Resource Management and QoS

EECS 489 Computer Networks

http://www.eecs.umich.edu/~zmao/eecs489

Z. Morley Mao

Tuesday Oct 12, 2004

Acknowledgement: Some slides taken from Kurose&Ross and Katz&Stoica

Quality of Service (QoS)

- The Internet's most contentious subject
- The Internet's most embarrassing failure
  - almost nothing was accomplished
- The textbook's worst chapter
  - a rosy description of bad work

Today's Lecture

- Will be about what "could be", not what is
  - today's Internet does not have, nor will soon have, a reasonable QoS solution
- Focus will be on what one could accomplish with simple (and not-so-simple) mechanisms
  - you will only be expected to know basic concepts
- I will not discuss current deployed mechanisms
  - an ugly hodge-podge of hacks

What's the Problem?

- Internet gives all flows the same "best effort" service
  - no promises about when or whether packets will be delivered
- Not all traffic is created equal
  - different "owners", different application requirements
  - some applications require service "assurances"
- How can we give traffic different "quality of service"?
  - Thus begins the problem of QoS

Three Basic Problems

- Want to control how a link is shared:
  - Link sharing
- Want to give some traffic better service
  - Differentiated service
- Want to gives some flows “assured” service
  - Integrated service (and perhaps differentiated service)

A Different Taxonomy

- Giving better service can differ along three dimensions:
  - relative versus absolute
  - dropping versus delay
  - flows versus aggregates
- Each of these choices requires different set of mechanisms
  - router scheduling and dropping decisions
  - signaling protocols

Three Basic Questions

- How does a router service this packet?
  - scheduling (various forms of priority and RR)
  - dropping (fancy versions of RED)
- How did the router know what to do with this packet?
  - bits in packet header or explicit signaling
- How can one control the level of traffic?
  - service level agreements (SLAs) or admission control

Back to Thee Basic Problems

- Link sharing (one slide)
- Differentiated Services (long)
- Integrated Services (even longer)

Link Sharing

- Two organizations share an access link and want to share it equally
- One approach: partition the link
- Second approach: use FQ, with one queue for each organization's packets
- Which is better?

Differentiated Services

- Some traffic should get better treatment
  - application requirements: interactive vs bulk transfer
  - economic arrangements: first-class versus coach
- What kind of better service could you give?
  - measured by drops, or delay (and drops)
- How do you know which packets to give better service to?
  - bits in packet header

Traffic Limitations

- Can’t give all traffic better service!
- Must limit the amount of traffic that gets better service
- Service Level Agreements (SLA)
  - source agrees to limit amount of traffic in given class
  - network agrees to give that traffic “better” service
    - for a price!
  - economics play an important (fatal?) role in QoS

DiffServ “Code Points”

- Use six of the ToS bits in IP packet header
- Define various “code points”
- Each code point defines a desired per-hop behavior
  - a description of the service the packet should get
  - not a description of the router implementation of that service

“Expedited Forwarding”

- Give packet minimal delay and loss service
  - e.g., put EF packets in high priority queue
- To make this a true “absolute” service,
  - all SLAs must sum to less than the link speed
  - unlikely
- More likely, a way to assure relatively low delay

Is Delay the Problem?

- With RED, most queues are small
- Packets are dropped when queue starts to grow
- Thus, delays are mostly speed-of-light latency
- Service quality is mostly expressed by drop-rate
- Want to give traffic different levels of dropping

“Assured Forwarding”

- Packets are all serviced in order
  - makes TCP implementations perform well
- But some packets can be marked as low-drop and others as high-drop
  - think of it as priority levels for dropping
- Can be implemented using variations of RED
  - different drop probabilities for different classes

Example

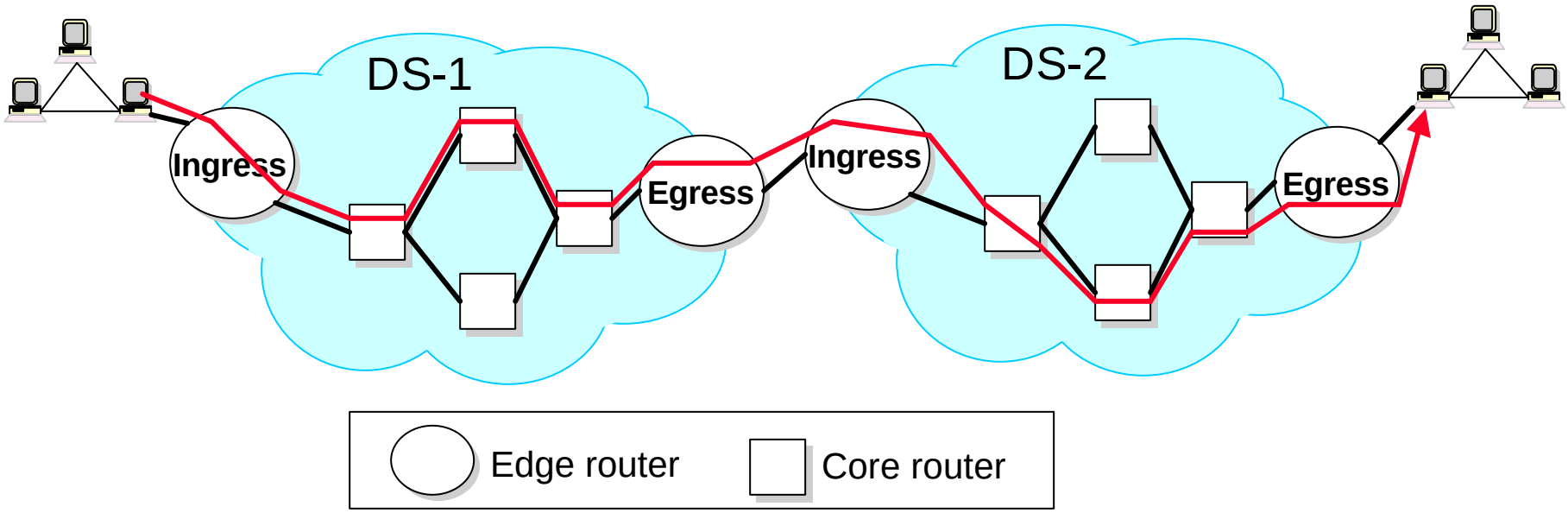
- 10% premium traffic, 90% ordinary traffic
- Overall drop rate is 5%
- Can give premium traffic 0% drops, and ordinary traffic a 5.55% drop rate
- Can get a large improvement in service for the small class of traffic without imposing much of a penalty on the other traffic
  - count on SLAs to control premium traffic

Advantages of DiffServ

- Very simple to implement
- Can be applied to different granularities
  - flows
  - institutions
  - traffic types
- Marking can be done at edges or by hosts
- Allows easy peering (bilateral SLAs)

DiffServ Peering

- Ingress routers
  - Police/shape traffic
  - Set Differentiated Service Code Point (DSCP) in DiffServ (DS) field
- Core routers
  - Implement Per Hop Behavior (PHB) for each DSCP
  - Process packets based on DSCP



Disadvantages of DiffServ

- Service is still “best effort”, just a better class of best effort
  - except for EF, which has terrible efficiency
  - all traffic accepted (within SLAs)

- Some applications need better than this
  - certainly some apps need better service than today’s Internet delivers
  - but perhaps if DiffServ were widely deployed premium traffic would get great service (recall example)
  - nonetheless, let’s plunge ahead....

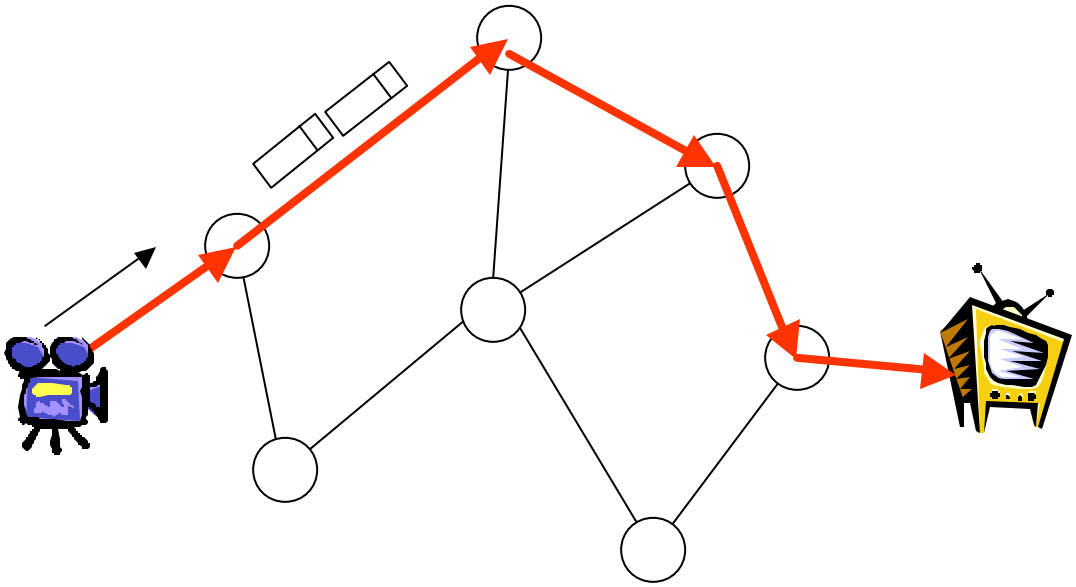
Integrated Services

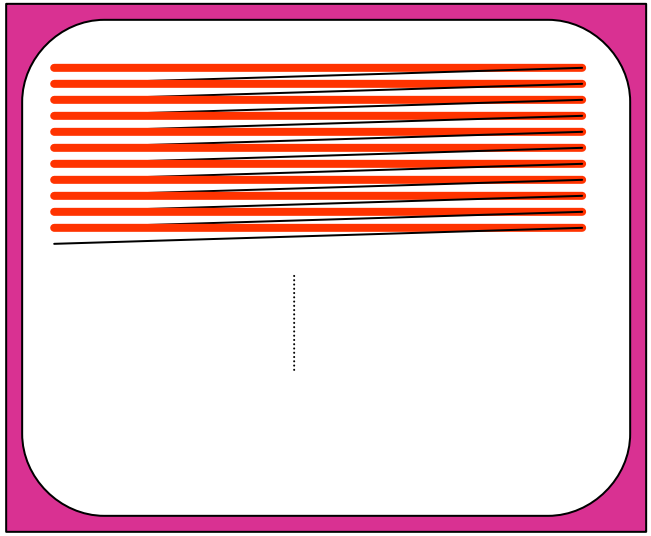
- An attempt to integrate service for “real-time” applications into the Internet
- Known as IntServ
- A total, massive, and humiliating failure
  - 1000s of papers
  - IETF standards
  - and no deployment....

Key Differences

- All assurances on per-flow basis
- Traffic can be turned away
- Note:
  - all this co-exists with best-effort service
  - similar mechanisms proposed for ATM but
    - QoS central in ATM, best-effort an afterthought
    - Best-effort central in Internet, QoS an afterthought

Example: Video





Simplify by assuming that
Camera sends at a fixed rate

Nick Mckeown

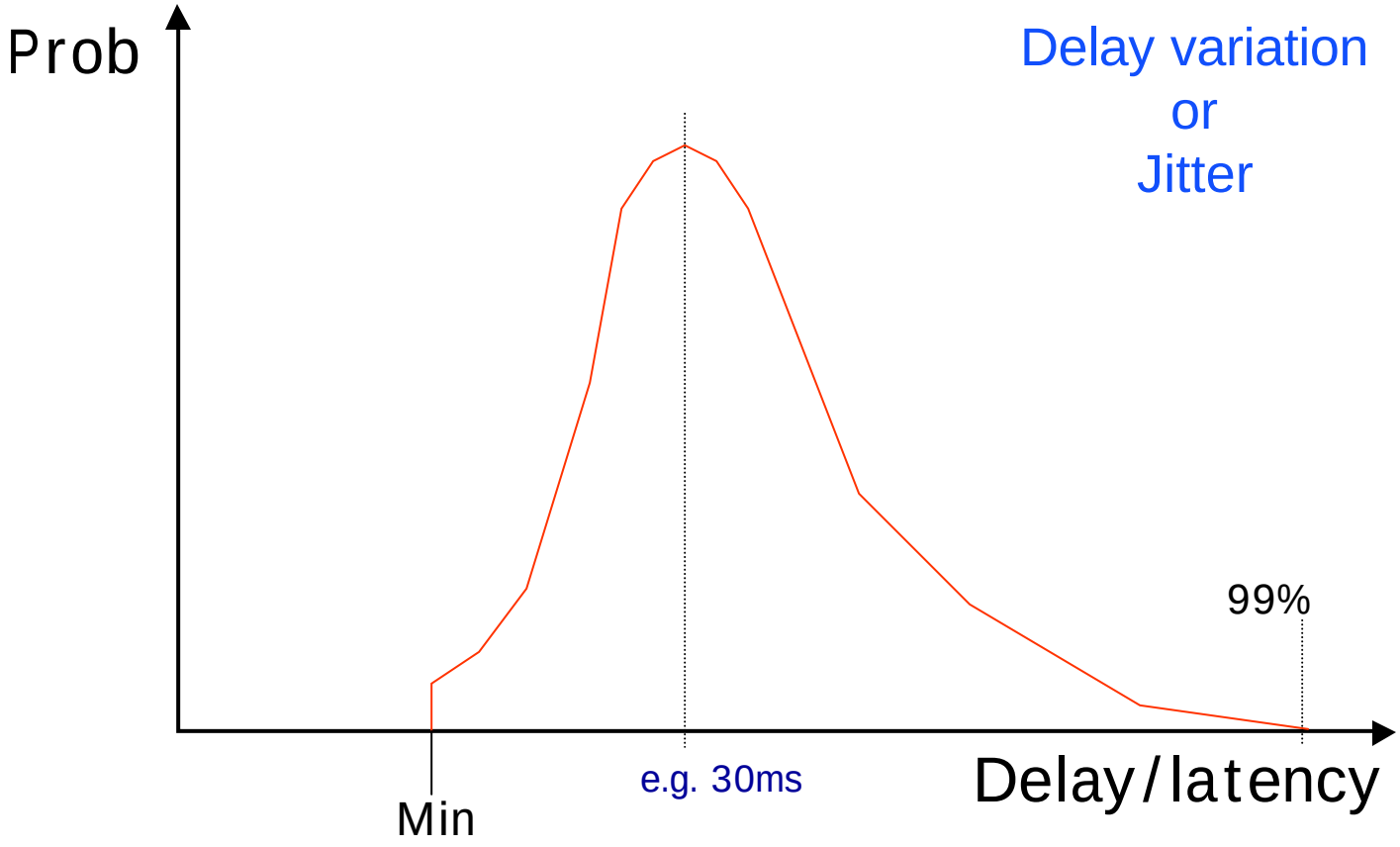
Circuit-Switched Networks

- Each packet experiences exactly the same delay
- Packet data is displayed as soon as it arrives
- Signal at receiving end is faithful representation

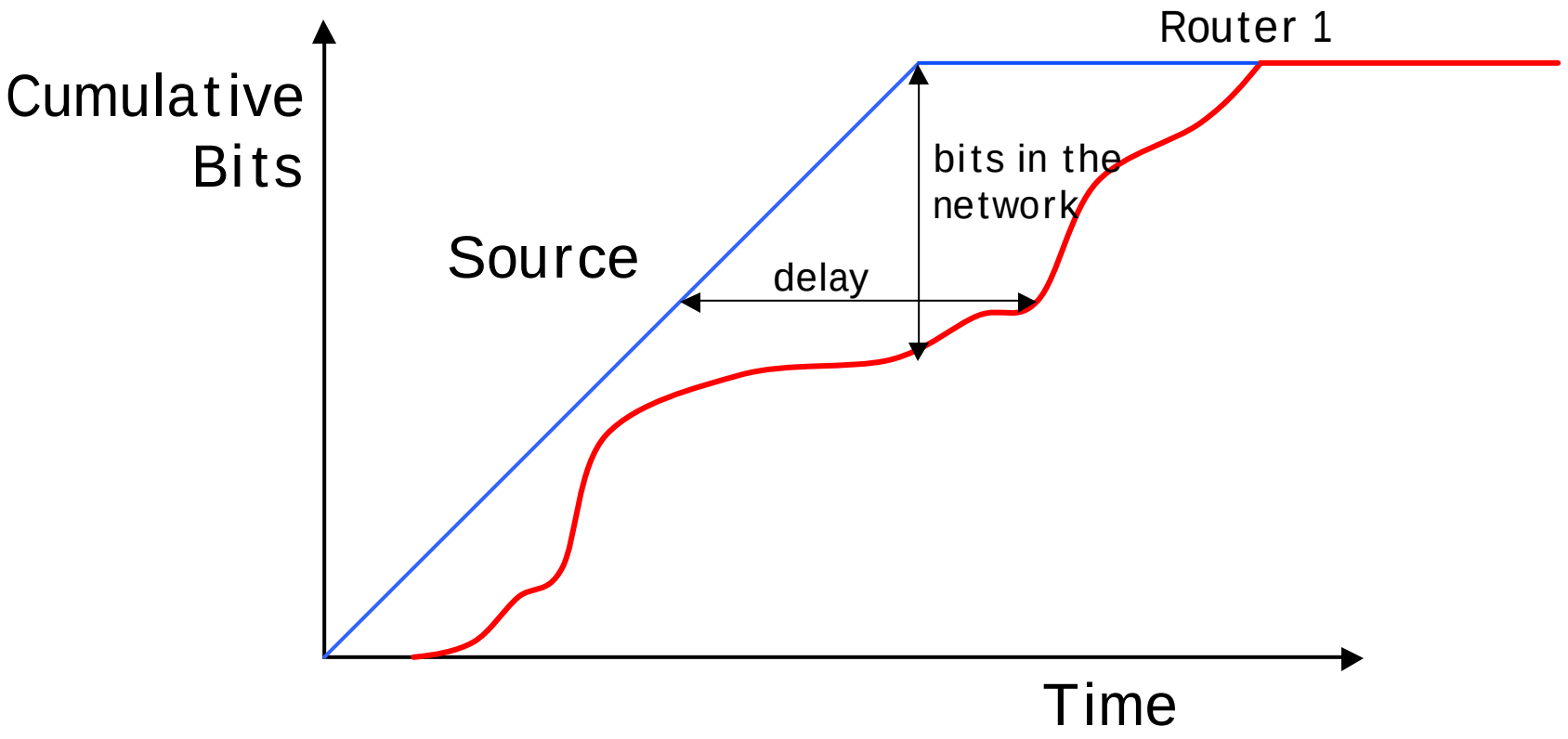
Internet

- Individual packets experience different delays
- Can’t treat network as “wire”
- Application must adapt to network service

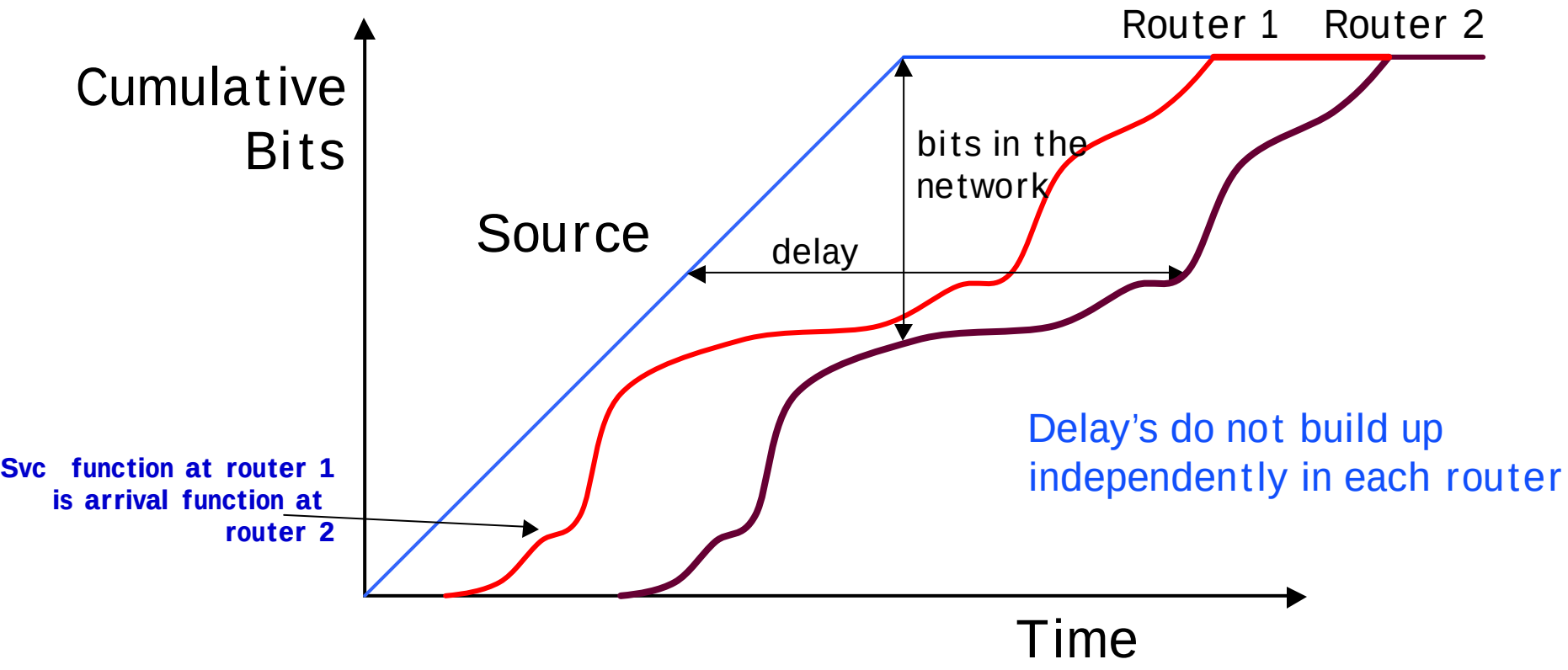
Router Effect on Delay



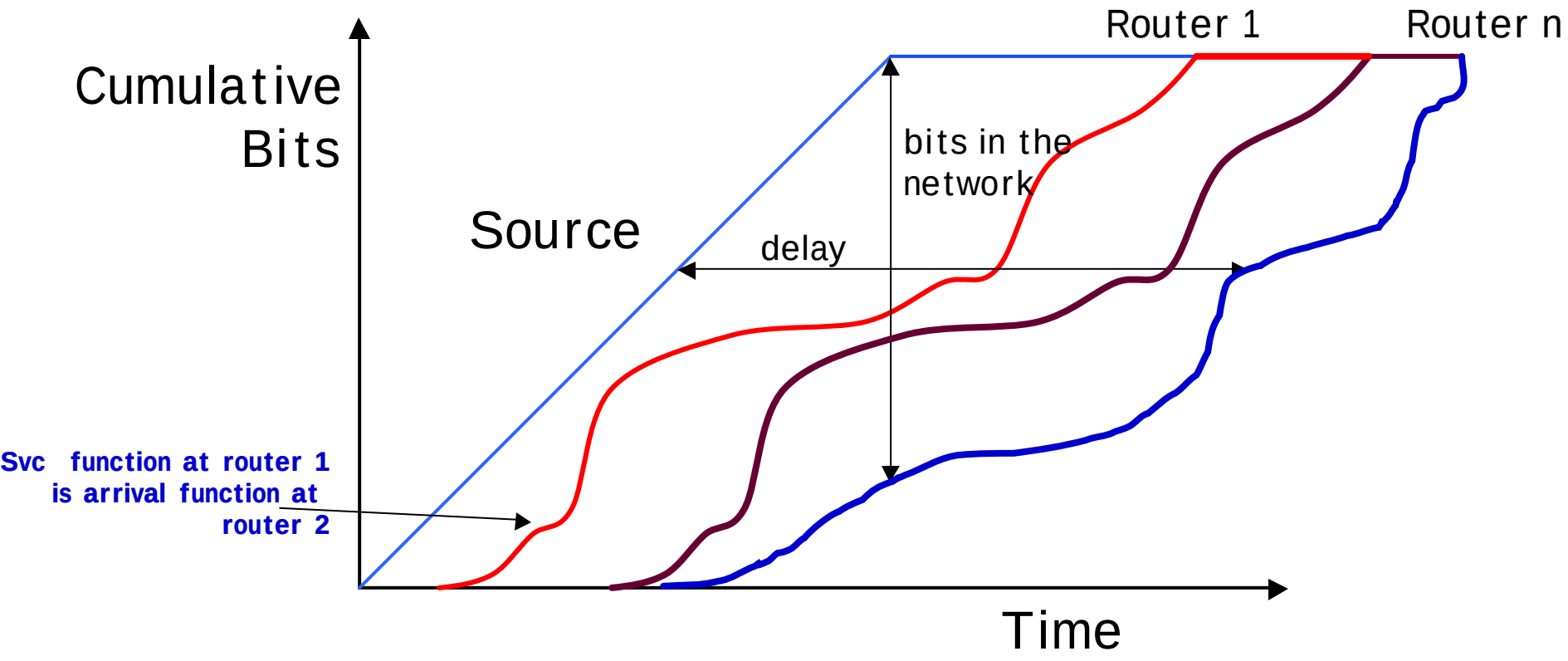
Router Effects on Traffic



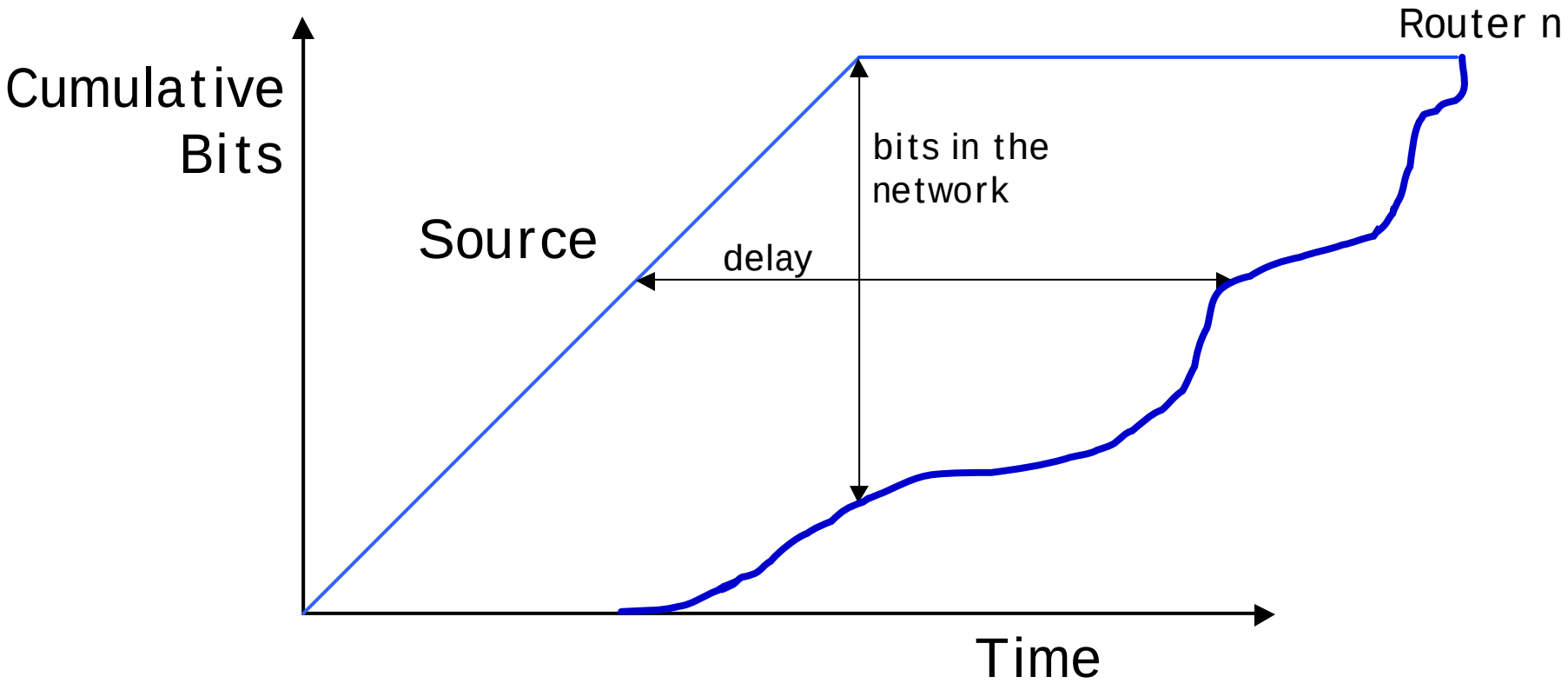
Network Effects on Traffic



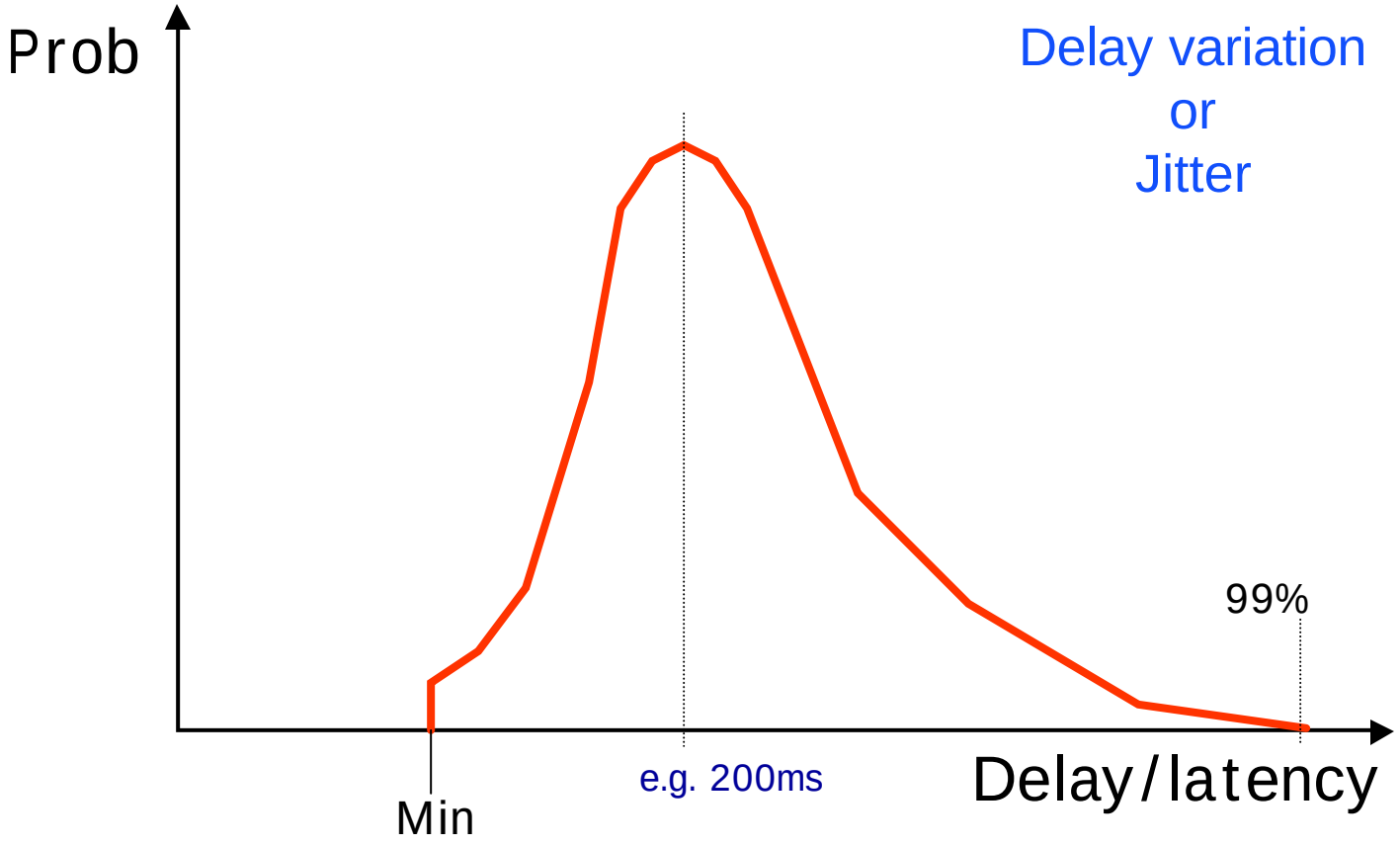
Network Effects on Traffic



Network Effects on Traffic



Network Effect on Delay



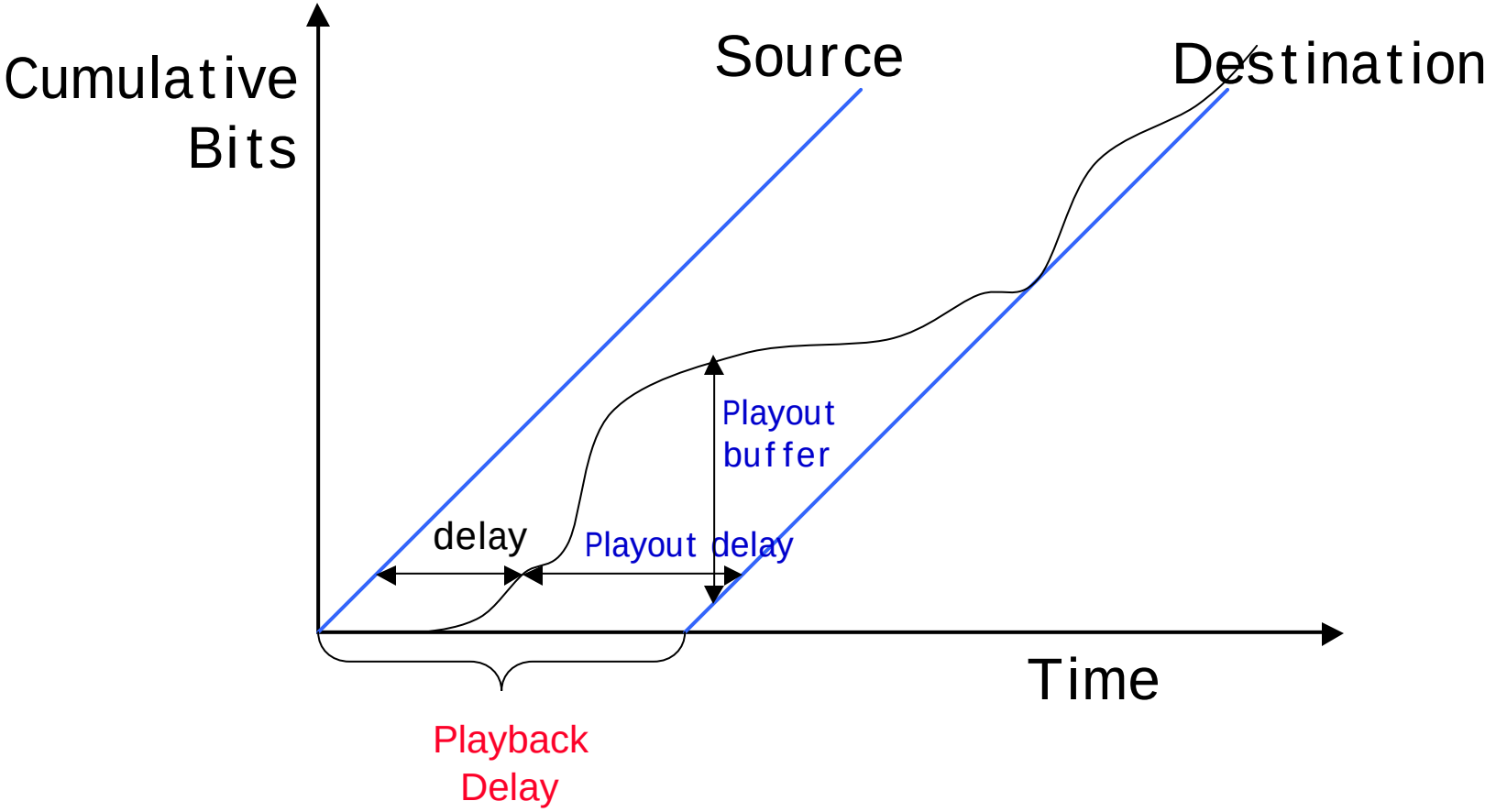
Nick Mckeown

Choices

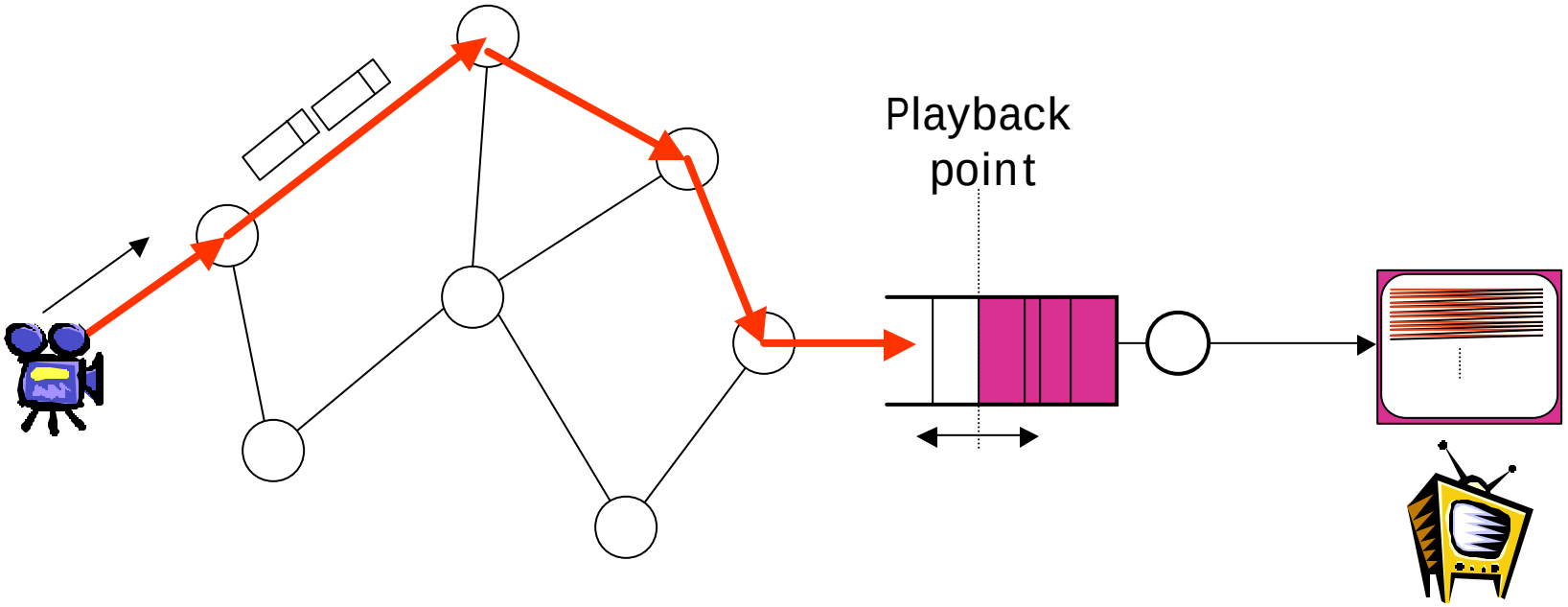
- Play back data upon arrival
  - distorted signal
- Buffer data for a while (playback buffer)
  - extra delay, less distortion
- Tradeoff depends on application (and use)
  - noninteractive: absorb delay, eliminate all distortion
  - interactive: absorb only a little delay, eliminate some distortion

Playback Buffer

Play back data a fixed time interval after it was sent



Playback Point



Nick Mckeown

Adaptation

- Can move playback point as delays vary

- Moving playback point:
  - increases distortion
  - but allows lower delays

Application Taxonomy (Oversimplified and Fanciful)

- Elastic versus “real-time”
  - traditional data apps are elastic
  - streaming media are real-time
- RT intolerant versus RT tolerant
  - intolerant applications need all data
- Tolerant nonadaptive versus tolerant adaptive
  - not clear why any tolerant app couldn’t adapt
- Rate-adaptive versus delay-adaptive (or both)

Key Points

- Some apps don't need to know maximal delay, just need it to be controlled
  - tolerant, delay-adaptive applications will move playback point to reduce delay
  - can absorb occasional outliers

- Some apps need to know maximal delay
  - can't tolerate loss or distortion
  - need to fix playback point and so need a priori knowledge of delay bound
  - bound is typically much worse than actual delays

Two Service Classes

- Controlled Load
  - keep delays under control, but no bound
- Guaranteed Service
  - explicit delay bound

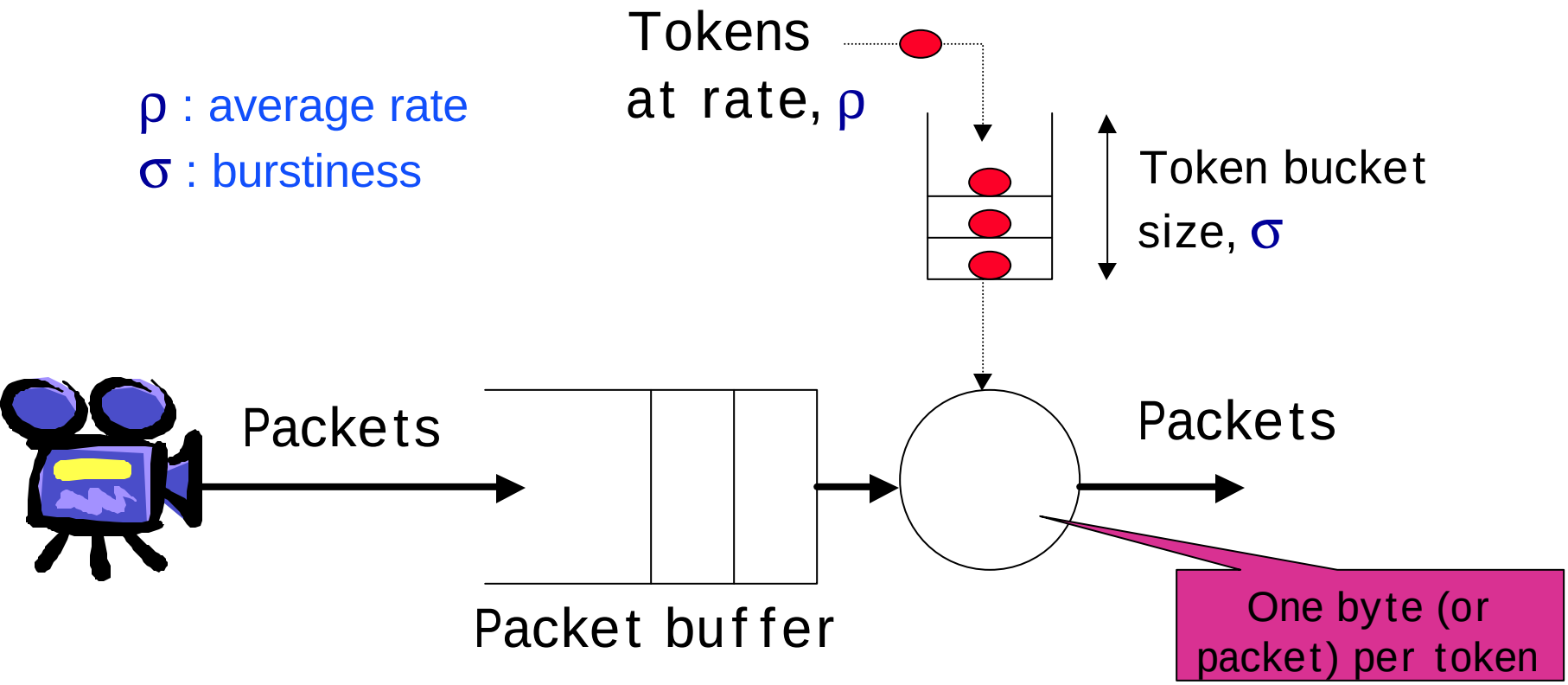
Process

- Flow requests service from network
  - service request specification (RSpec)
    - controlled load: nothing
    - guaranteed: service rate (can calculate delay)
  - traffic specification (TSpec) (next slide)
- Routers decide if they can support request
  - admission control
- If so, traffic is classified and scheduled at routers based on per-flow information

Problem

- How do you describe bursty traffic?
- Network needs some description of traffic
- But video source is bursty (due to coding)
  - can't predict in advance the exact behavior
- Describe "envelope" of traffic: rate and burstiness
- Bits sent between times s and t: $A(s,t) = \sigma + \rho(t-s)$

TSpec: The Token Bucket



Bits sent between times s and t: $A(s,t) = \sigma + \rho(t-s)$

Nick Mckeown

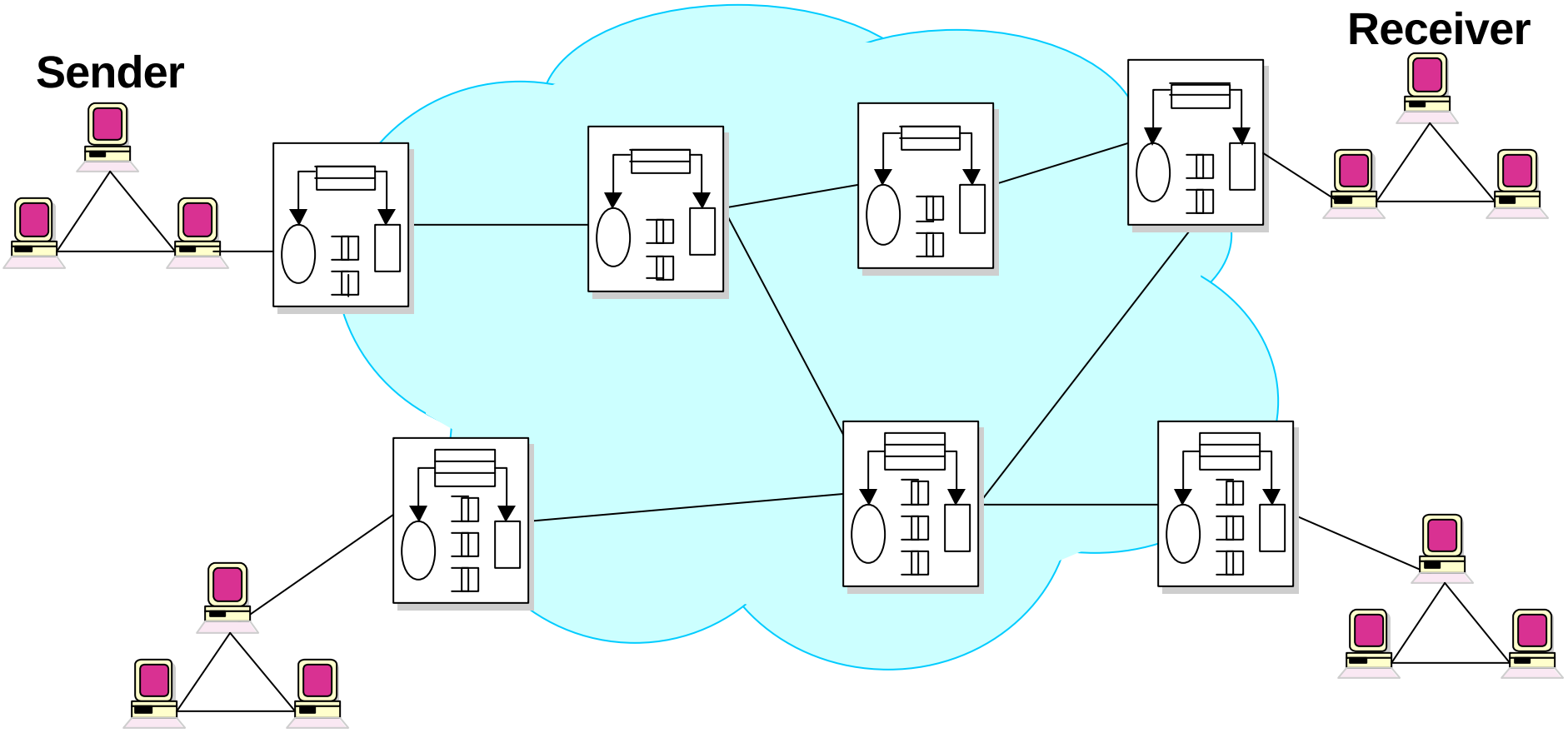
Required Elements

- Reservation Protocol
  - how service request gets from host to network
- Admission control algorithm
  - how network decides if it can accept flow
- Packet scheduling algorithms (next lecture)
  - so routers can deliver service

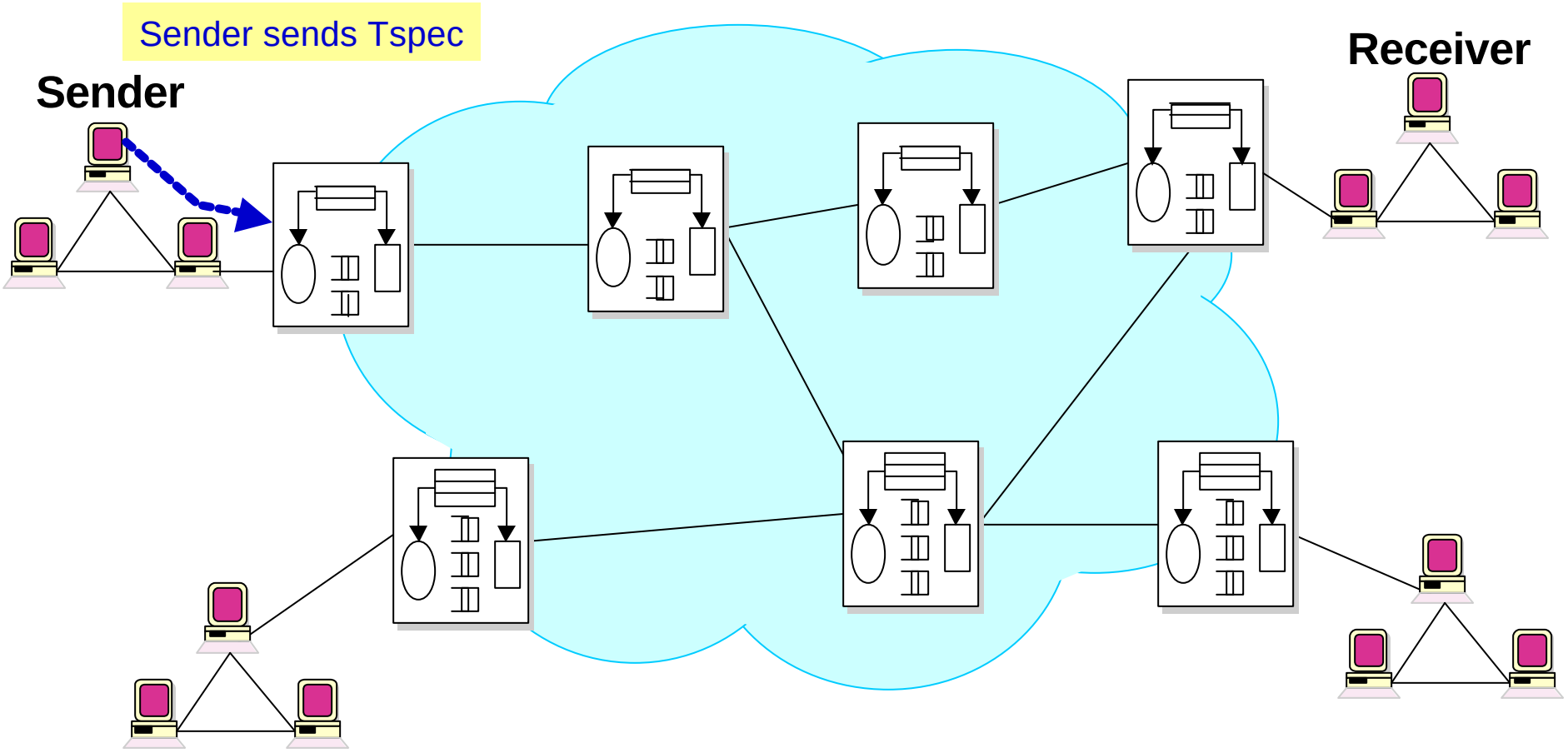
Control Plane versus Data Plane

- Plane as in geometry, not airplane
- Control plane:
  - how information gets to routers
- Data plane:
  - what routers do with that information to data packets

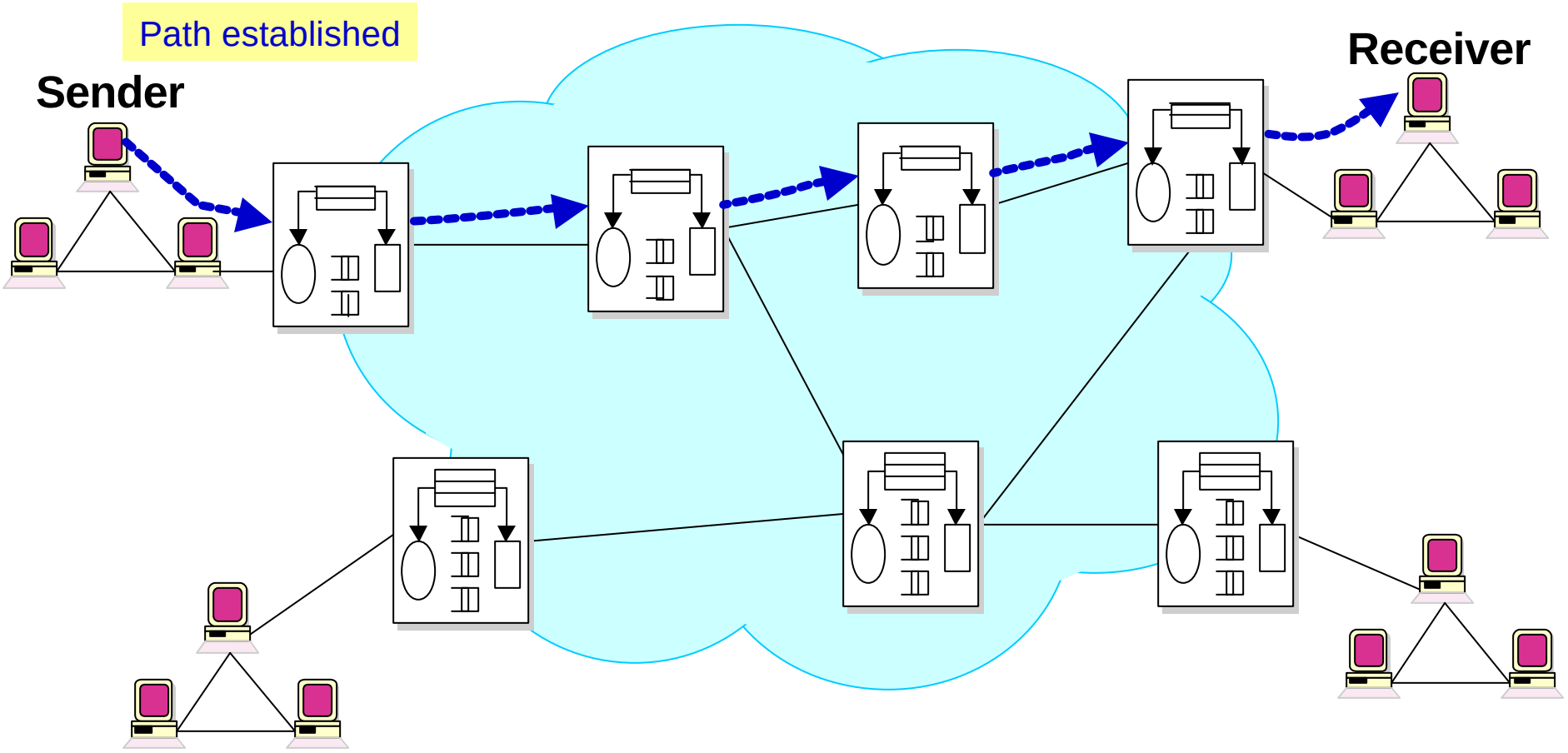
Control Plane: Resource Reservation



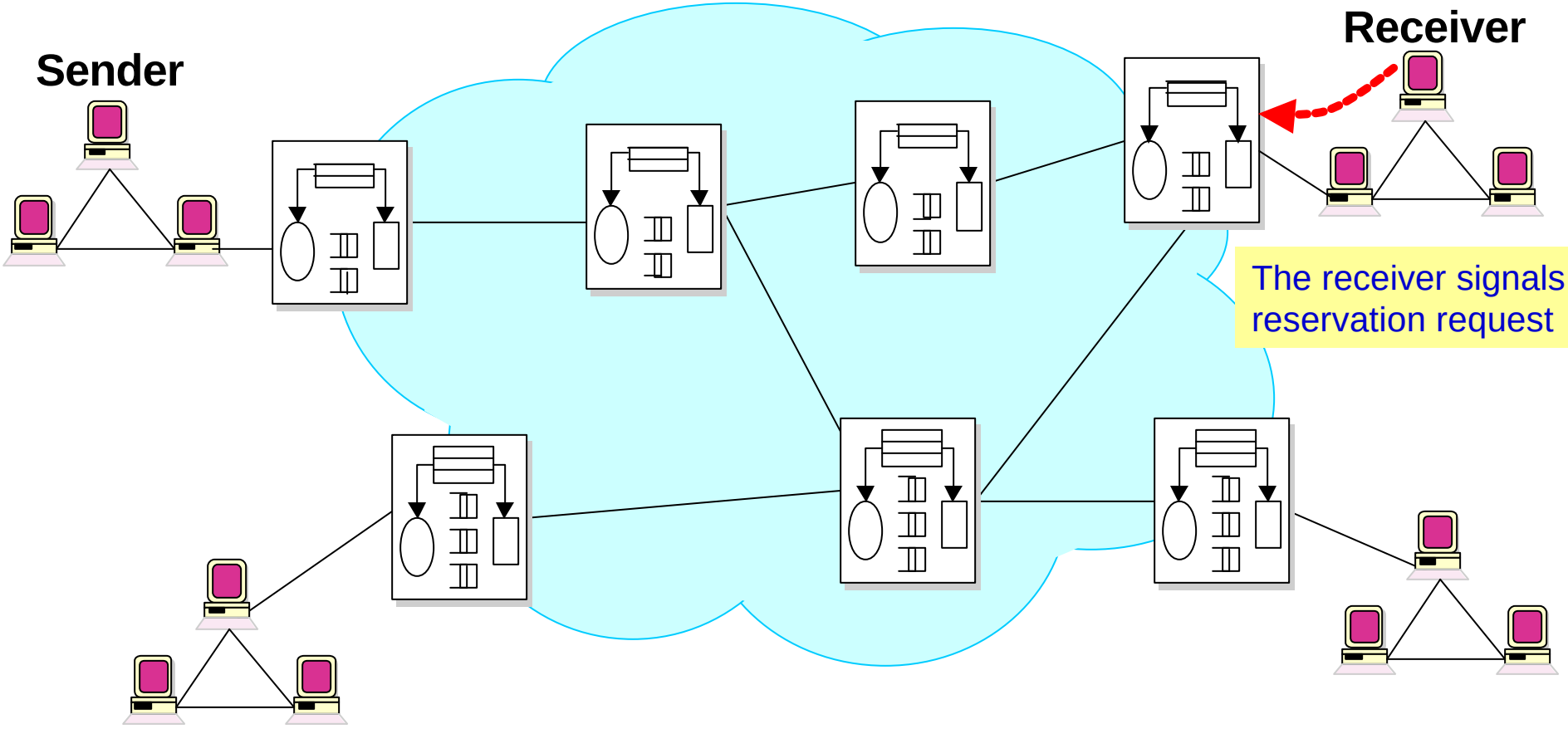
Control Plane: Resource Reservation



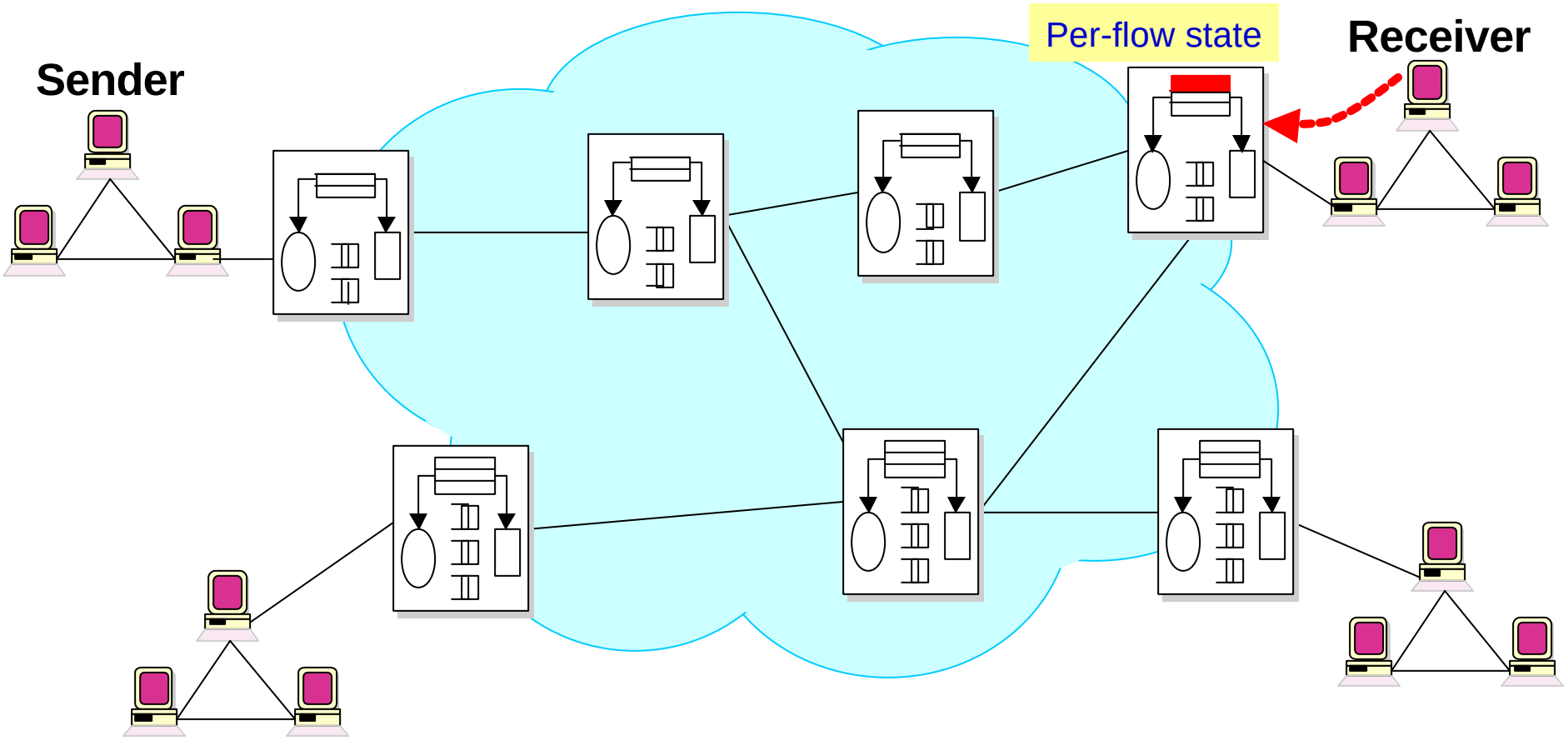
Control Plane: Resource Reservation



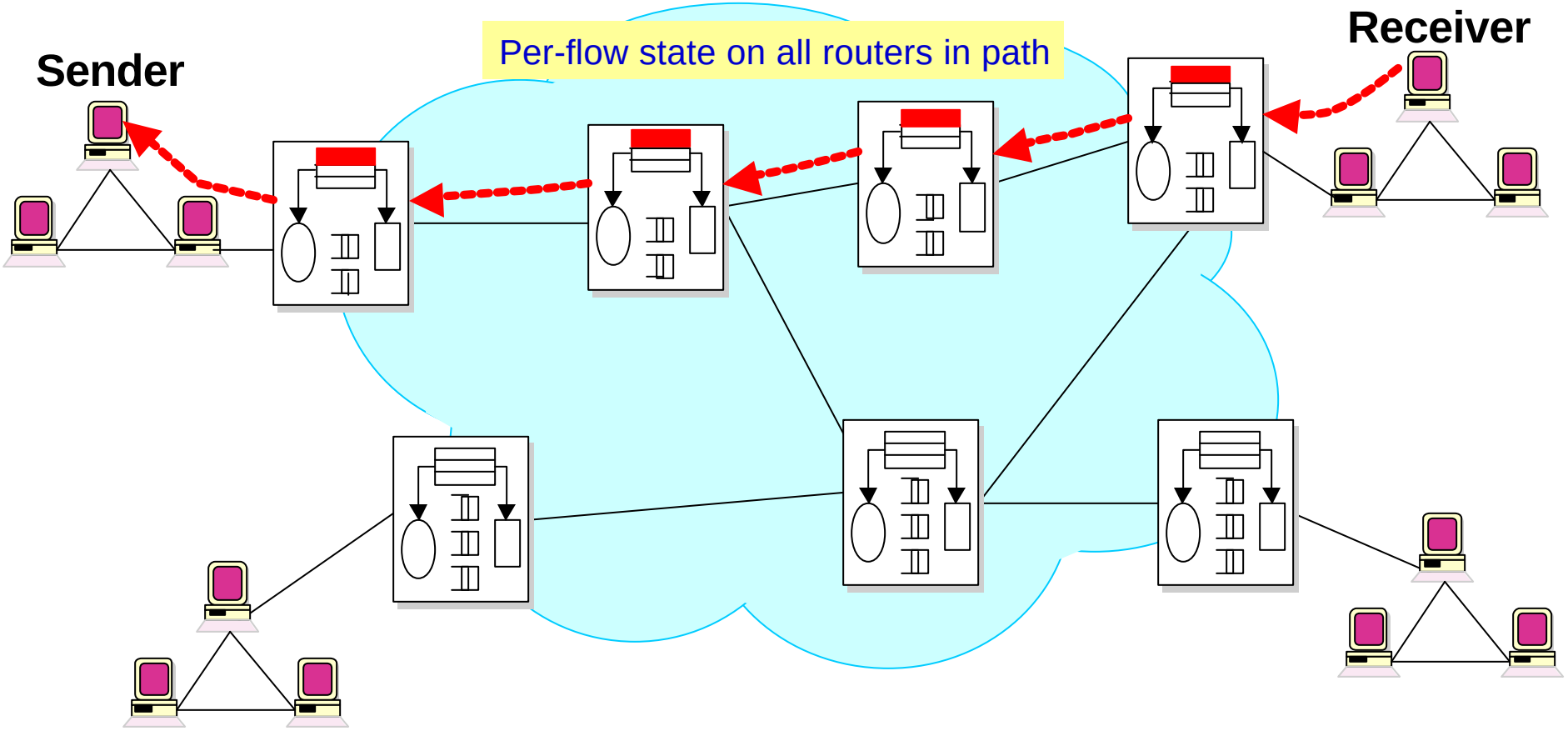
Control Plane: Resource Reservation



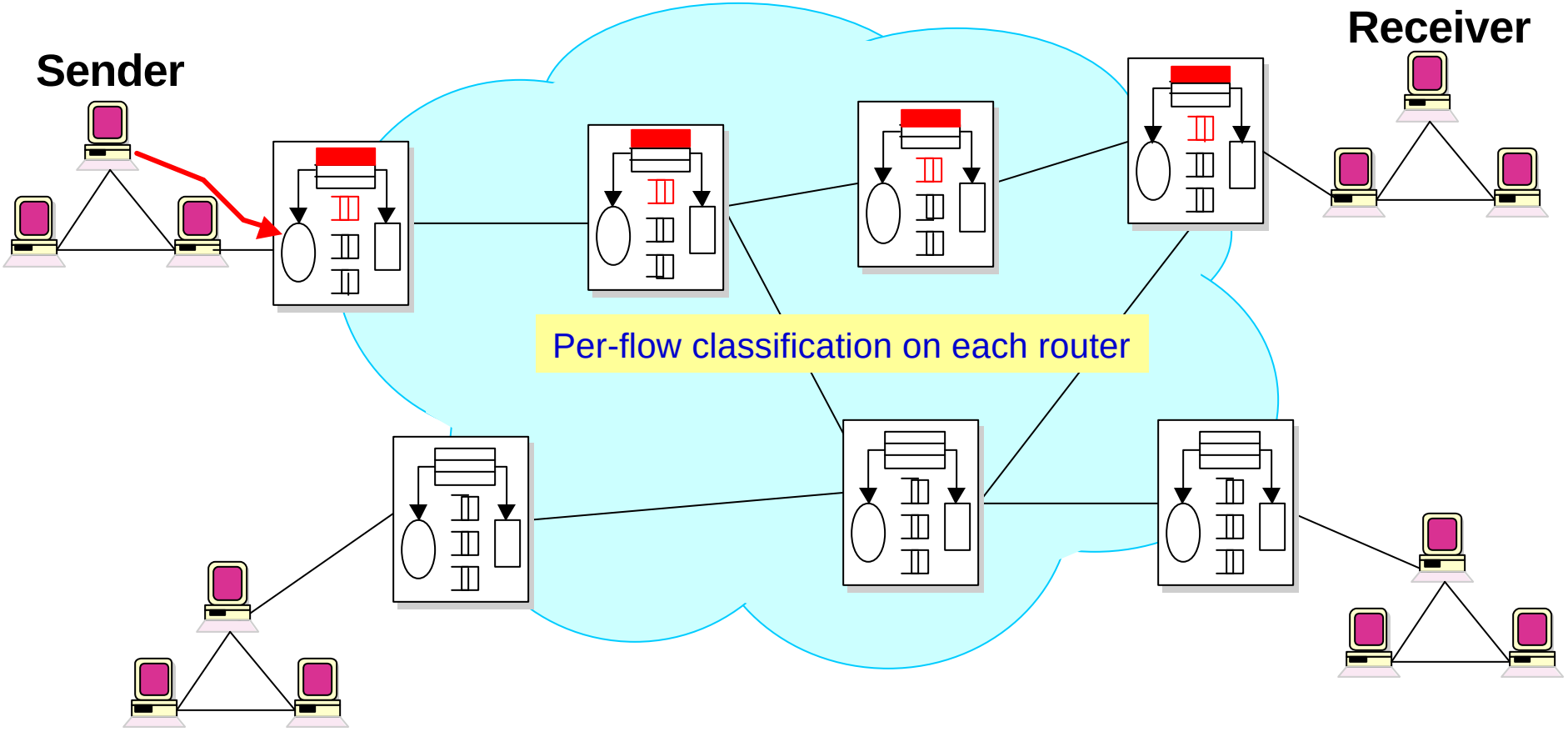
Control Plane: Admission Control



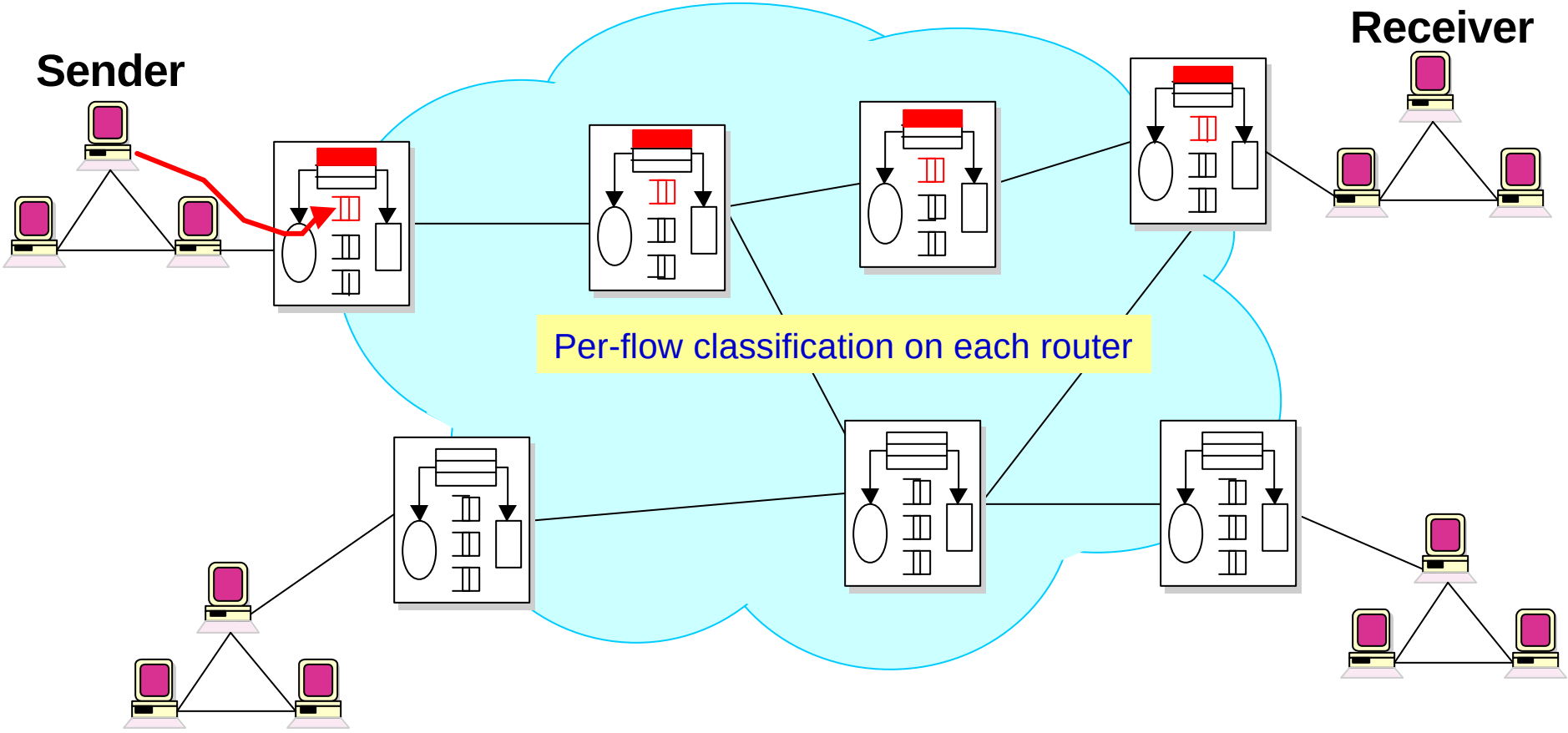
Control Plane: Admission Control



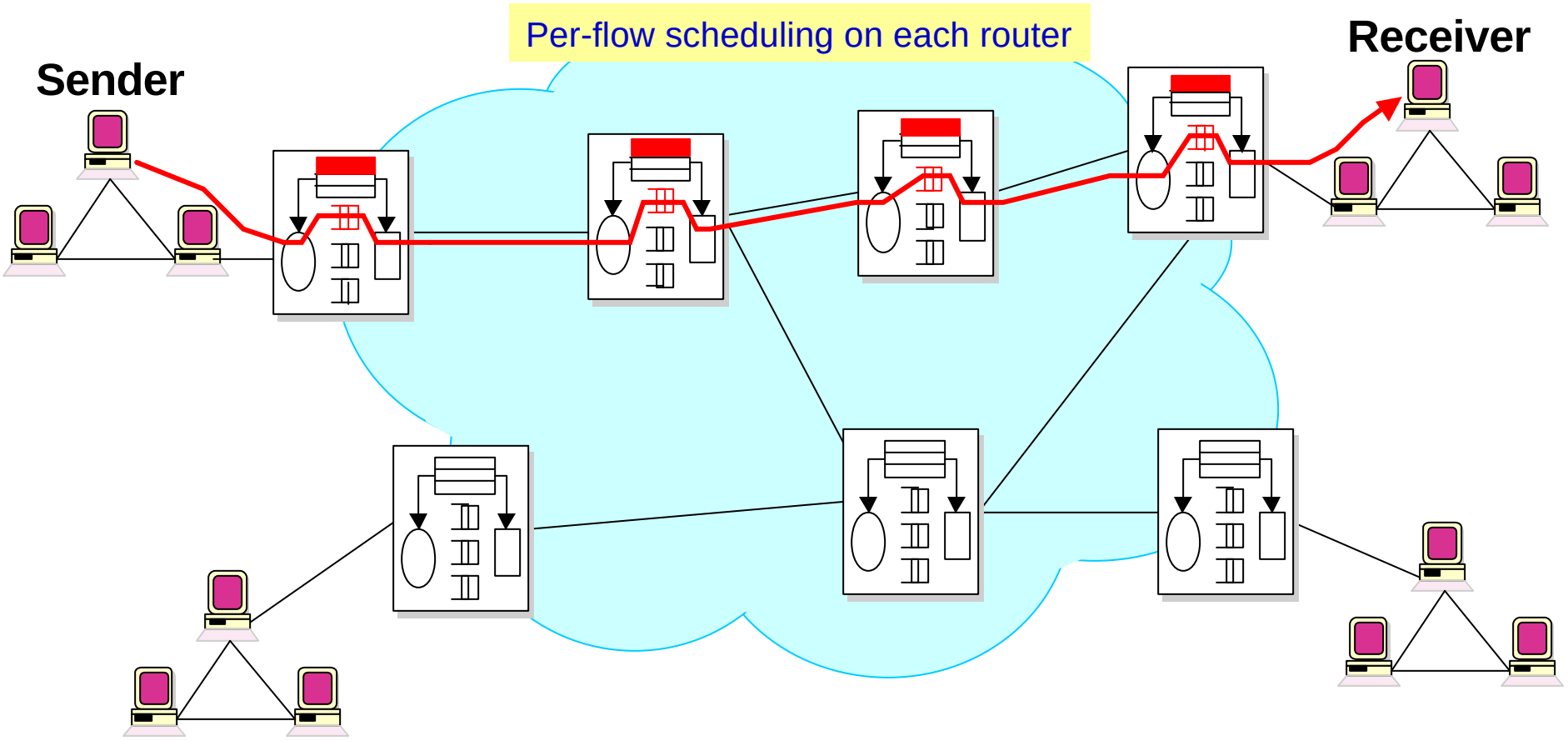
Data Plane



Data Plane



Data Plane



Resource Reservation Protocol: RSVP

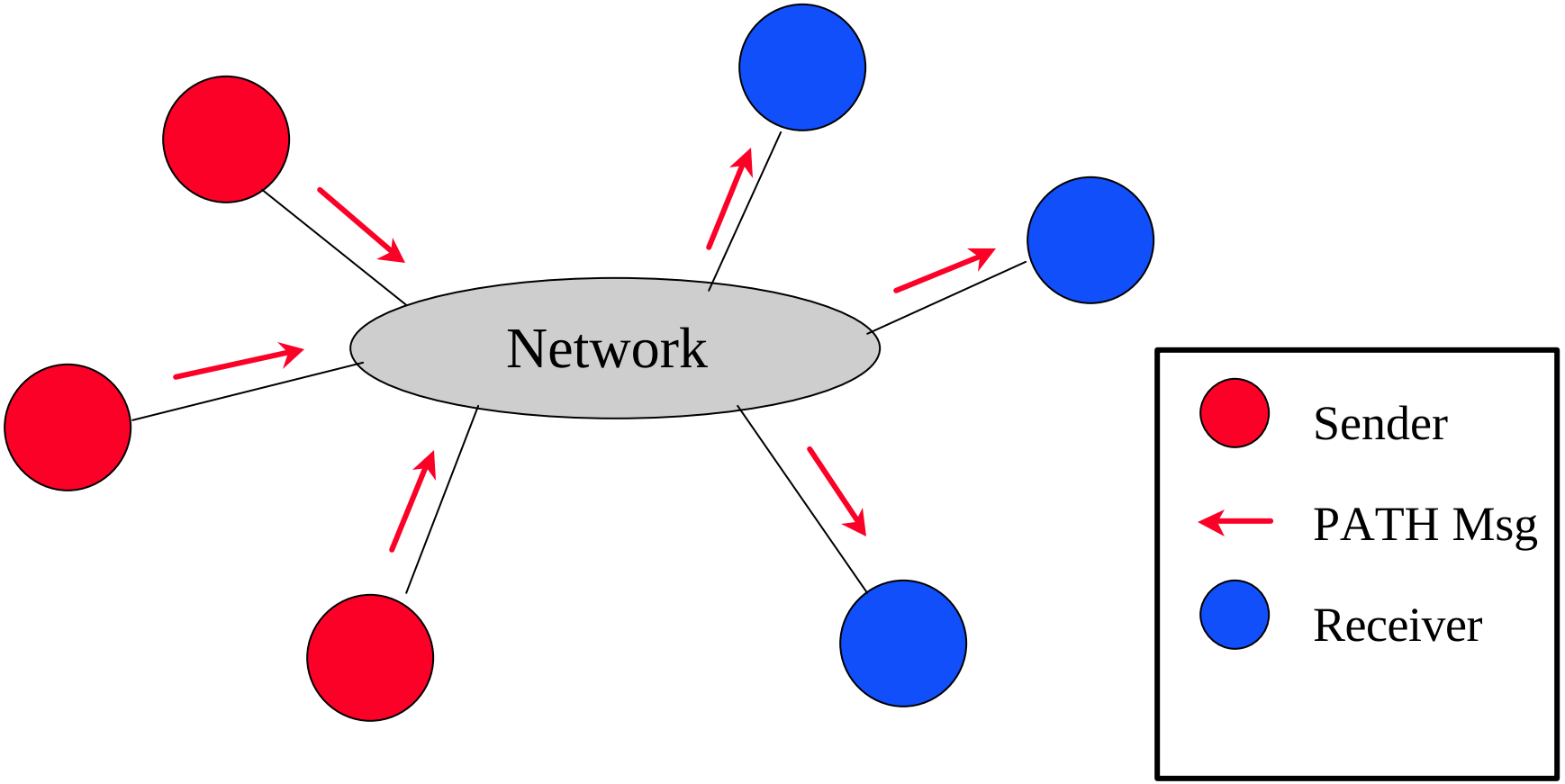
- Establishes end-to-end reservations over a datagram network
- Designed for multicast (which will be covered later in course).

- Sources: send TSpec
- Receivers: respond with RSpec Network
- Network: responds to reservation requests

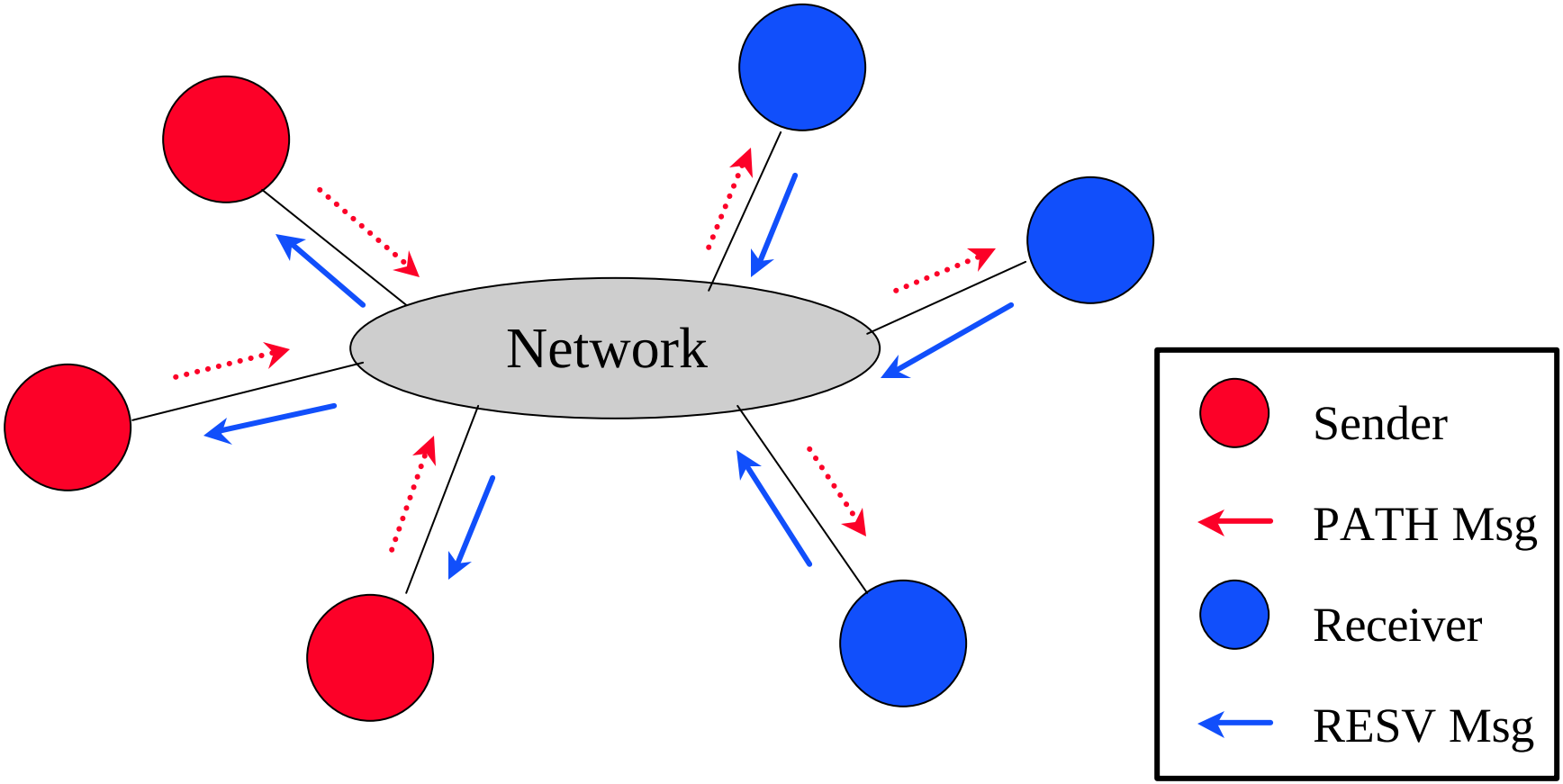
PATH and RESV Messages

- Sender sends PATH messages
  - TSPEC: use token bucket
  - Set up the path state on each router including the address of previous hop (route pinning)
  - Collect path information (for guaranteed service)
- Receiver sends RESV message on the reverse path
  - Specify RSpec and TSpec
  - Sets up the reservation state at each router

The Big Picture



The Big Picture

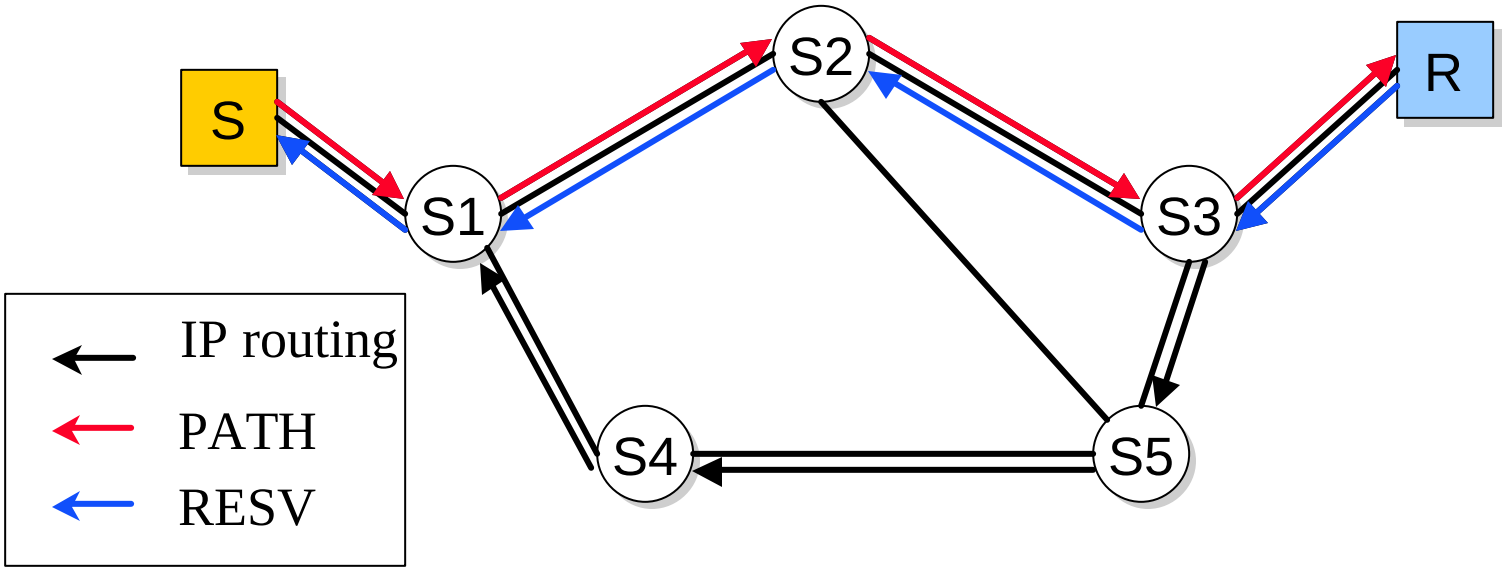


Soft State

- Per session state has a timer associated with it
  - Path state, reservation state
- State deleted when timer expires
- Sender/Receiver periodically refreshes the state, resends PATH/RESV messages, resets timer
- Advantages:
  - No need to clean up dangling state after failure
  - Can tolerate lost signaling packets
  - Easy to adapt to route changes

Route Pinning

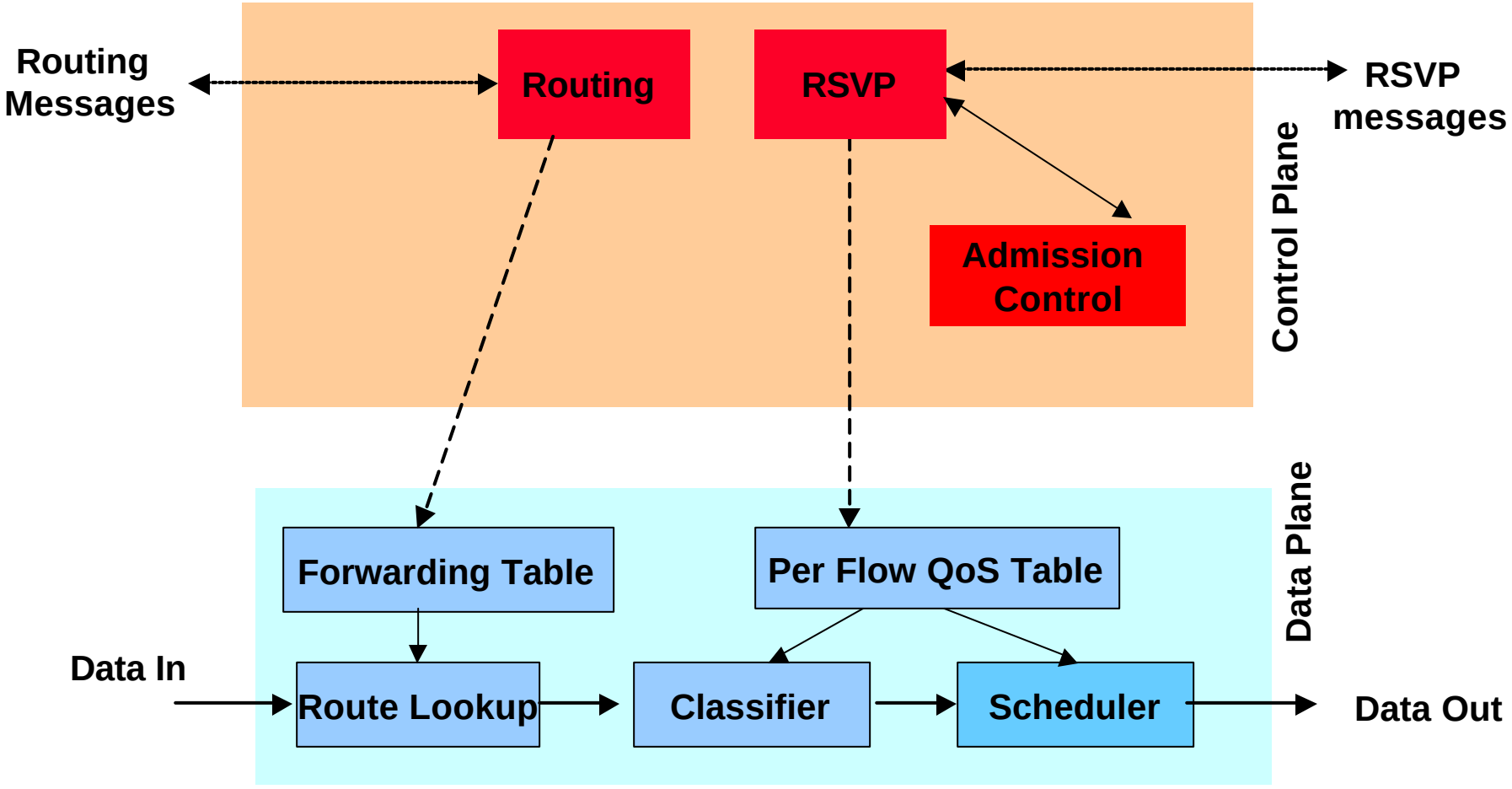
- Problem: asymmetric routes
  - You may reserve resources on R→S3→S5→S4→S1→S, but data travels on S→S1→S2→S3→R !
- Solution: use PATH to remember direct path from S to R, i.e., perform route pinning



Admission Control

- Parameter-based: worst cast analysis
  - guaranteed service
  - low utilization
- Measurement-based: measure current traffic
  - controlled load service
  - higher utilization
- Remember that best-effort service co-exists
  - no need for IntServ traffic to achieve high utilization

IntServ Node Architecture



Advantages of IntServ

- Precise QoS delivered at flow granularities
  - good service, given exactly to who needs it
- Decisions made by hosts
  - who know what they need
  - not by organizations, egress/ingress points, etc.
- Fits multicast and unicast traffic equally well

Disadvantages of IntServ

- Scalability: per-flow state, classification, etc.
  - we goofed, bigtime
  - aggregation/encapsulation techniques can help
  - can overprovision big links, per-flow ok on small links
  - scalability can be fixed, but no second chance
- Economic arrangements:
  - need sophisticated settlements between ISPs
  - right now, settlements are primitive (barter)
- User charging mechanisms: need QoS pricing

What You Need to Know

- Three kinds of QoS approaches
  - Link sharing, DiffServ, IntServ
- Some basic concepts:
  - differentiated dropping versus service priority
  - per-flow QoS (IntServ) versus per-aggregate QoS (DiffServ)
  - Admission control: parameter versus measurement
  - control plane versus data plane
  - controlled load versus guaranteed service
  - codepoints versus explicit signaling
- Various mechanisms:
  - playback points
  - token bucket
  - RSVP PATH/RESV messages

Factors Limiting QoS Deployment

- Prevalence of overprovisioning
  - if all links are only at 40% utilization, why do you need QoS?
  - lore says that inter-ISP links are not overprovisioned
- Primitive inter-ISP financial arrangements
  - QoS requires financial incentives to enforce tradeoffs
  - Current peering arrangements are not able to carry these incentives through in a meaningful way
    - must agree on pricing and service
    - currently agree on neither!
- End-users not used to pricing/performance options

QoS Debates

- Is overprovisioning enough?
  - if so, is this only because access links are slow?
  - what about Korea, Japan, and other countries with fast access links?
  - Disconnect: ISPs overprovision, users get bad service
- Is differentiated services enough?
  - can one really deliver reliable service just using relative priorities?
  - is EF service a viable option?
- It all depends on adaptability of applications