXPs) are large PoPs where ISPs come together to connect with each other (often for free)

Q2. BGP Route Selection



- When a router imports more than one route to a destination IP prefix, it selects route based on:
  1. local preference value attribute (import policy decision -- set by network admin)
  2. shortest AS-PATH
  3. closest NEXT-HOP router
  4. Several additional criteria: You can read up on the full, complex, list of criteria, e.g., at https://www.cisco.com/c/en/us/support/docs/ip/border-gateway-protocol-bgp/13753-25.html

Problems with BGP

Approaches to bring flexibility:
Flexible control logic for path selection (Google, Facebook)
Detour/overlay routing (Akamai)

- Not designed for efficiency

  1. local preference value attribute (import policy decision -- set by network admin)
  2. shortest AS-PATH
  3. closest NEXT-HOP router

  Nothing to do with path length, delay, or available capacity.

- Only a single path per destination
- Slow to converge after a change
- Vulnerable to bugs & malice

Example 3: Layer-2 switching

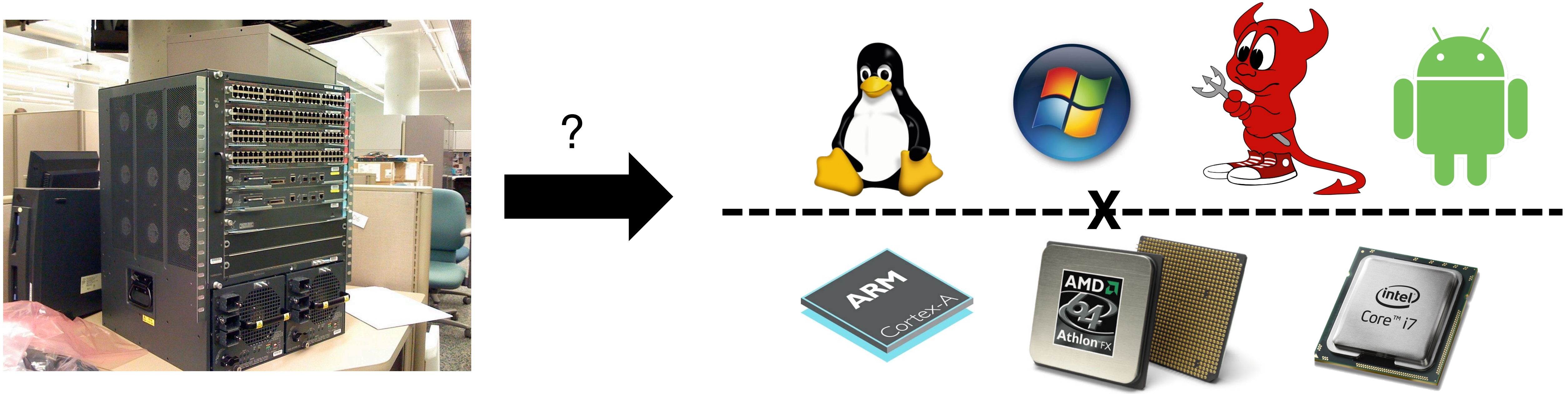
Layer-2 switching

- Switch: move packets based on link layer addresses
- Provide an illusion of a single link connecting many endpoints
  - Without every endpoint necessarily hearing every other endpoint
- Learning switch: zero configuration or control plane.
  - All endpoints in the same IP network
  - Flood packets when dest MAC address unknown
  - Use source MAC of incoming packets and associate with the incoming switch port: use later for forwarding
- Works even if endpoints move, so long as they are in the same IP prefix

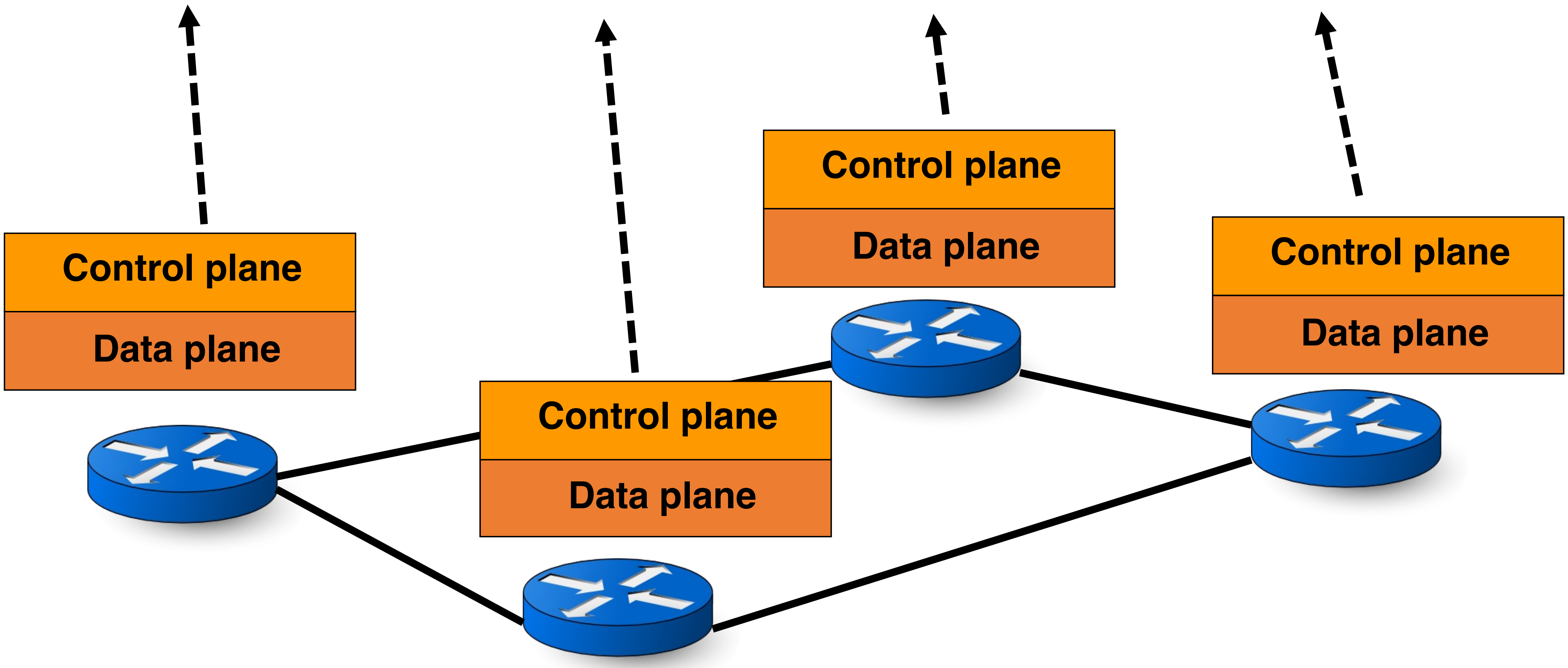
Centralized Routing

Problems with distributed control planes

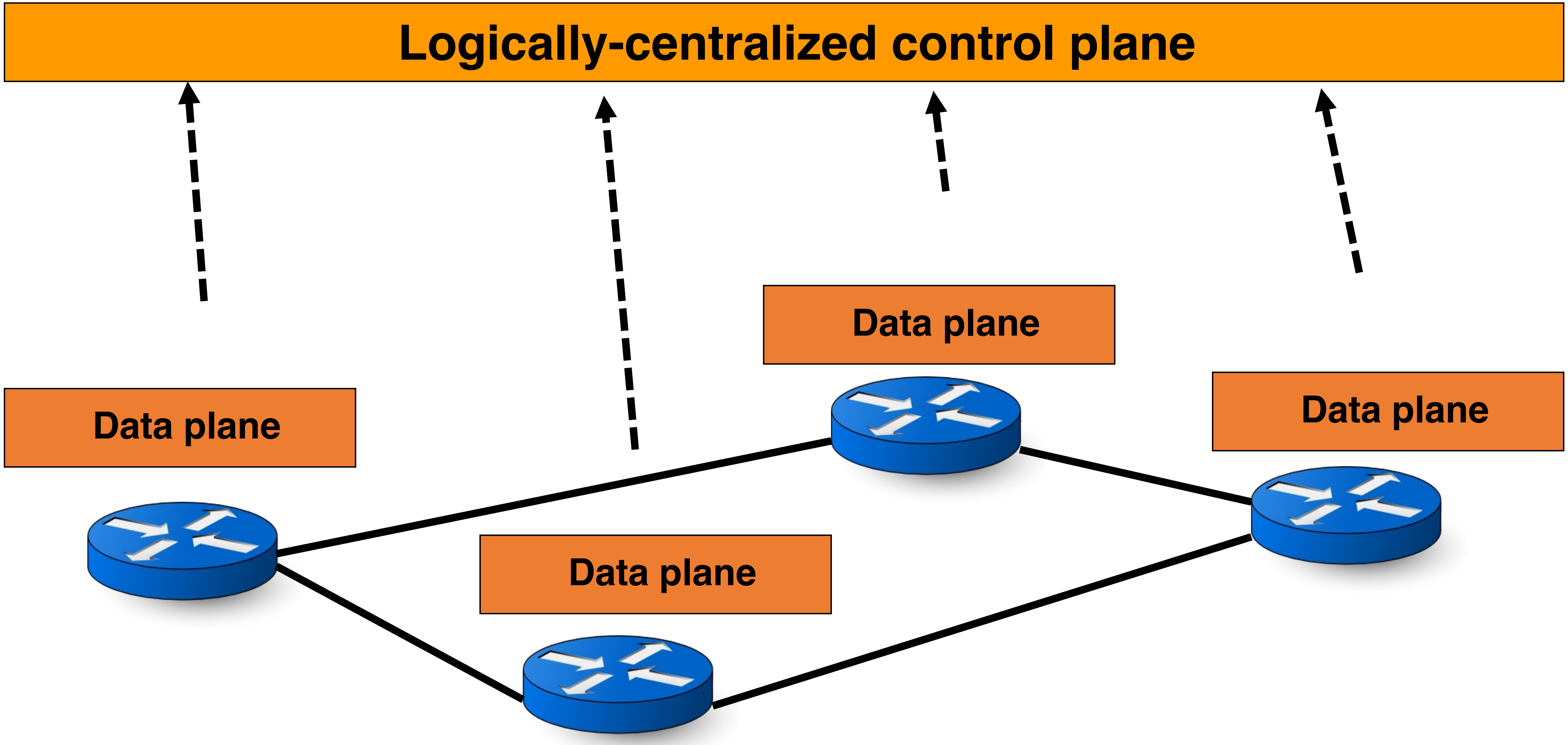
- Management decisions tied to distributed protocols
  - Ex: Set OSPF link weights to force traffic through desired path
  - Ex: Non-deterministic network state after a link failure
- Data and control plane controlled by vendors: proprietary interfaces



Traditional IP network



Software-defined network

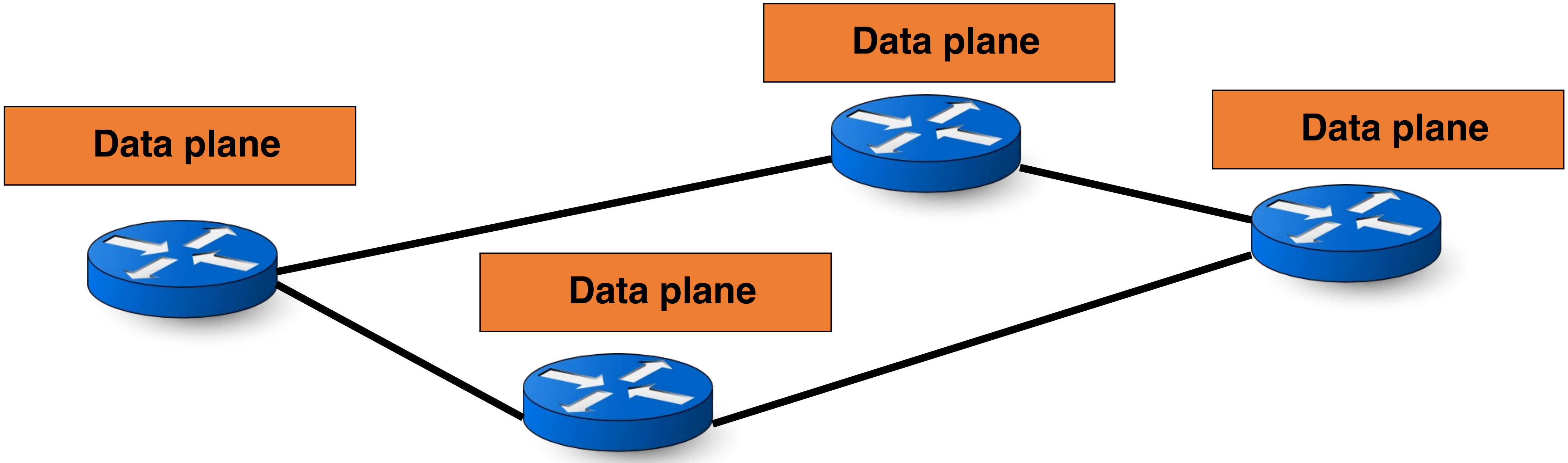


Software-Defined Networking

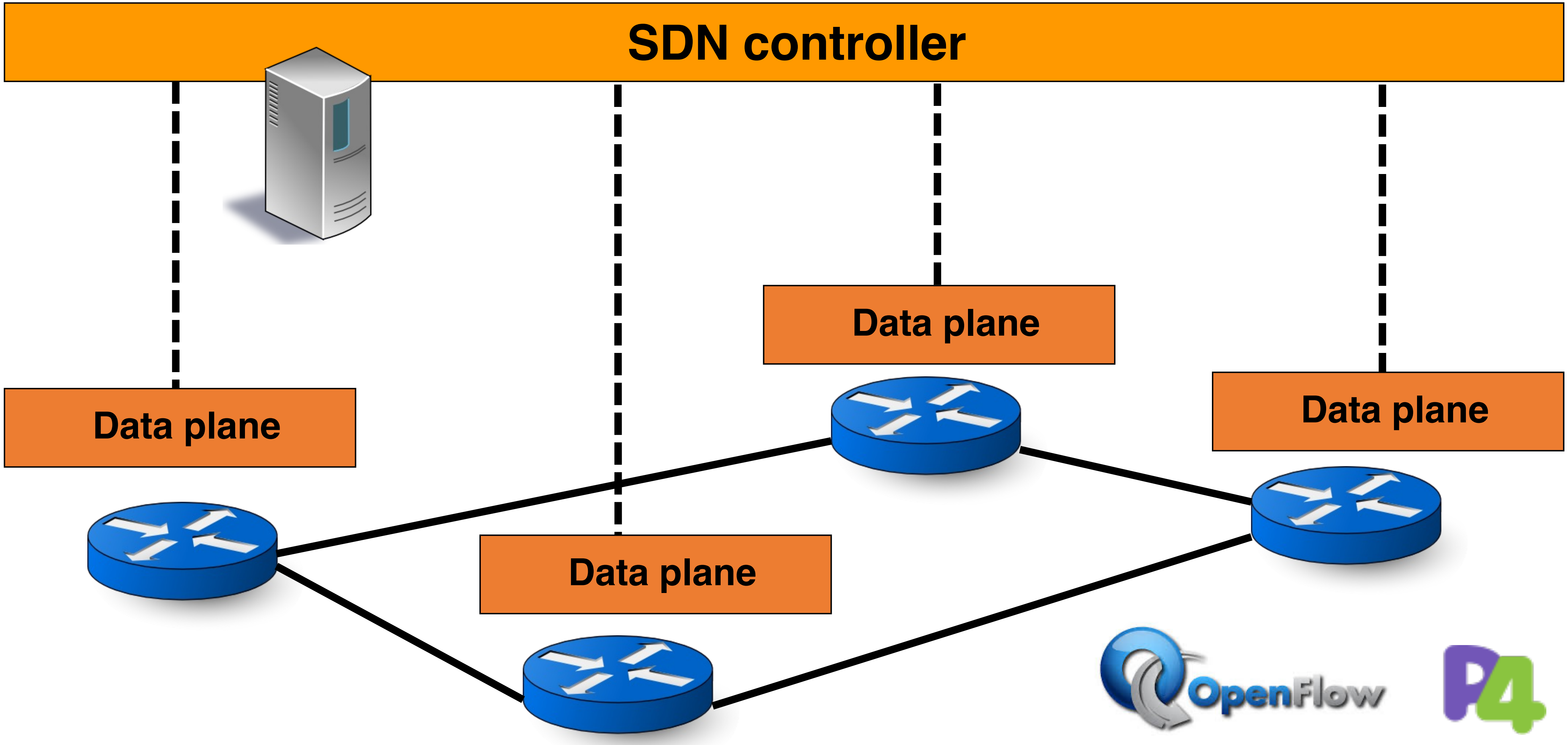
SDN (1/2): Centralized control plane

SDN controller

Control planes lifted from switches
… into a logically centralized *controller*
… running in a compute cluster

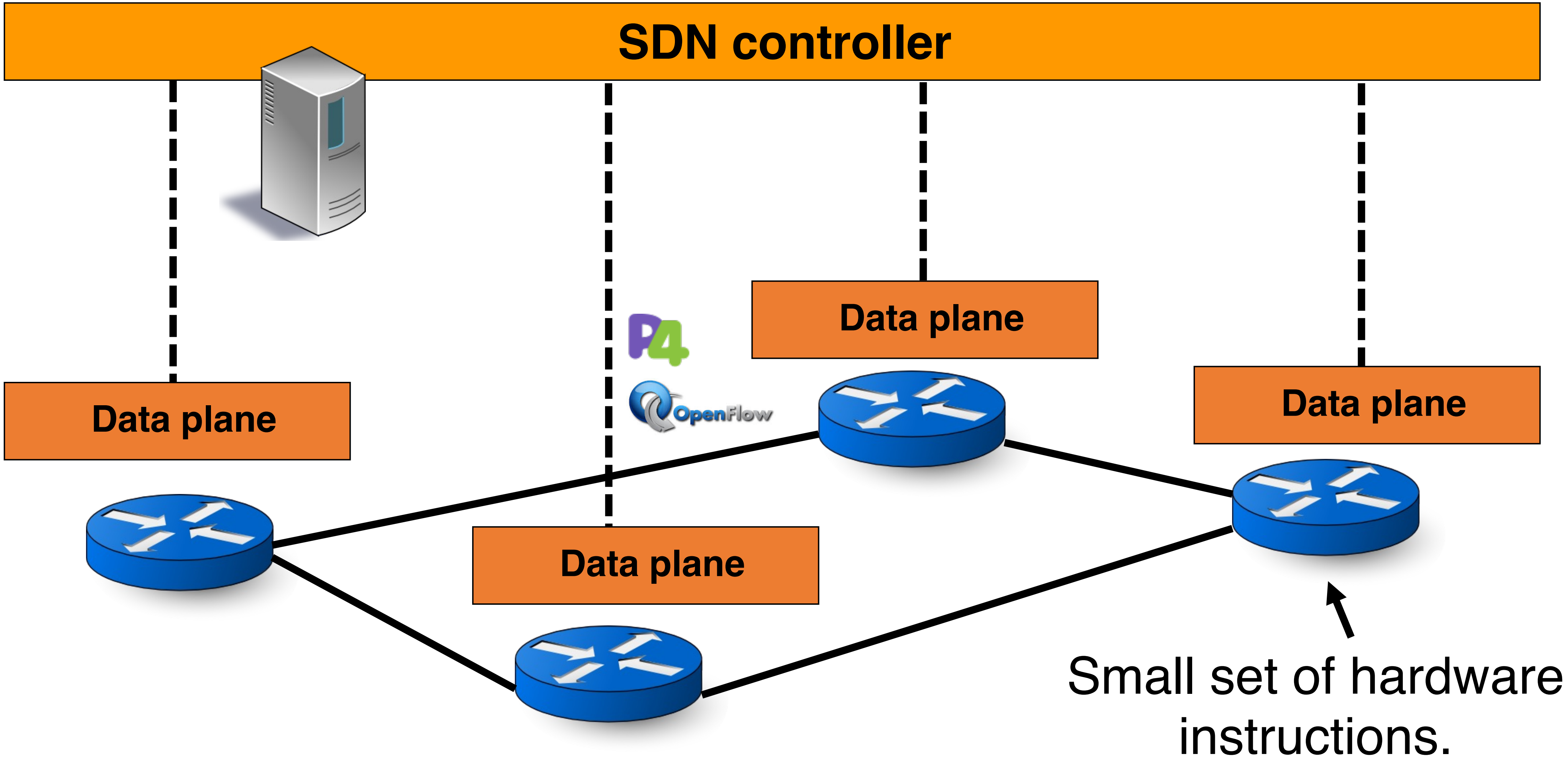


SDN (2/2): Open interface to data plane



Some immediate consequences

(1) Simpler switches



Data plane primitive: Match-action rules

- Match arbitrary bits in the packet header

`Match: 1000x01xx01001x`

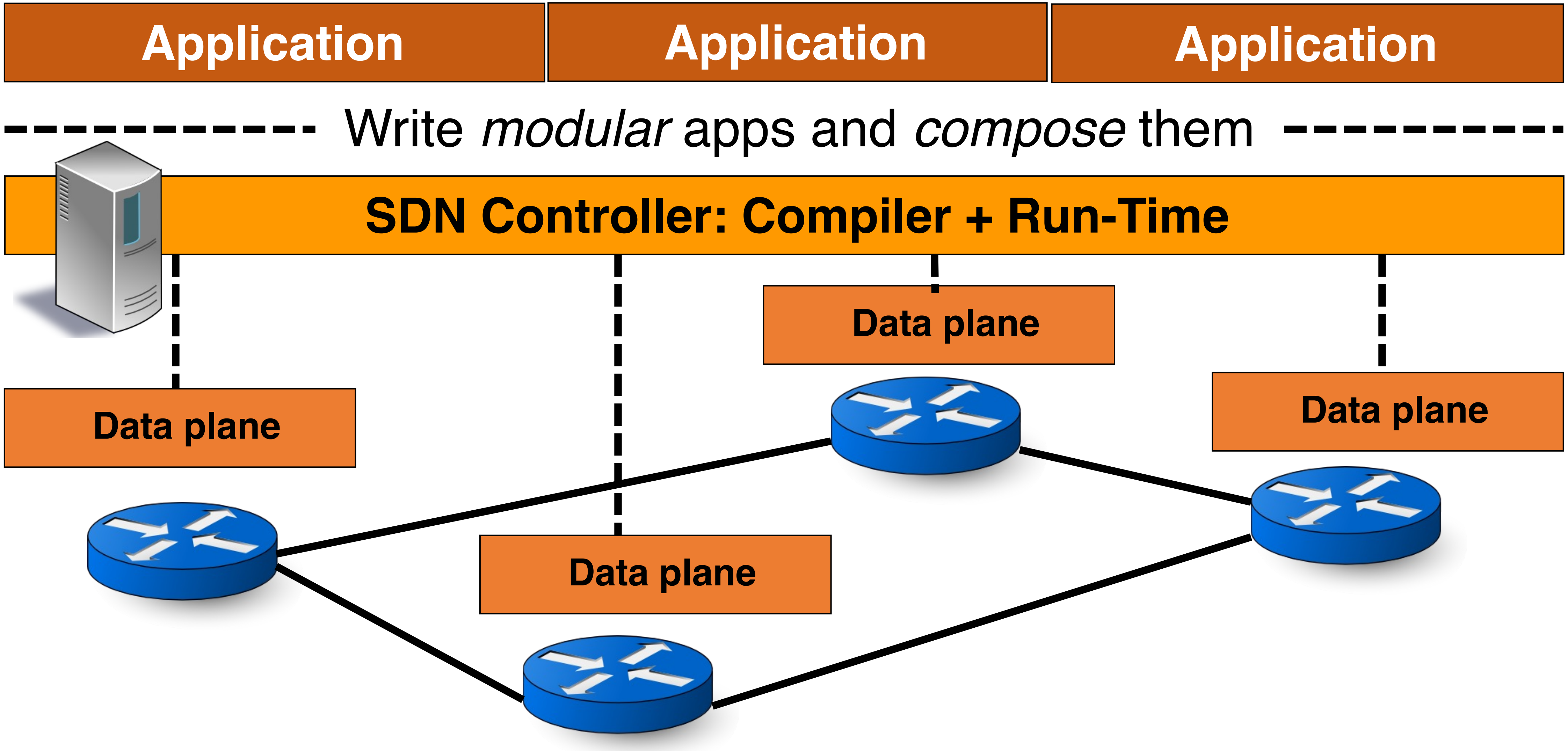
  - Match on any header, or new header
  - Match exact, a subset (ternary), or over a range
  - Allows any flow granularity
- Actions
  - Forward to port(s), drop, send to controller, count,
  - Overwrite header with mask, push or pop, …
  - Forward at specific bit-rate

`Action: fwd(port 2)`

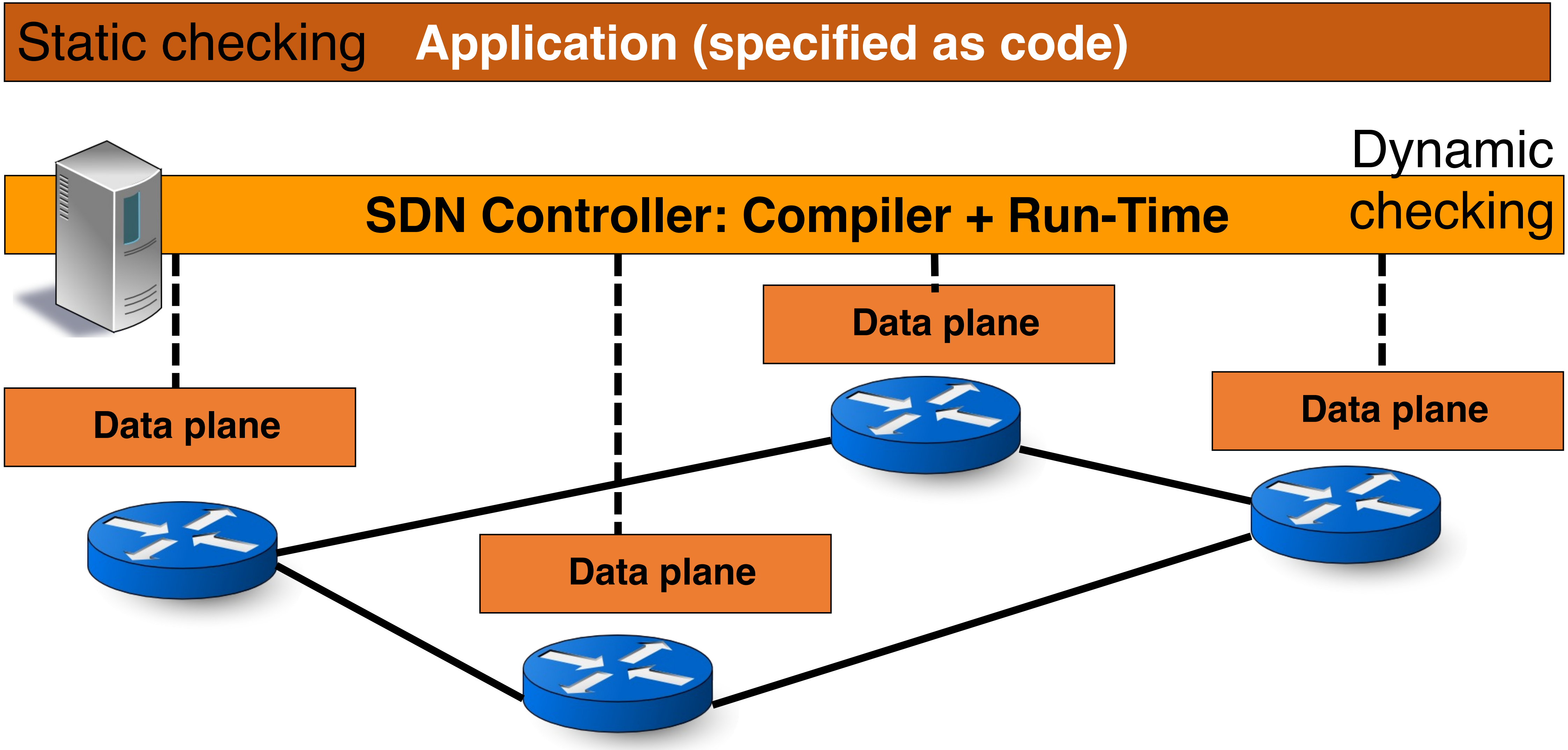
- Prioritized list of rules

`Priority: 65500`

(2) Network programming abstractions



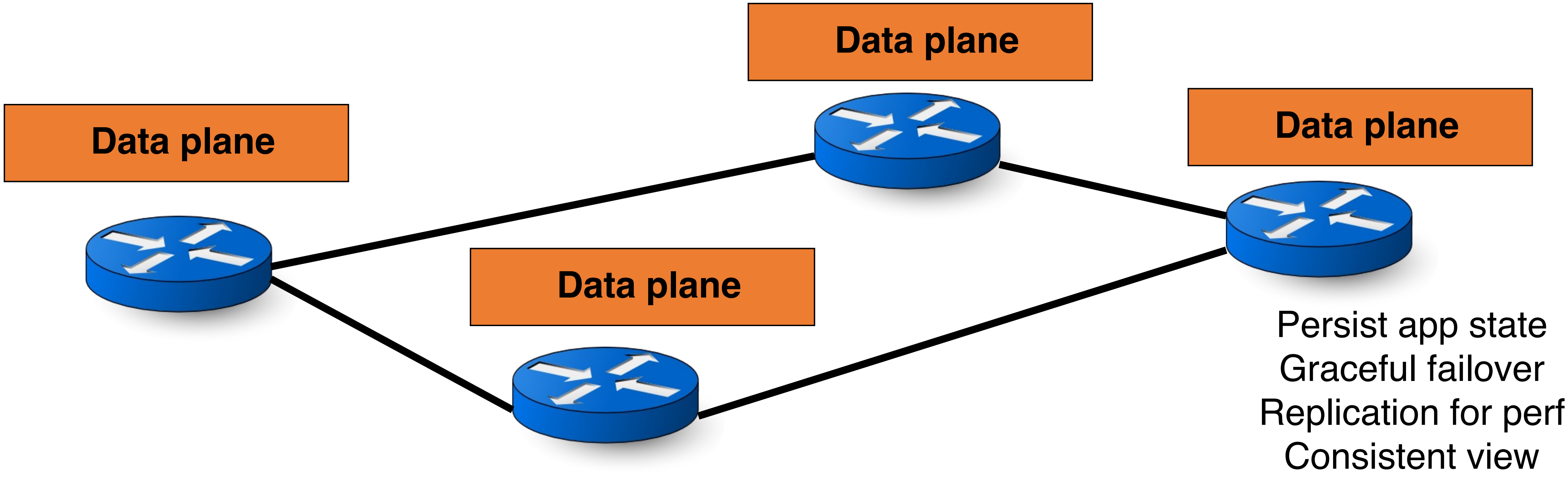
(3) Formal verification of Network Policy



(4) Unified network operating system

Application	Application	Application
Network Operating System		

Separate distributed system concerns from expressing intent

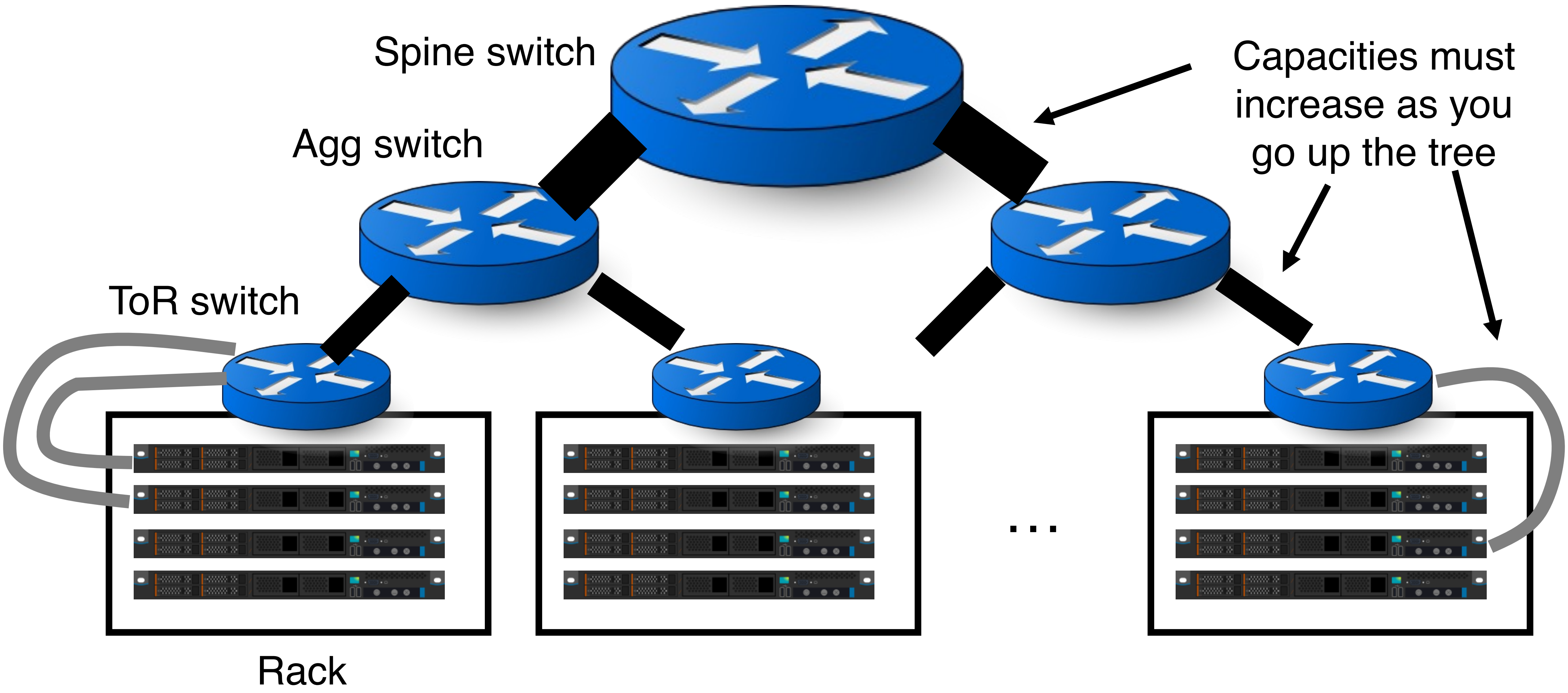


New technical challenges of SDN

- Availability: surviving failures of the controller
- Controller scalability: many routers, many events
  - Response time: Delays between controller and routers
- Consistency: Ensuring multiple controllers behave consistently
- Designing flexible router mechanisms
- Compilation: translating intent to mechanisms
- Verification: ensuring controller policy is faithfully implemented
- Security: entire network owned if the controller is exploited
- Interoperability: legacy routers; neighboring domains; …

Virtualizing Networking in a Shared Cluster

Typical network structure: Fat Trees

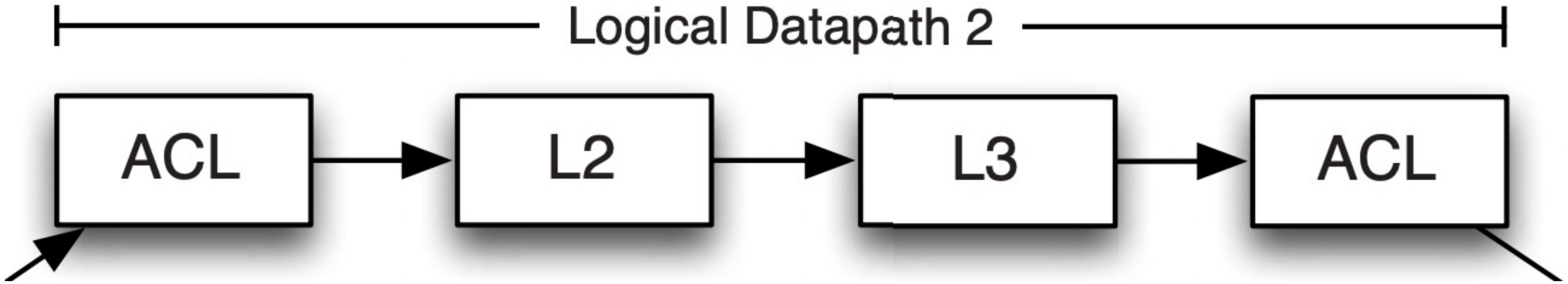


Networking in a multi-tenant cloud

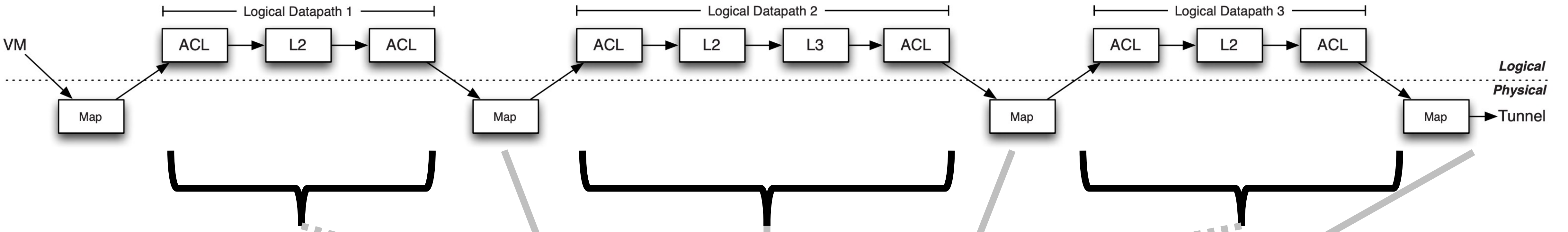
- Problems: Many tenants, time-varying demands.
  - Want homogeneity across data center on use of compute capacity
  - Where to provision VMs?
  - How to migrate VMs or scale the number of VMs?
- Idea (1): VMs get their own network addresses
  - network address virtualization
- Idea (2): tenants should be able to use custom topologies
  - Facilitate migration, consistent view for monitoring and maintenance tools, etc.
  - Needed "in practice" rather than "in principle"
  - But, important to do!

How cloud network looks to a tenant

- Control abstraction: pipeline of lookup tables
- Packet abstraction: send to IP addresses of your own
  - Processed through switch/router topology
  - Data plane behavior defined through control plane configuration
- Design of NVP (nicira virtualization platform):
  - Push all interesting data plane behaviors to the edge (hypervisor, OVS)
  - The core of the network (switches/routers) just moves data using tunnel headers



Topology and Address Virtualization



Topology virtualization

Address virtualization
(separation of tenant and provider addresses through tunneling)

Effect: Move packets from source vNIC to dest vNIC

Performance: Caching

Controller design

- Declarative design: language to specify tuples of rules/relations
  - No need to implement a state machine to transition rule sets
  - Use a compiler to emit correct, up to date logical datapaths (tuples)
- Shared-nothing parallelism to scale
  - Different logical datapaths easily distributed
  - "Template" rules output from logical datapaths may be independently specialized to specific hypervisors and VMs
- Controller availability maintained using standard leader election mechanisms
- Control and data paths fail independently
  - Existing OVS hypervisor rules can process packets even if controller fails
  - Fast failover through precomputed failover installed in the data path

Making old software use new networks usually means making new networks behave like old ones.