# Central Control Plane

Jennifer Rexford

Fall 2010 (TTh 1:30-2:50 in COS 302)

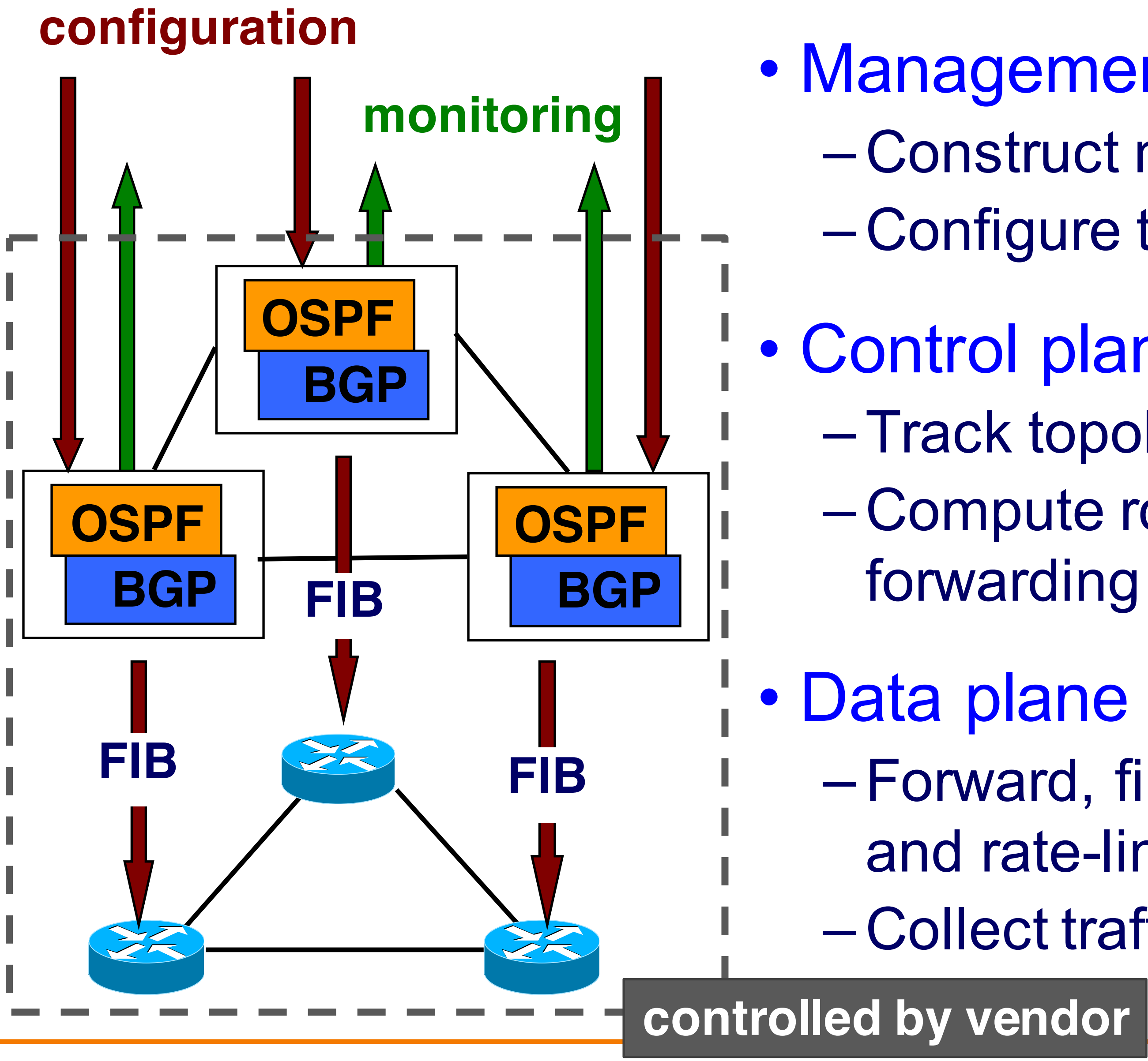
COS 561: Advanced Computer Networks

http://www.cs.princeton.edu/courses/archive/fall10/cos561/

# Outline

- Motivation for refactoring the "planes"
- Central control plane
  - Routing Control Platform (RCP)
  - 4D architecture
  - OpenFlow/NOX
- Technical challenges
  - Scalability, reliability, failover time, security, consistency, backwards compatibility
- Discussion of the papers

# Today's IP Routers




- Management plane
  - Construct network-wide view
  - Configure the routers
- Control plane
  - Track topology changes
  - Compute routes and install forwarding tables
- Data plane
  - Forward, filter, buffer, mark, and rate-limit□packets
  - Collect traffic statistics

# (Re)Move the Control Plane?

- Faster pace of innovation
  - Remove dependence on vendors and the IETF
- Simpler management systems
  - No need to “invert” control-plane operations
- Easier interoperability between vendors
  - Compatibility necessary only in “wire” protocols
- Simpler, cheaper routers
  - Little or no software on the routers

# Can We Remove the Control Plane?

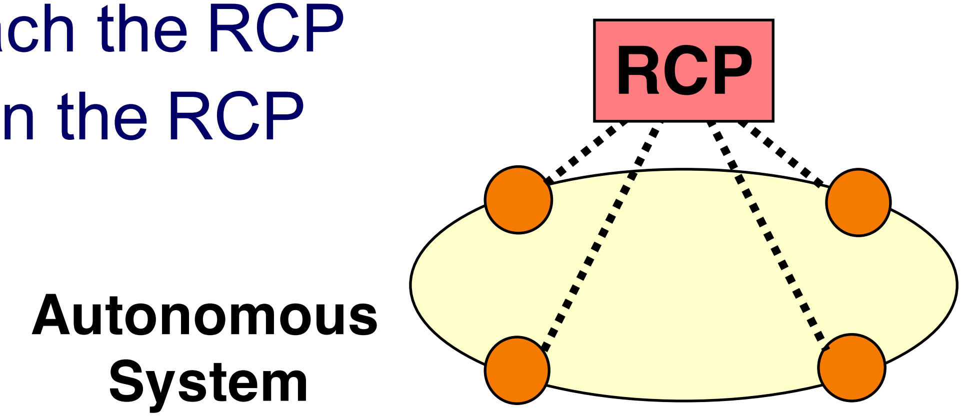
- Control software can run elsewhere
  - The control plane is just software anyway
- State and computation is reasonable
  - E.g., 300K prefixes, a few million changes/day
- System overheads can be amortized
  - Mostly redundant data across routers
- Easier access to other information
  - Layer-2 risks, host measurements, biz goals, …
- Some control could move to end hosts

# Routing Control Platform (RCP)

Removing *Inter*domain Routing from Routers

# Separating *Inter*domain Routing

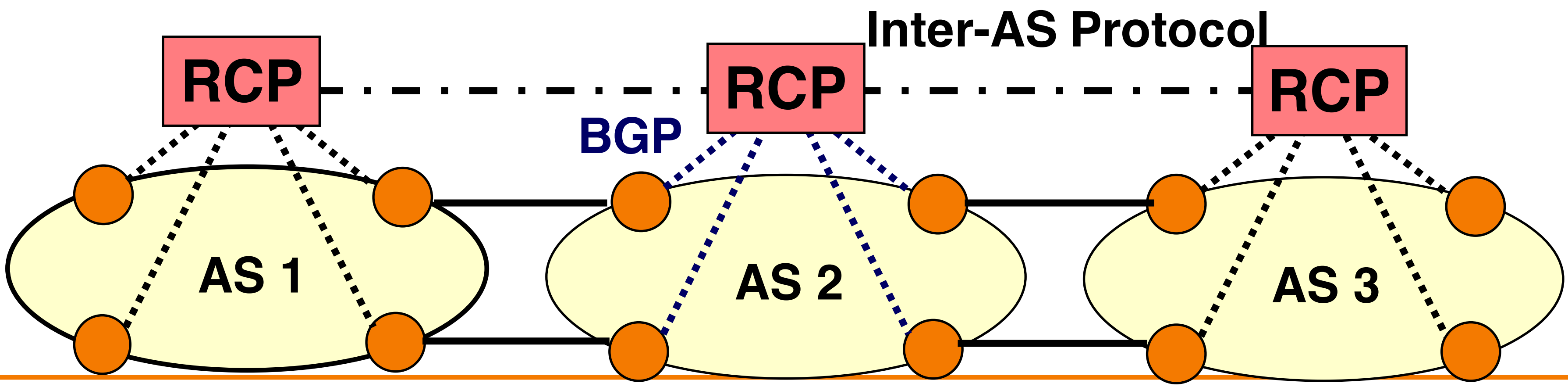
- Compute interdomain routes for the routers
  - Input: BGP-learned routes from neighboring ASes
  - Output: forwarding-table entries for each router
- Backwards compatibility with legacy routers
  - RCP speaks to routers using BGP protocol
  - Installing <destination prefix, next-hop address>
- Routers still run intradomain routing protocol
  - So the routers can reach the RCP
  - To reduce overhead on the RCP



# Incremental Deployability

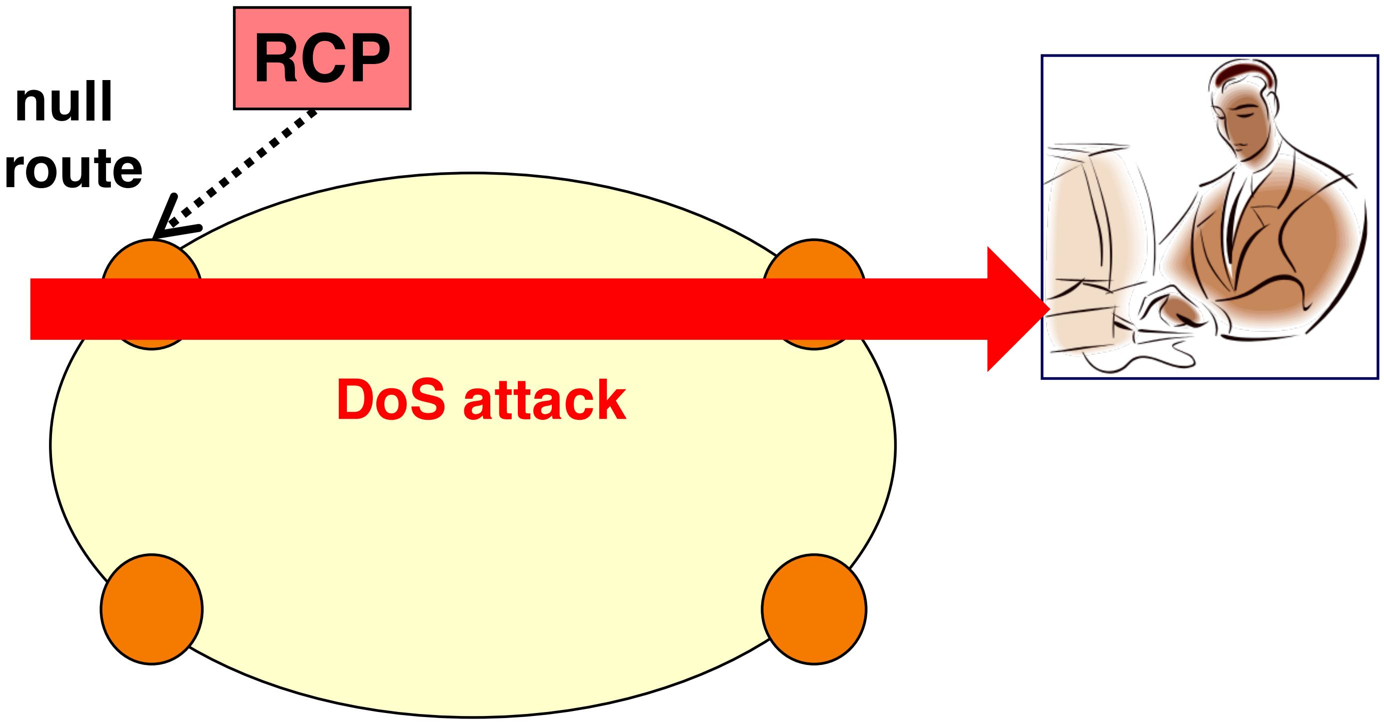
- Backwards compatibility
  - Work with existing routers and protocols
- Incentive compatibility
  - Offer significant benefits, even to the first adopters

RCP tells routers how to forward traffic



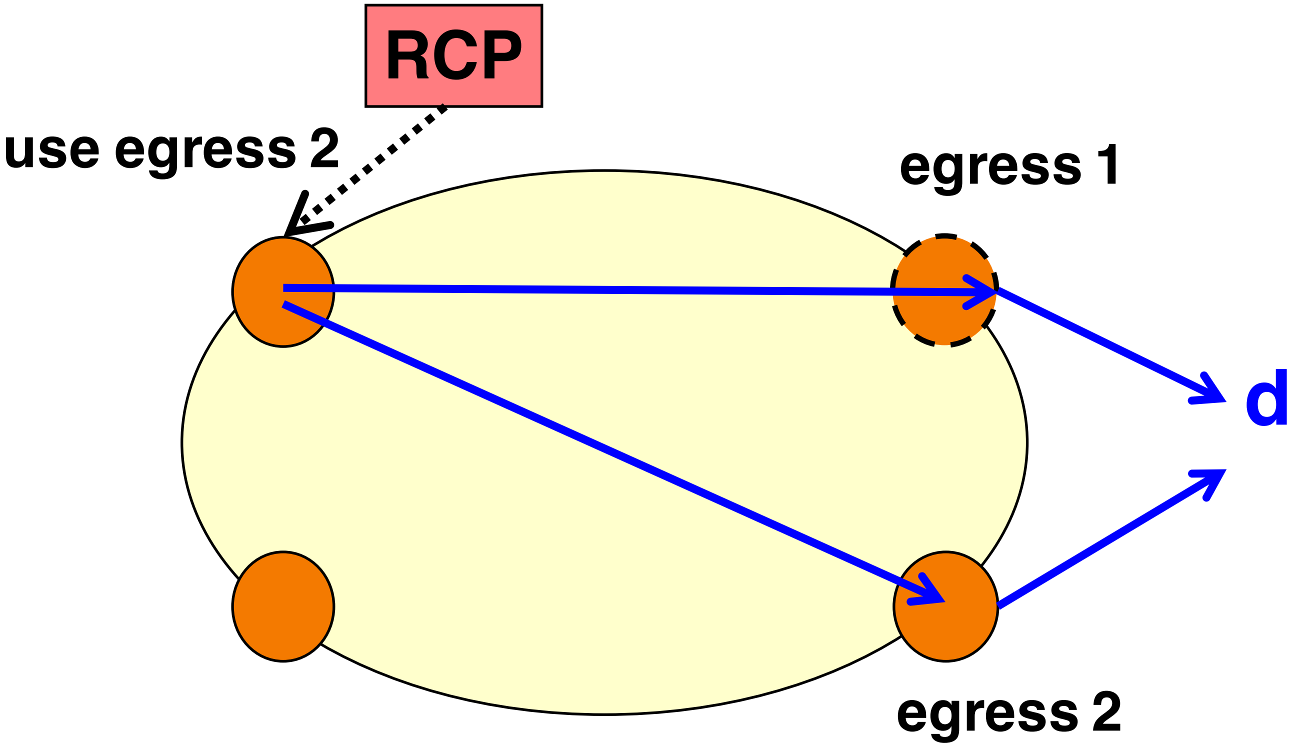
# Example: DoS Blackholing

- Filtering attack traffic
  - Measurement system detects an attack
  - Identify entry point and victim of attack
  - Drop offending traffic at the entry point



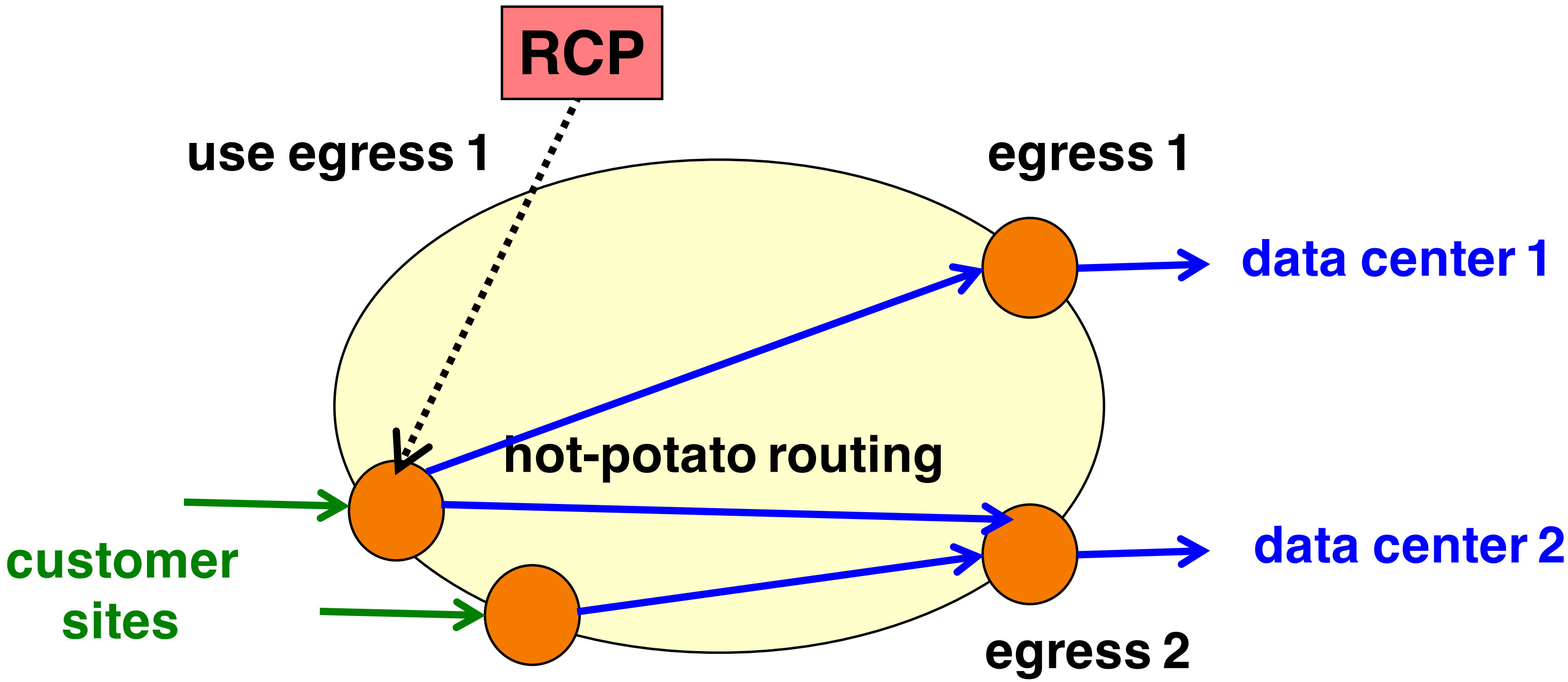
# Example: Maintenance Dry-Out

- Planned maintenance on an edge router
  - Drain traffic off of an edge router
  - Before bringing it down for maintenance



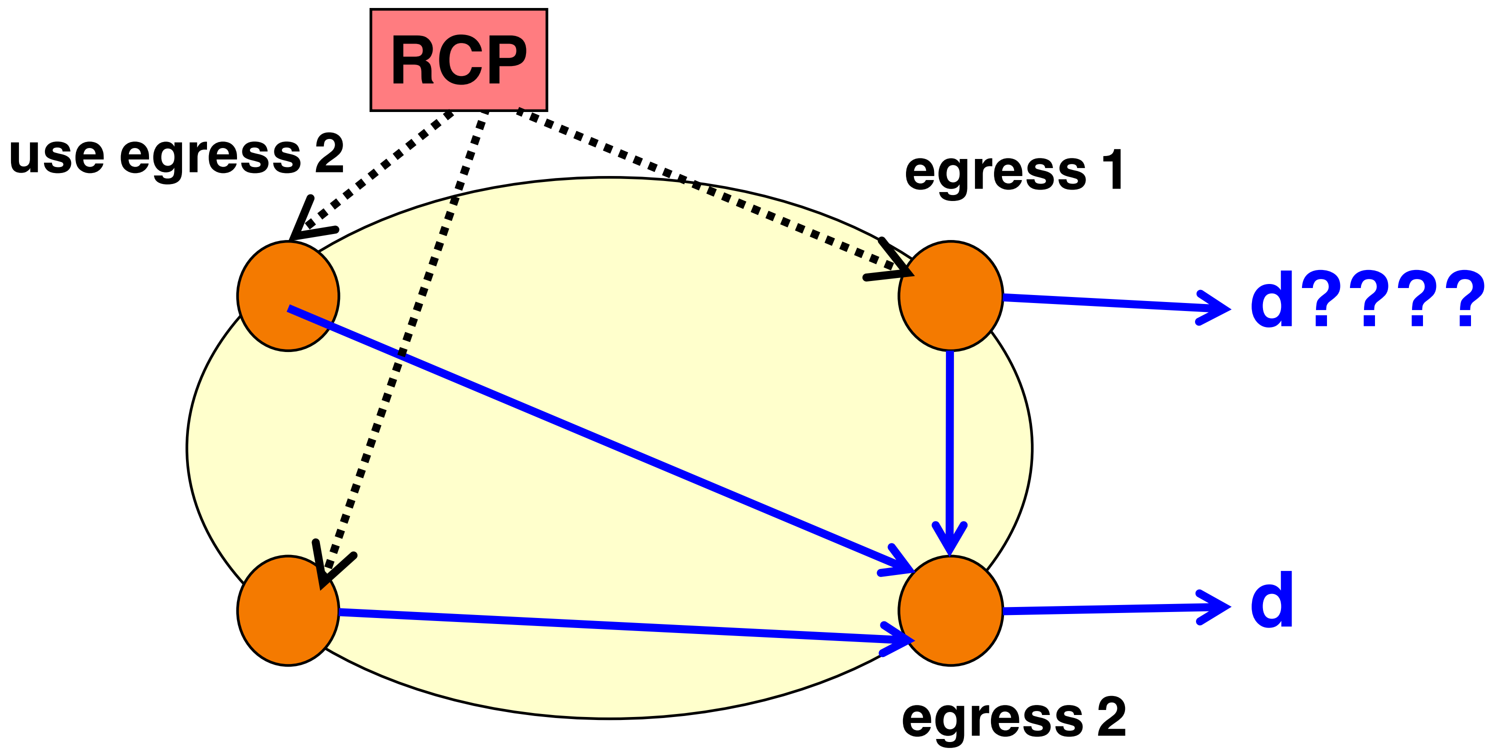
# Example: Egress Selection

- Customer-controlled egress selection
  - Multiple ways to reach the same destination
  - Giving customers control over the decision



# Example: Better BGP Security

- Enhanced interdomain routing security
  - –Anomaly detection to detect bogus routes
  - –Prefer “familiar” routes over unfamiliar



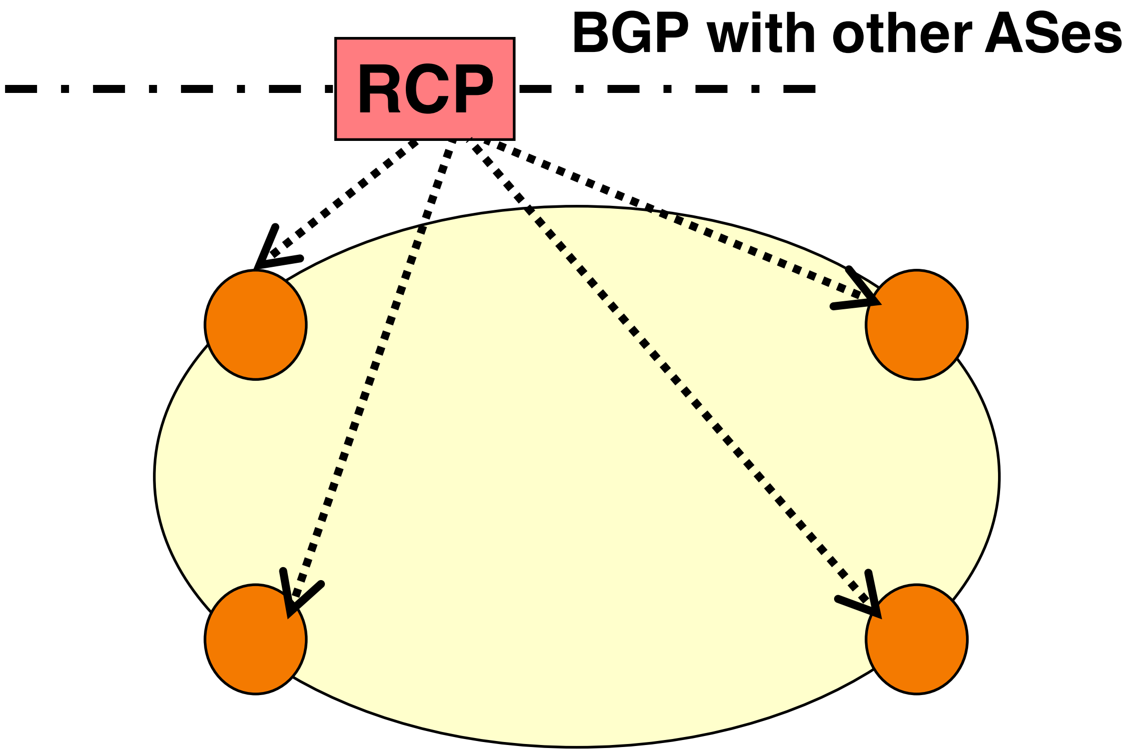
# Example: Saving Router Memory

- Reduce memory requirements on routers
  - Strip BGP route attributes (except prefix and next-hop)
  - Combine related prefixes into a single route

12.0.0.0/16 → nh 1
12.1.0.0/16 → nh 1

↓

12.0.0.0/15 → nh 1



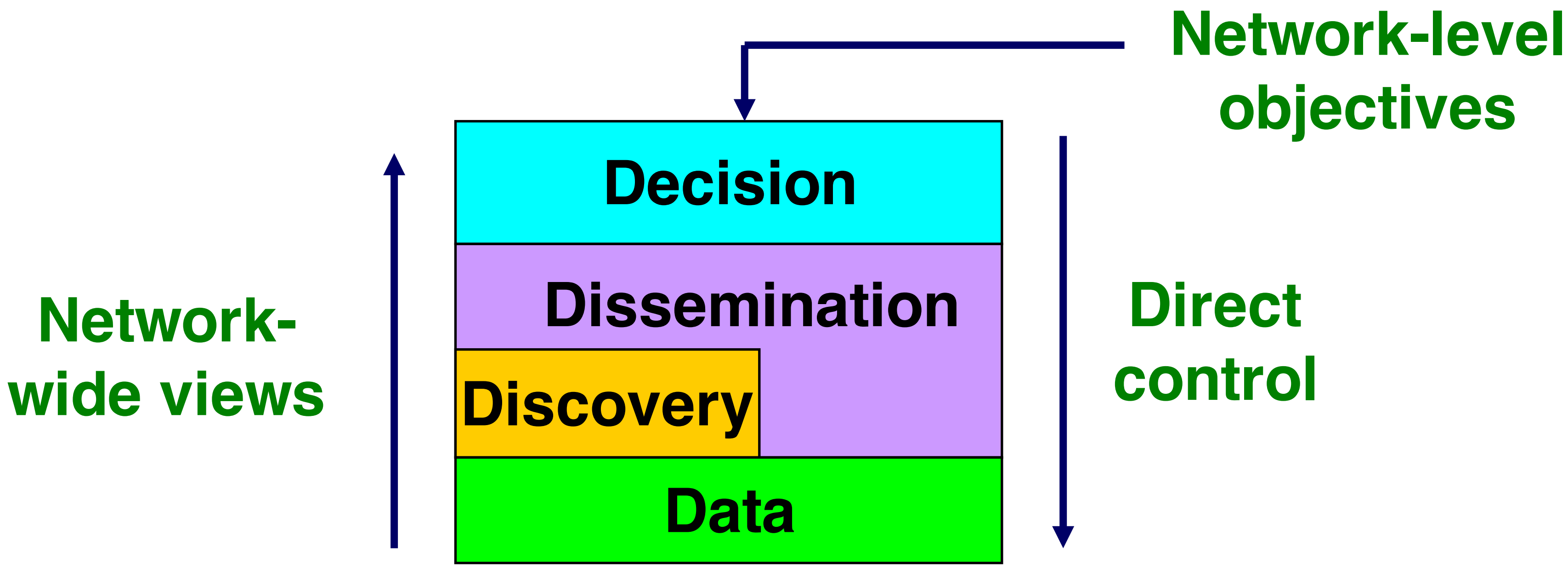
# Clean-Slate 4D Architecture

Generalizing the Approach

# Three Goals of 4D Architecture

- Network-level objectives
  - Configure the *network*, not the routers
  - E.g., minimize the maximum link utilization
  - E.g., connectivity under all layer-two failures
- Network-wide views
  - Complete *visibility* to drive decision-making
  - Traffic matrix, network topology, equipment
- Direct control
  - Direct, sole control over data-plane configuration
  - Packet forwarding, filtering, marking, buffering…

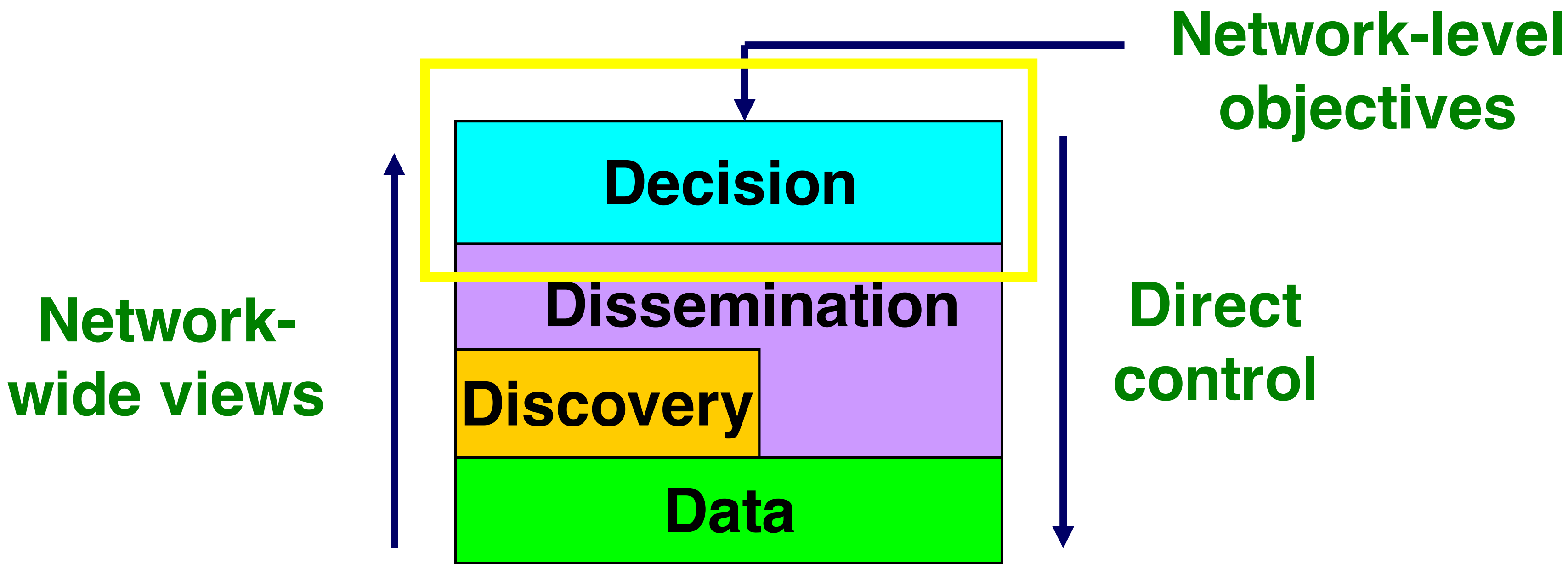
# 4D: The Four Planes




- Decision: all management and control logic
- Dissemination: communication to/from the routers
- Discovery: topology and traffic monitoring
- Data: packet handling

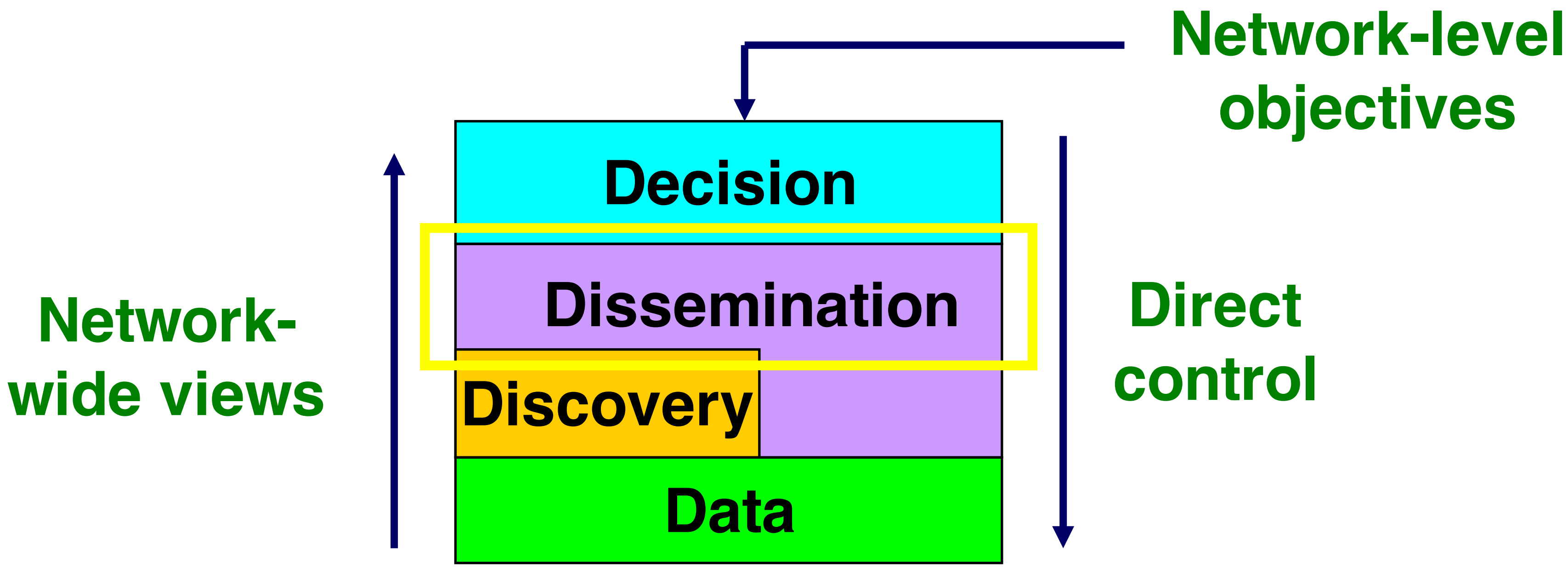
routers

# Decision Plane



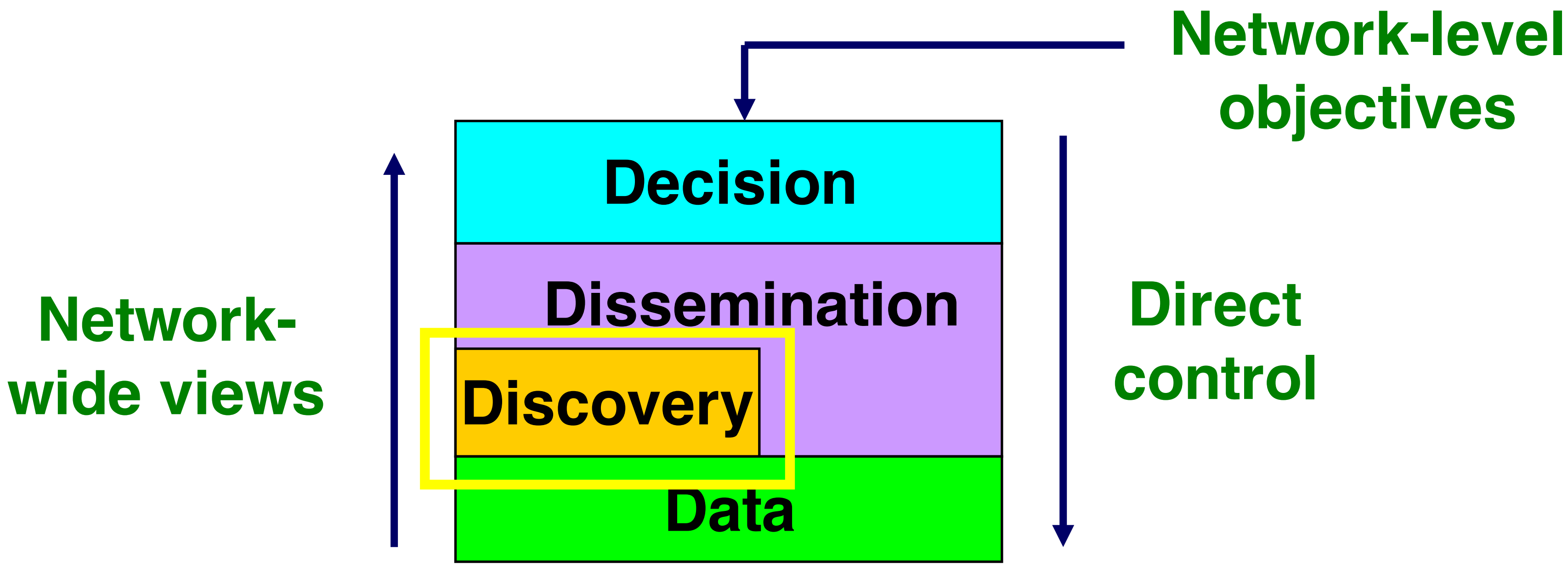

- *All* management logic implemented on centralized servers making *all* decisions
- *Decision Elements* use views to compute data plane state that meets objectives, then directly writes this state to routers

# Dissemination Plane



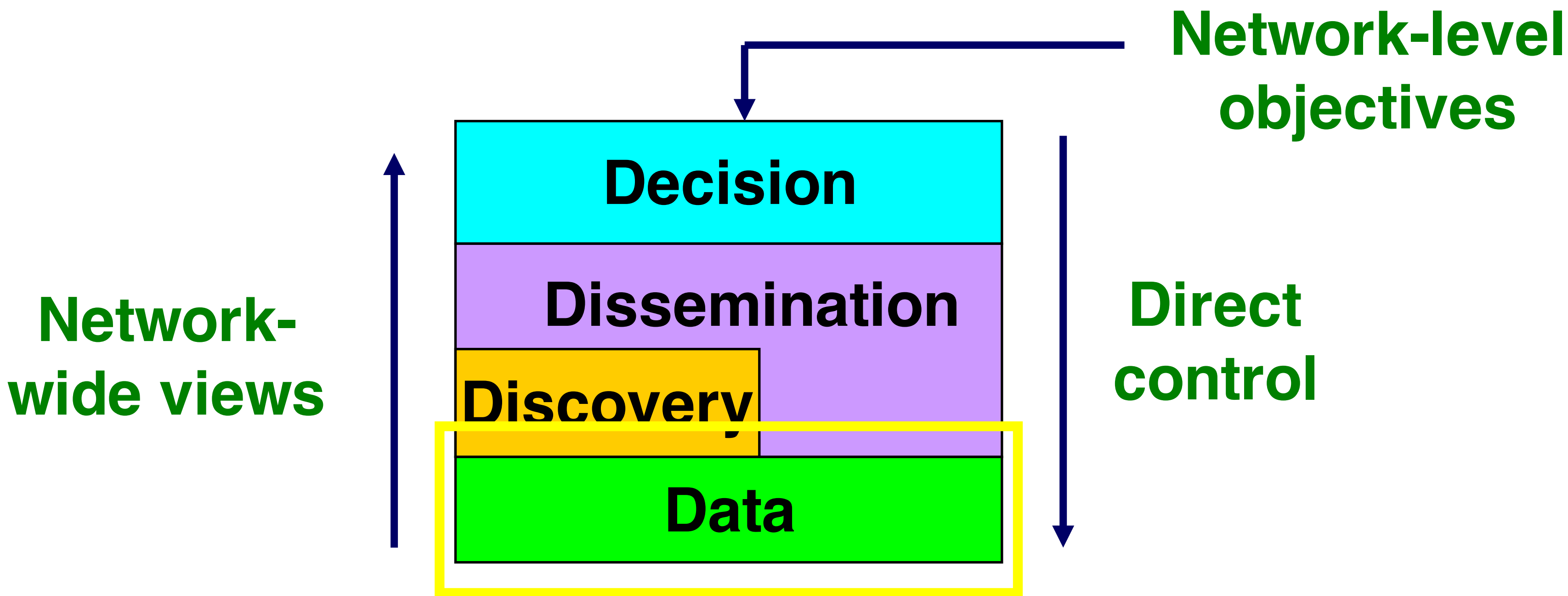

- Provides a robust communication channel to each router – and robustness is the ***only*** goal!
- May run over same links as user data, but logically separate and independently controlled

# Discovery Plane



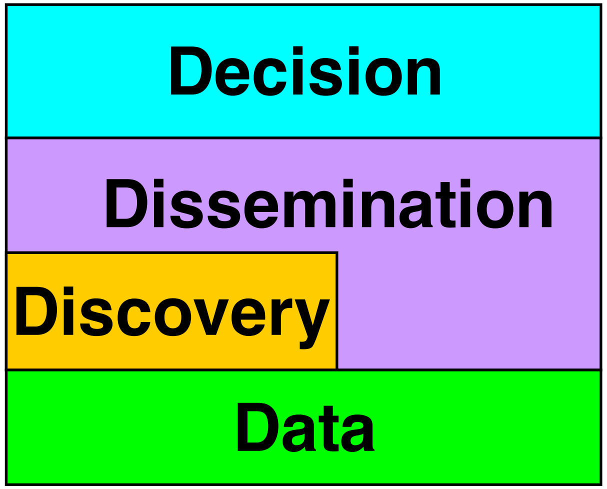

- Each router discovers its own resources and its local environment
- And propagates information (e.g., topology, traffic) to the decision elements via dissemination plane

# Data Plane




- Spatially distributed routers/switches
- Forward, drop, buffer, shape, mark, rewrite, …
- Can deploy with new or existing technology

# RCP as an Example 4D System

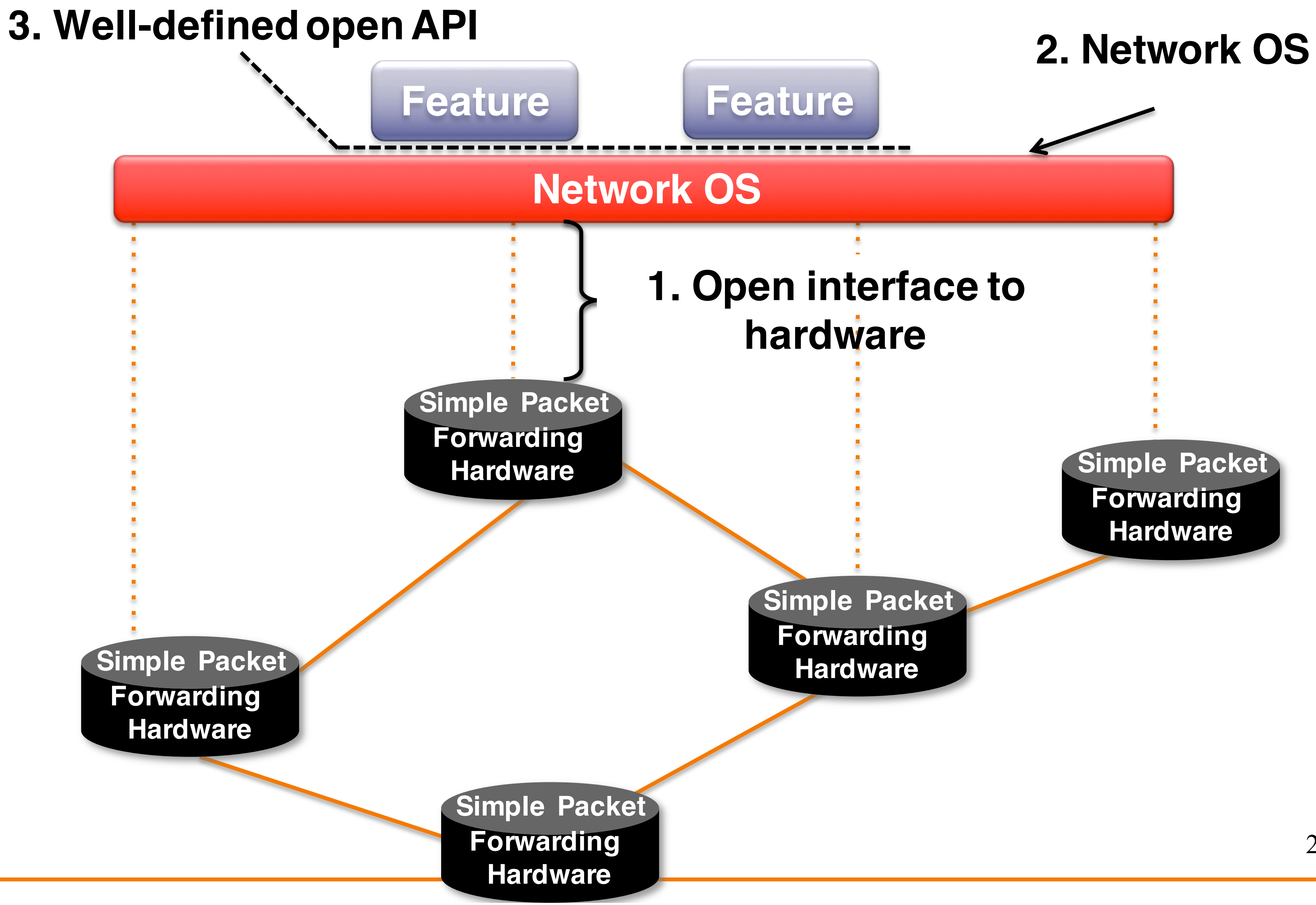



- Decision elements: RCP server
- Dissemination: BGP messages to legacy routers
- Discovery: OSPF (topology) and BGP (routes)
- Data: legacy destination-based IP forwarding

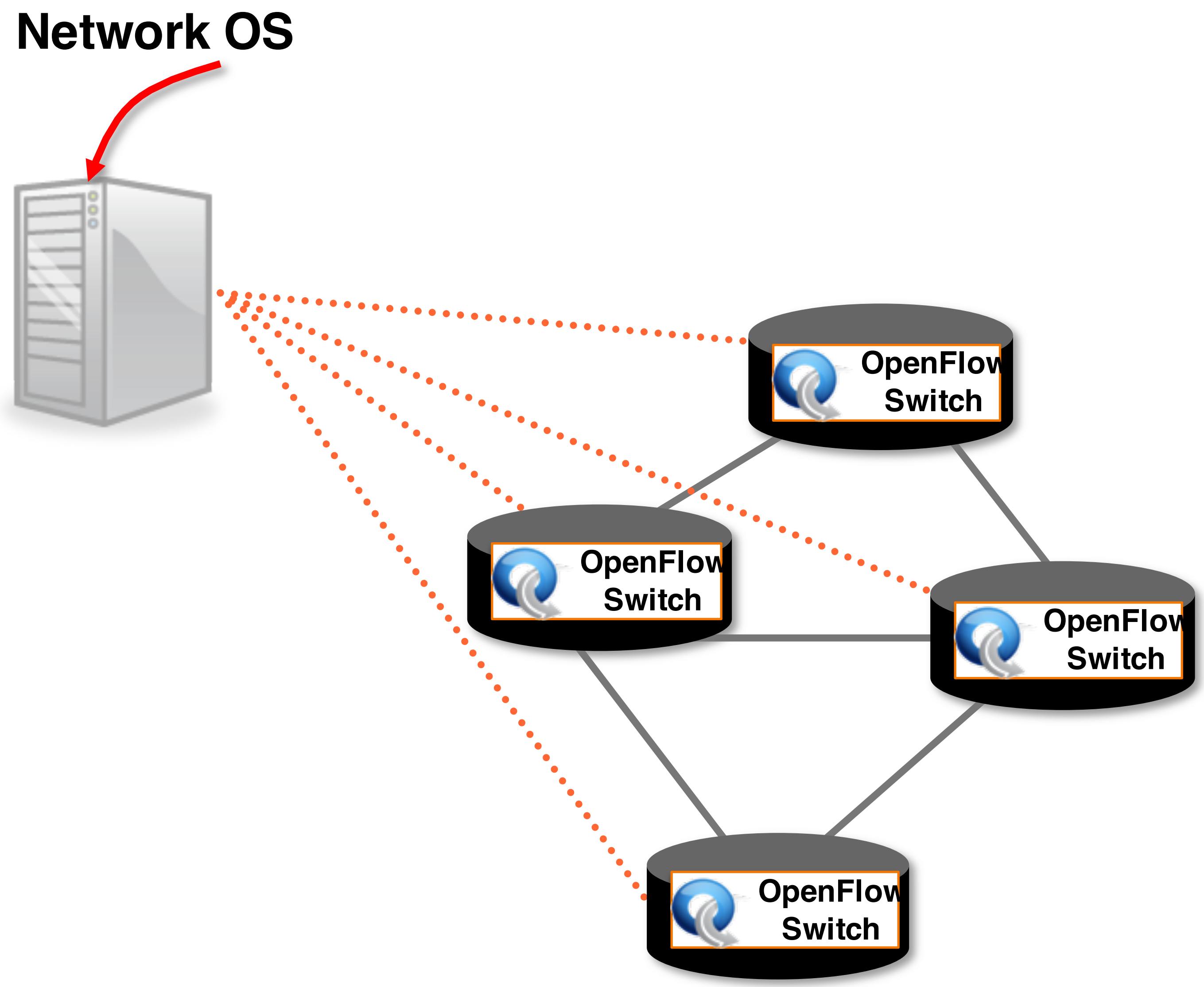
# OpenFlow/NOX

Standard API to Switches, and a Programmable Controller

# Software-Defined Networking

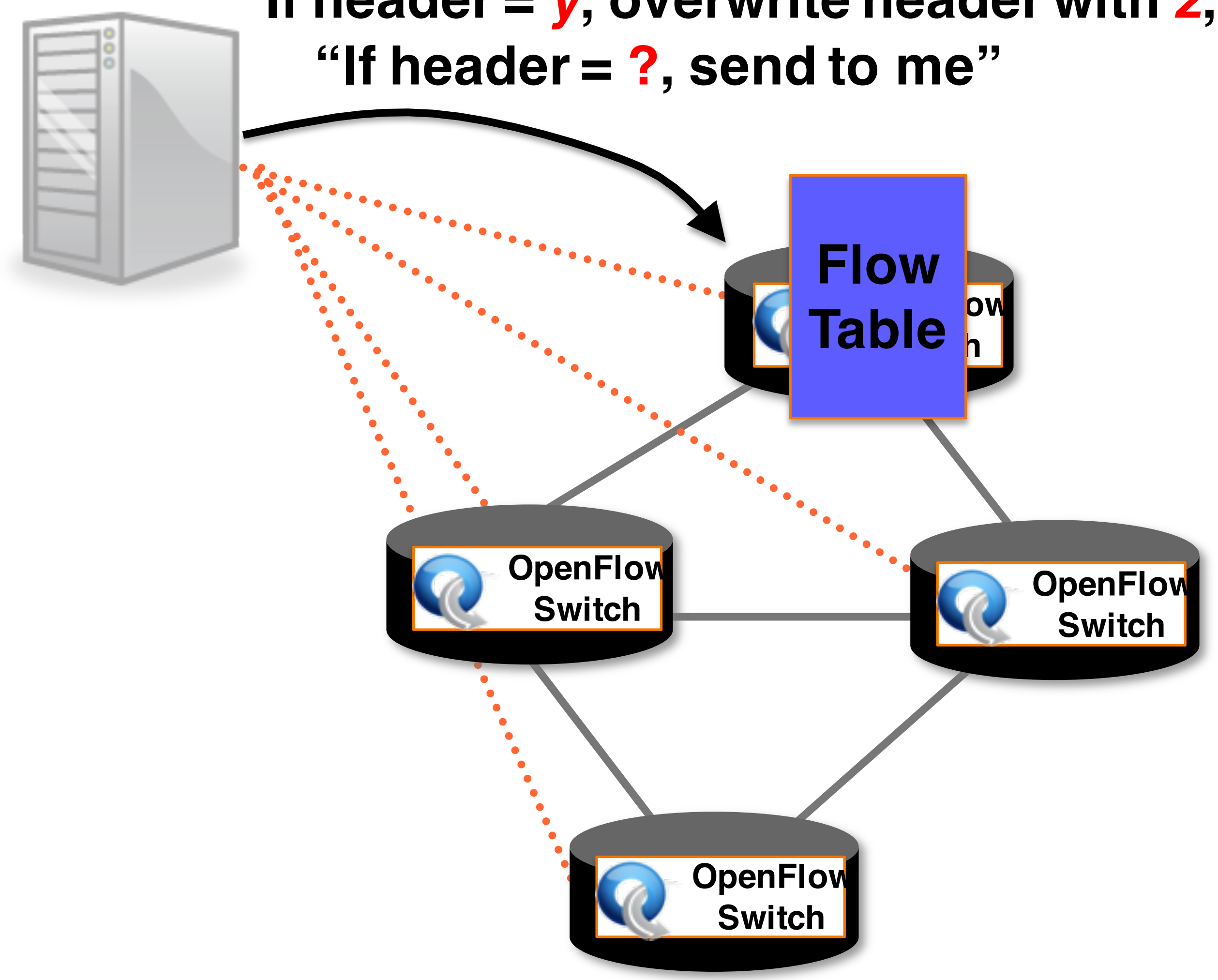


# Separate Control and Data Paths



# Cache Decisions in Data Path

“If header = *x*, send to port 4”
“If header = *y*, overwrite header with *z*, send to ports 5,6”
“If header = ?, send to me”



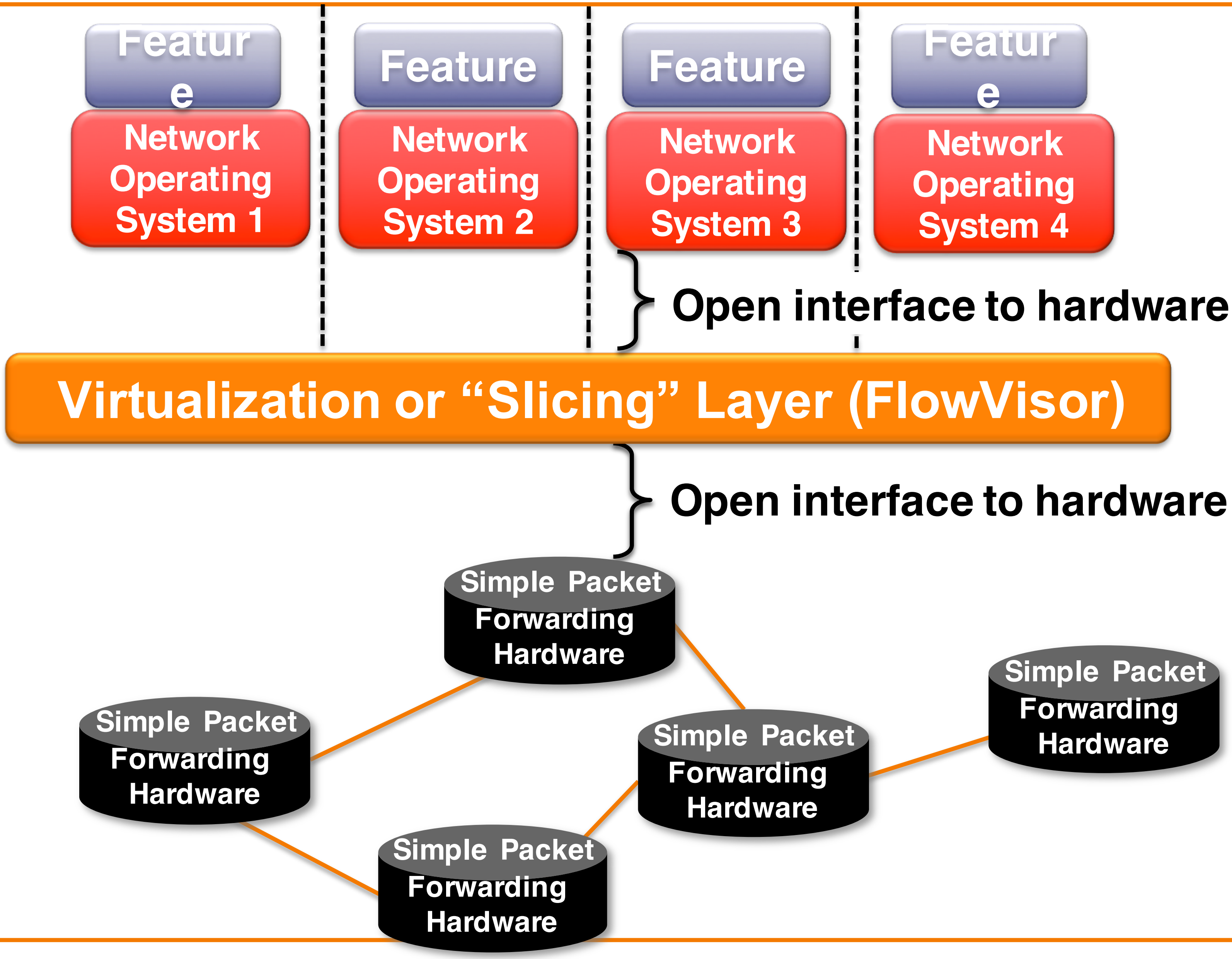
# Data-Path Primitives

- Match arbitrary bits in the packet header


**Match: 1000x01xx0101001x**

  - Match on any header; or new header
  - Allows any flow granularity
- Actions:
  - Forward to port(s), drop, send to controller
  - Overwrite header with mask, push or pop, …
  - Forward at specific bit-rate

# Virtualization of the Network



# Example Applications

- Ethane
  - Flow-level access control
- Plug-n-serve
  - Load balancing over replicated Web servers
- ElasticTree
  - Selectively shutting down equipment to save energy
- VM migration
  - Migrating a virtual machine to a new location
- <Insert your idea here>

**http://www.openflowswitch.org/wk/index.php/OpenFlow_based_Publications**

# Technical Challenges

# Practical Challenges

- Scalability
  - Decision elements responsible for many routers
- Response time
  - Delays between decision elements and routers
- Reliability
  - Surviving failures of decision elements and routers
- Consistency
  - Ensuring multiple decision elements behave consistently
- Security
  - Network vulnerable to attacks on decision elements
- Interoperability
  - Legacy routers and neighboring domains

# RCP: Scalable Implementation

- Eliminate redundancy
  - –Store a *single* copy of each BGP-learned route
- Accelerate lookups
  - –Maintain *indices* to identify affected routers
- Avoid recomputation
  - –Compute routes *once* for group of related routers
- Handle only BGP routing
  - –Leave *intradomain* routing to the routers

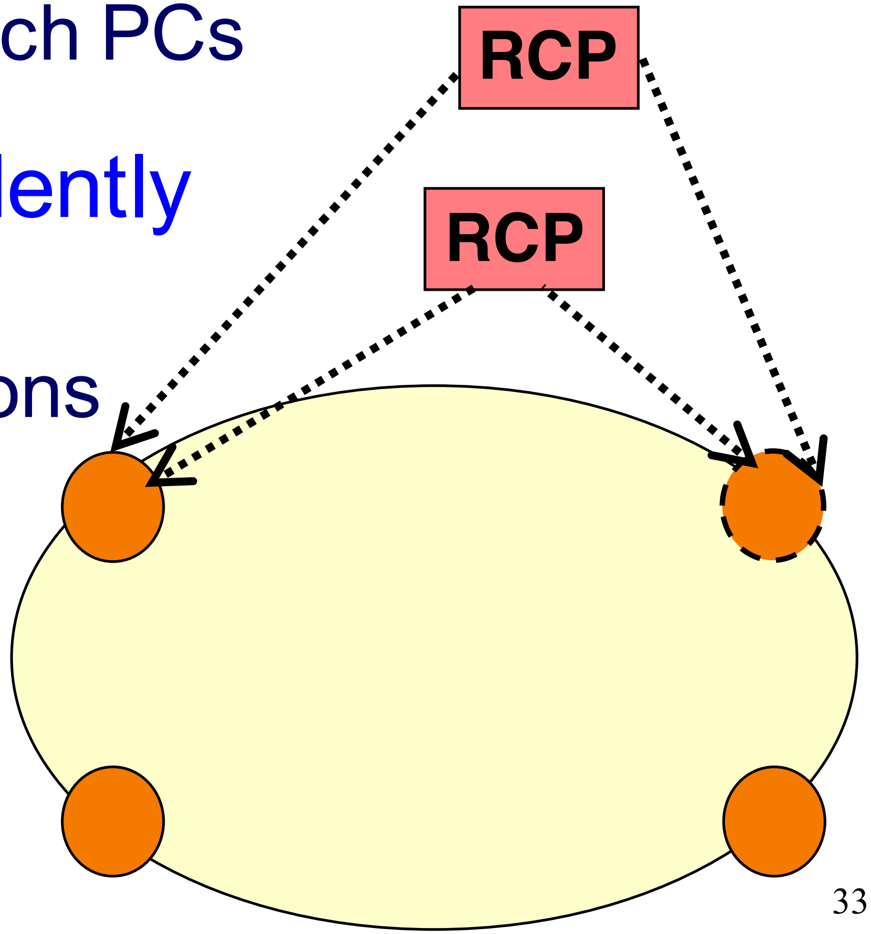
An extensible, scalable, "smart" route reflector

# Runs on a Single High-End PC

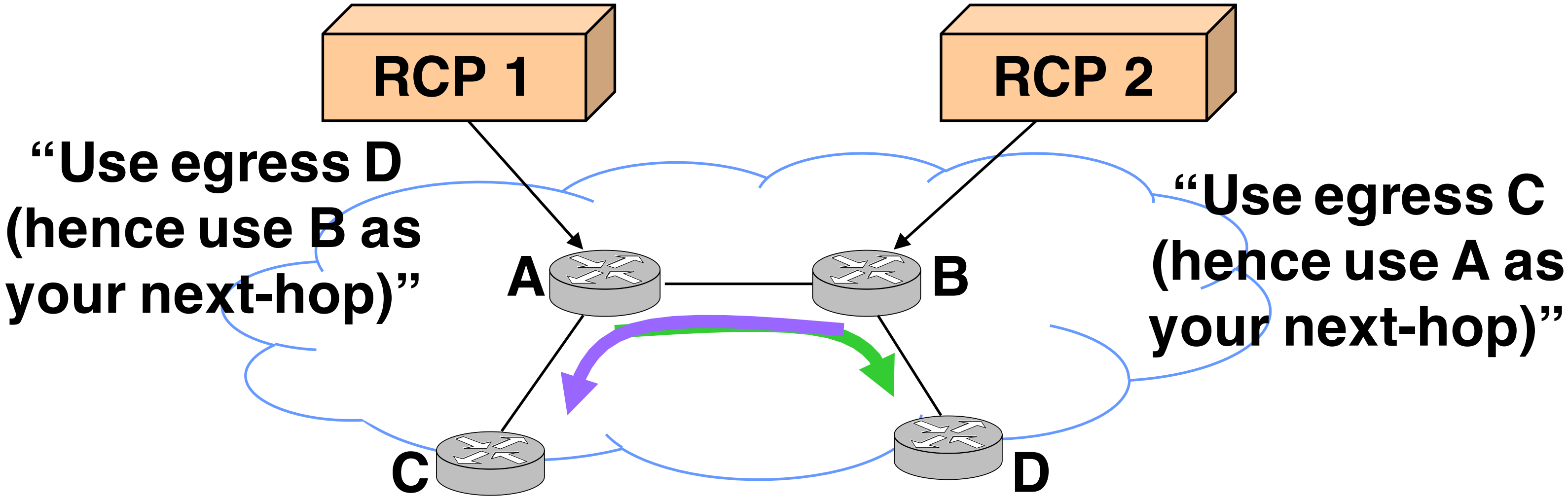
- Home-grown implementation on top of Linux
  - Experiments on 3.2 Ghz P4 with 4GB memory
- Computing routes for all AT&T routers
  - Grouping routers in the same point-of-presence
- Replaying all routing-protocol messages
  - BGP and OSPF logs, for 203,000 IP prefixes
- Experimental results
  - Memory footprint: 2.5 GB
  - Processing time: 0.1-20 msec

# Reliability

- Simple replication
  - Single PC can serve as an RCP
  - So, just run *multiple* such PCs
- Run replicas independently
  - Separate BGP update feeds and router sessions
  - Same inputs, and the same algorithm
  - No need for replica consistency protocol

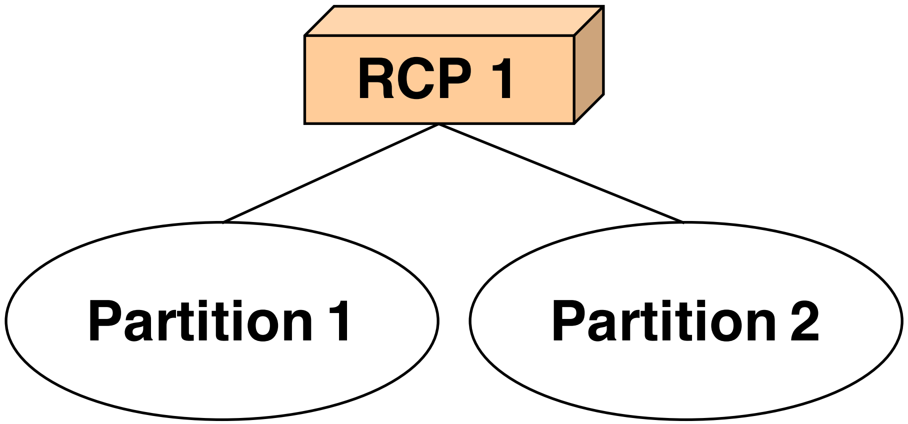


# Potential Consistency Problem



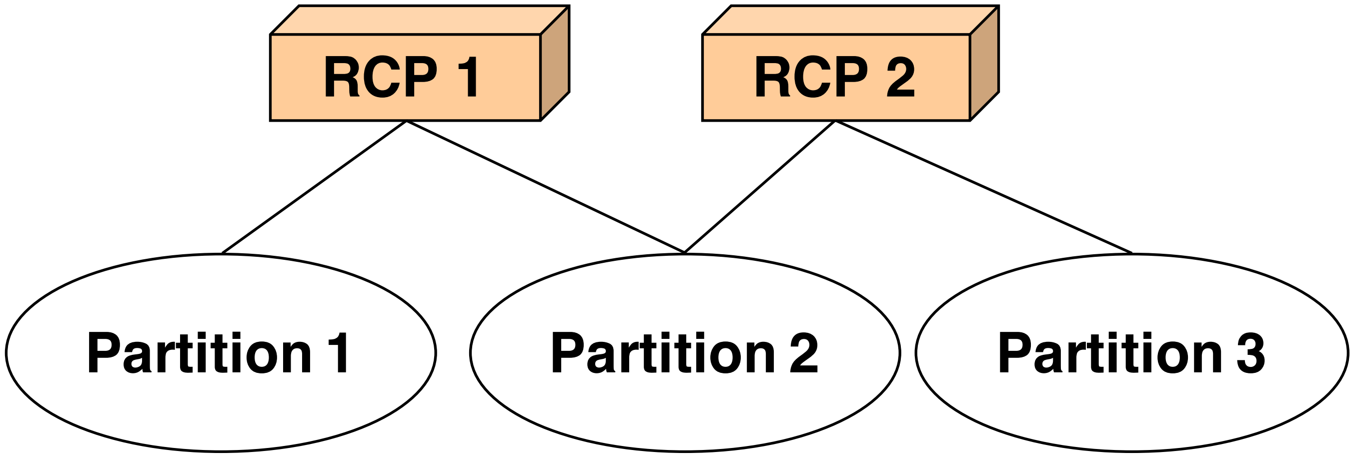

- Need to ensure routes are consistently assigned
  - Even in presence of failures/partitions
- Fortunately…
  - Flooding-based IGP means each RCP knows what partition(s) it connects to

# Single RCP Under Partition




- Solution: Only use state from router's partition in assigning its routes
  - Ensures next hop is reachable

# Multiple RCPs Under Partition




- Solution: RCPs receive same IGP/BGP state from each partition they can reach
  - IGP provides complete visibility and connectivity
  - RCS only acts on partition if it has complete state for it

**→No consistency protocol needed to guarantee consistency in steady state**

# ONIX (OSDI'10 Paper)

- Network Information Base (NIB)
  - Represented as a graph of objects
  - Applications can read and write the NIB
  - Automatically updates switches and controllers
- State distribution tools
  - Replicated transactional (SQL) storage
    - Strong consistency for critical, stable state
    - E.g., switch topology
  - One-hop memory based DHT
    - Eventual consistency for less-critical, dynamic state
    - E.g., IP-to-MAC address mapping

# ONIX (OSDI'10 Paper)

- Distributed coordination
  - Integrated with ZooKeeper
  - Useful for leader election, locking, barriers, etc.
- Scalability
  - Partition: different tasks, switches, or parts of the NIB,
  - Aggregate: combine statistics and topology information
- Reliability
  - Network failures: application's responsibility
  - Reachability to ONIX: reliable protocol, multipath, etc.
  - ONIX failure: distributed coordination amongst replicas

# Conclusions

- Today's routers and switches
  - Too complicated
  - Too difficult to manage
  - Too hard to change
- Dumb routers, smart decision elements
  - Routers forward packets & collect measurement
  - … at the behest of the decision elements
- Many research problems remain
  - Networking meets distributed systems!