Master Course
Computer Networks
IN2097

Prof. Dr.-Ing. Georg Carle
Christian Grothoff, Ph.D.
Stephan Günther

Chair for Network Architectures and Services
Department of Computer Science
Technische Universität München
http://www.net.in.tum.de

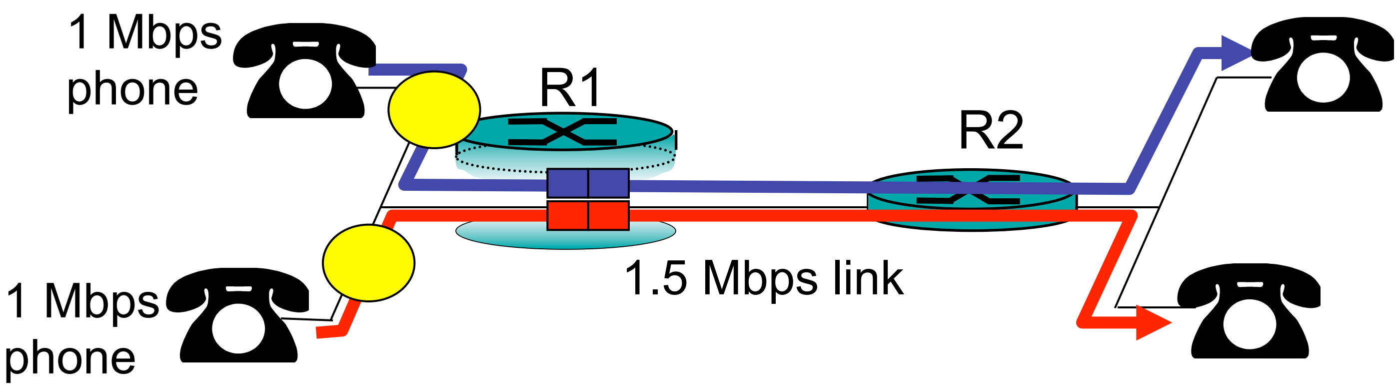
Chapter:
Quality of Service Support

Chapter outline – Quality-of-Service Support

- Providing multiple classes of service
- **Providing QoS guarantees**
- Signalling for QoS

Principles for QOS Guarantees (more)

- *Basic fact of life:* can not support traffic demands beyond link capacity

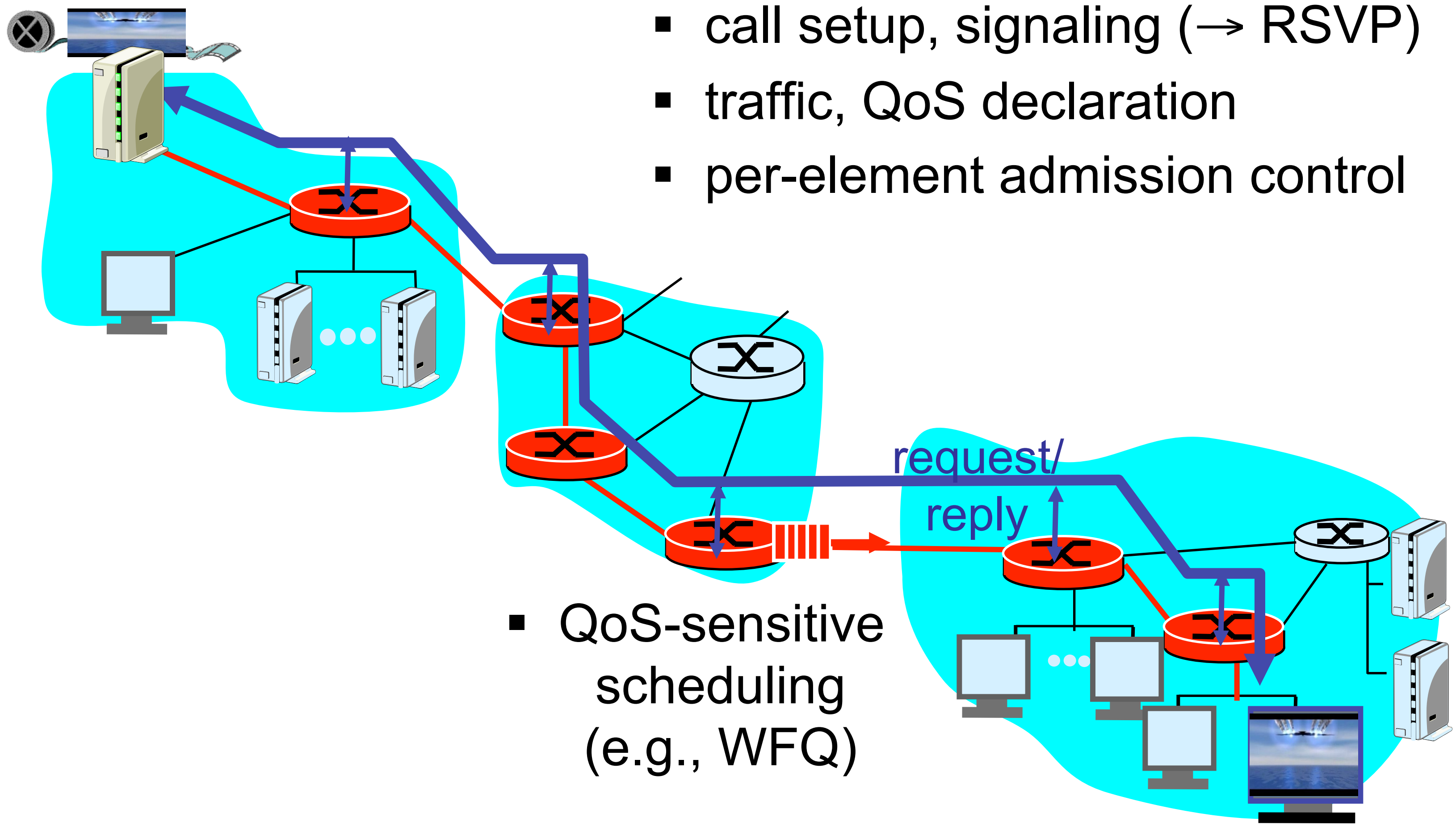


Principle

Call Admission: flow declares its needs, network may block call (e.g., busy signal) if it cannot meet needs

QoS Guarantee Scenario

- Resource reservation
  - call setup, signaling (→ RSVP)
  - traffic, QoS declaration
  - per-element admission control

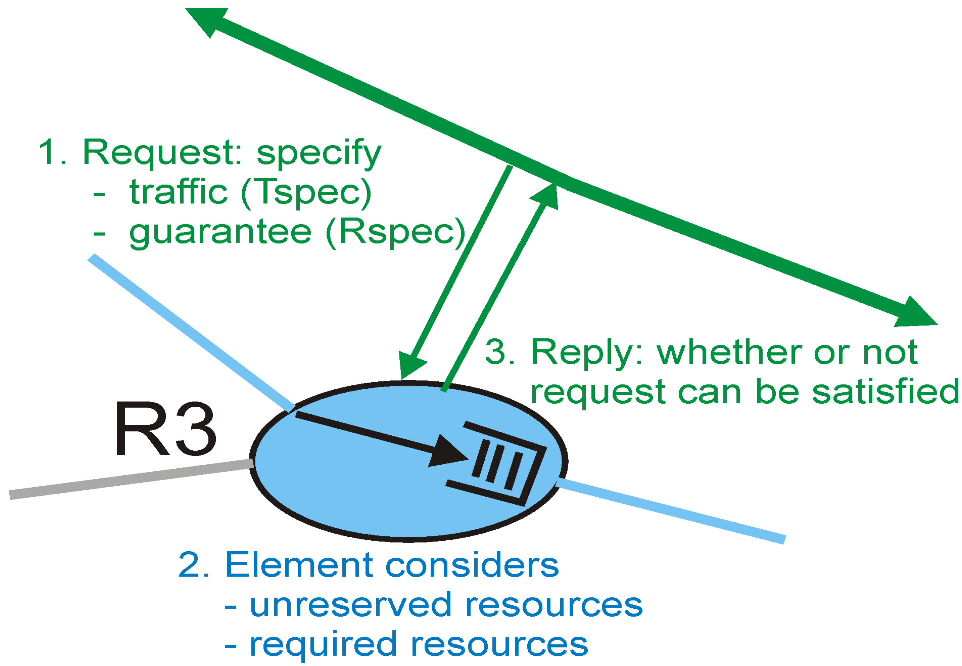


- QoS-sensitive scheduling (e.g., WFQ)

Call Admission

Routers will admit calls based on:

- Flow behavior:
  - T-spec (Traffic specification)
  - R-spec (Reservation specification)
- current resources allocated
  at the router to other calls (flows)



IETF Integrated Services

- Architecture for providing QOS guarantees in IP networks for individual application sessions
- Resource reservation: routers maintain state info (as for VCs) of allocated resources, QoS requests
- Admit/deny new call setup requests

Question: can newly arriving flow be admitted with performance guarantees while not violated QoS guarantees made to already admitted flows?

Call Admission

Arriving session must :

- characterize traffic it will send into network
  - T-spec: defines traffic characteristics
- declare its QoS requirement
  - R-spec: defines the QoS being requested
- signaling protocol: needed to carry T-spec and R-spec to routers (where reservation is required)
  - RSVP

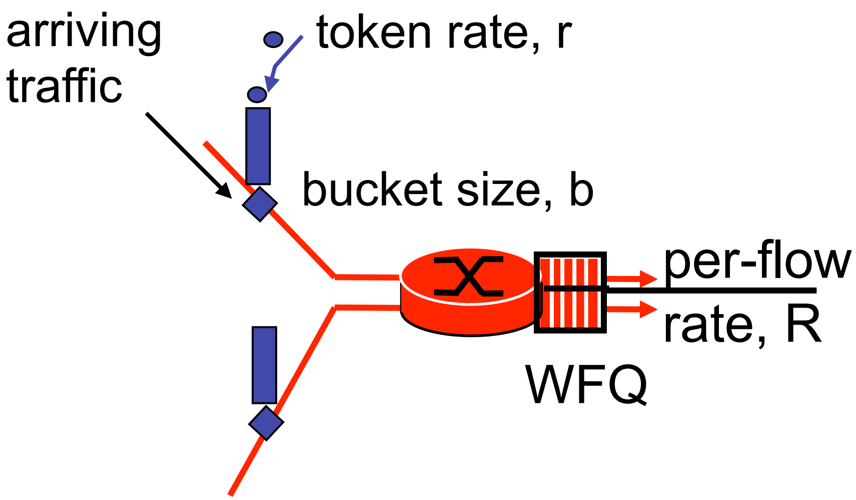
Intserv QoS: Service models [RFC 2211, RFC 2212]

Guaranteed service:

- worst case traffic arrival: leaky-bucket-policed source
- simple (mathematically provable) *bound* on delay [Cruz 1988, Parekh 1992]

Controlled load service:

- "a quality of service closely approximating the QoS that same flow would receive from an unloaded network element."

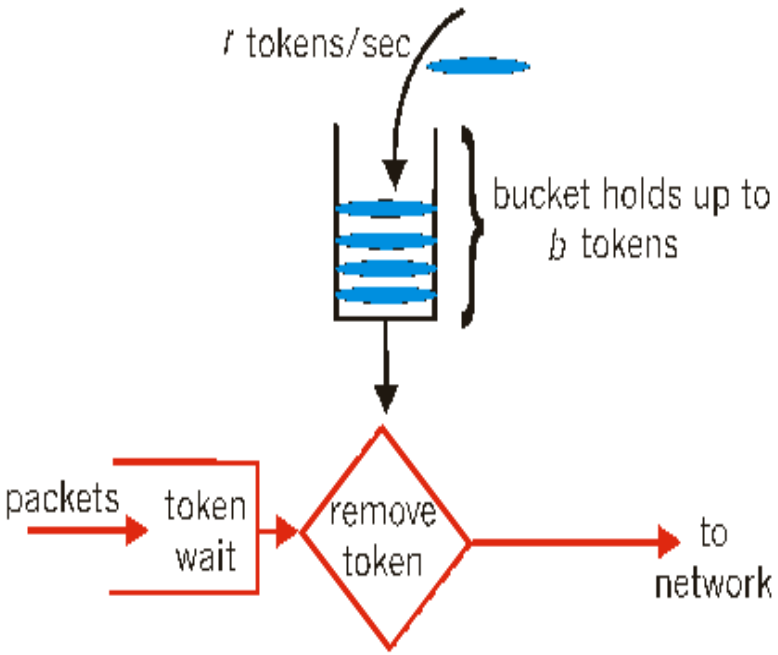


$$D_{max} = b/R$$

Guaranteed Service

- Leaky Bucket parameters (r,b)
  - r: Token bucket rate
  - b: Token bucket size
- T-spec:
  - p: Peak data rate
  - m: Minimum policed unit
  - M: Maximum packet size
- R-spec:
  - R: Reserved rate (R>>r)
  - S: slack term

    (Signify the difference between the desired delay and the delay obtained by using reservation level R)
- Simple Delay bound : b/R
  - Request guarantee transmission rate is R
  - Amount of traffic generated over interval t is bound by rt + b
  - The maximum queueing delay experienced by any packet is bound by b/R



Chapter outline – Quality-of-Service Support

- Providing multiple classes of service
- Providing QoS guarantees
- **Signalling for QoS**

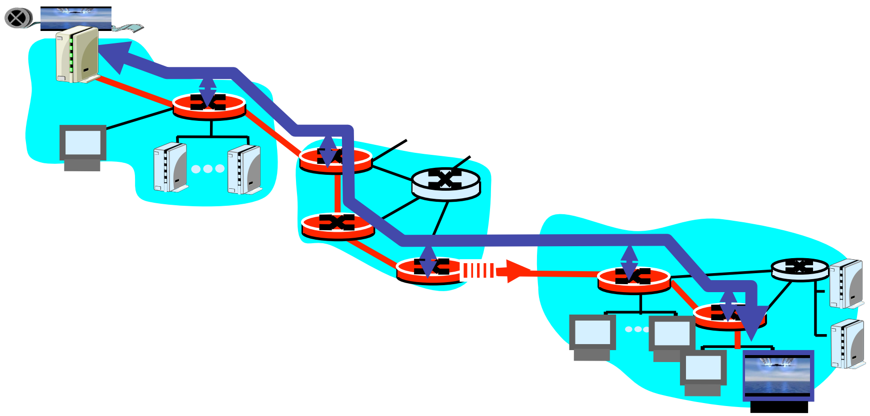
Signaling in the Internet

connectionless (stateless) forwarding by IP routers + best effort service = no network signaling protocols in initial IP design

- New requirement: reserve resources along end-to-end path (end system, routers) for QoS for multimedia applications
- RSVP: Resource Reservation Protocol [RFC 2205]
  - " … allow users to communicate requirements to network in robust and efficient way." i.e., signaling !
- earlier Internet Signaling protocol: ST-II [RFC 1819]

RSVP Design Goals

1. accommodate heterogeneous receivers (different bandwidth along paths)
2. accommodate different applications with different resource requirements
3. support multicast, adaptat to multicast group membership
4. leverage existing multicast/unicast routing, with adaptation to changes in underlying unicast, multicast routes
5. control protocol overhead to grow (at worst) linear in # receivers
6. modular design for heterogeneous underlying technologies



RSVP: does not…

- specify *how* resources are to be reserved
  - rather: a mechanism for *communicating needs*
- determine routes packets will take
  - that's the job of routing protocols
  - signaling decoupled from routing
- interact with forwarding of packets
  - separation of control (signaling) and data (forwarding) planes

RSVP: overview of operation

- senders, receiver join a multicast group
  - done outside of RSVP
  - senders need not join group
- sender-to-network signaling
  - *path message:* make sender presence known to routers
  - path teardown: delete sender's path state from routers
- receiver-to-network signaling
  - *reservation message:* reserve resources along path
  - reservation teardown: remove receiver reservations
- network-to-end-system signaling
  - path error
  - reservation error

RSVP Messages

Two types of messages

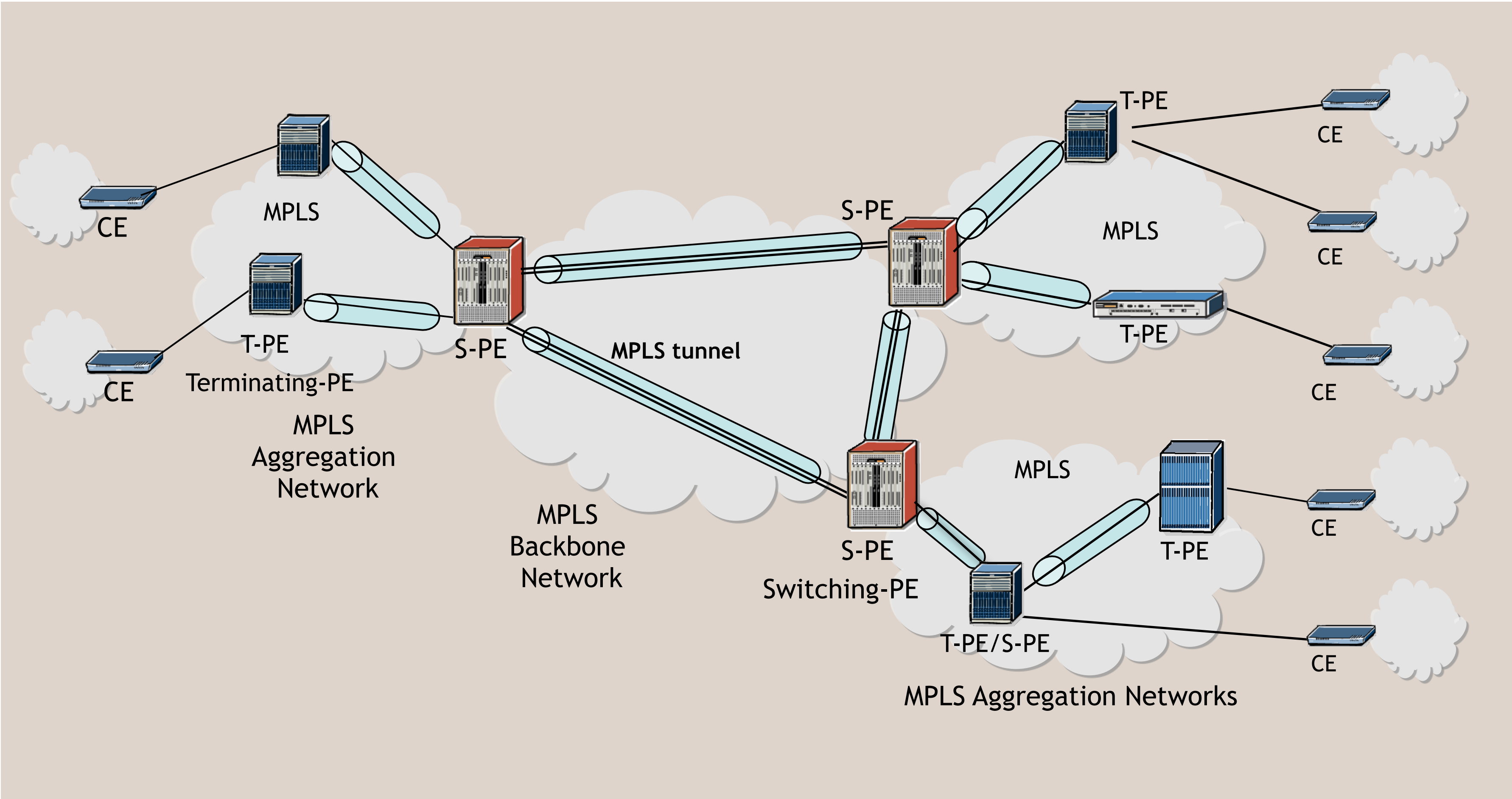
- Path messages (*path*)
  - sent from sender along data path and stores the *path state* in each node along the path
  - *path state* includes IP address of previous node, and data objects:
    - *sender template* - describes format of sender data
    - *sender T-spec* - describes traffic characteristics of data flow
    - *adspec* - carries advertising data (c.f. RFC 2210)
- Reservation messages (*resv*)
  - sent from the receiver to the sender host along reverse data path
  - At each node IP destination address of *resv* message changes to address of the next node on the reverse path, and IP source address to address of previous node address on reverse path
  - includes the *flowspec* data object that identifies needed resources, with service class, reservation specification, and flow description

RSVP RFCs

- RFC 2205: The version 1 functional specification admission (traffic) control that is based "only" on resource availability.
- RFC 2210: use of RSVP with controlled-load RFC 2211 and guaranteed RFC 2212 QoS control services.
- RFC 2211: specifies the network element behavior required to deliver Controlled-Load services.
- RFC 2212: specifies the network element behavior required to deliver guaranteed QoS services.
- RFC 2750: extension for supporting generic policy based admission control in RSVP.
- RFC 3209: RSVP-TE: Extensions to RSVP for LSP Tunnels“
- RFC 3473: Generalized Multi-Protocol Label Switching (GMPLS) Signaling Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Extensions.
- RFC 3936: Procedures for Modifying the Resource reSerVation Protocol (RSVP)
- RFC 4495: A Resource Reservation Protocol (RSVP) Extension for the Reduction of Bandwidth of a Reservation Flow.
- RFC 455: Node-ID Based Resource Reservation Protocol (RSVP) Hello: A Clarification Statement.

MPLS Signalling

MPLS-based VPN



MPLS Signalling

- Need of signalling in MPLS networks
  - All LSRs of (unidirectional) Label Switched Path (LSP) must be informed about path, initial label value, and possible label swapping
  - Downstream LSR needs mechanism to inform upstream LSR of label to use in outgoing MPLS packets
- Alternative MPLS signalling protocols
  - Label Distribution Protocol (LDP)
    - sets up LSPs hop-by-hop
    - depends on IGP to determine path of LSP
  - Resource Reservation Protocol with Traffic Engineering Extensions (RSVP-TE)
    - sets up LSP end-to-end (ingress-to-egress)
    - can set up paths independently of IGP optimal path
      ⇨supports Traffic Engineering

RSVP-TE

- RSVP-TE
  - uses Path messages and Resv messages
  - path message sent from ingress to egress
    - requests LSP setup hop-by-hop along path to egress, checking availability of needed resources
  - egress router sends a Resv message back to ingress
  - resources that can be reserved
    - bandwidth reserved for LSP
    - functions, such as Fast Reroute (FRR)
    - capabilities
      - ability of LSP to take resources from another LSPs
      - ability to resist having resources taken away
  - Explicit Route Object (ERO)
    - list of LSRs, specified by IP addresses, to be traversed

MPLS Fast Restauration

- RFC 3469 (informational)
  - Framework for Multi-Protocol Label Switching (MPLS)-based Recovery
  - ability to reroute traffic over precomputed failover path
- RFC 4090 (proposed standard)
  - Fast Reroute Extensions to RSVP-TE for LSP Tunnels
  - RSVP-TE extensions to establish backup label- switched path (LSP) tunnels for local repair of LSP tunnels
  - enable re-direction of traffic onto backup LSP tunnels in 10s of milliseconds in the event of a failure
  - one-to-one backup method
    - creates detour LSPs for each protected LSP at each potential point of local repair
  - The facility backup method
    - creates bypass tunnel by MPLS label stacking, to protect a set of LSPs with similar backup constraints

Maintaining network state

Maintaining network state

> state: information *stored* in network nodes by network protocols

- updated when network "conditions" change
- stored in multiple nodes
- often associated with end-system generated call or session
- examples:
  - ATM switches maintain lists of VCs: bandwidth allocations, VCI/VPI input-output mappings
  - RSVP routers maintain lists of upstream sender IDs, downstream receiver reservations
  - TCP: Sequence numbers, timer values, RTT estimates

Hard-state

- state *installed* by receiver on receipt of *setup message* from sender
- state *removed* by receiver on receipt of *teardown message* from sender
- *default assumption:* state valid unless told otherwise
  - in practice: failsafe-mechanisms (to remove orphaned state) in case of sender failure e.g., receiver-to-sender "heartbeat": is this state still valid?
- examples:
  - Q.2931 (ATM Signaling)
  - ST-II (Internet hard-state signaling protocol - outdated)
  - TCP

Soft-state

- state *installed* by receiver on receipt of setup (trigger) message from sender (typically, an endpoint)
  - sender also sends periodic *refresh* message: indicating receiver should continue to maintain state
- state *removed* by receiver via timeout, in absence of refresh message from sender
- default assumption: state becomes invalid unless refreshed
  - in practice: explicit state removal (*teardown*) messages also used
- examples:
  - RSVP, RTP/RTCP, IGMP

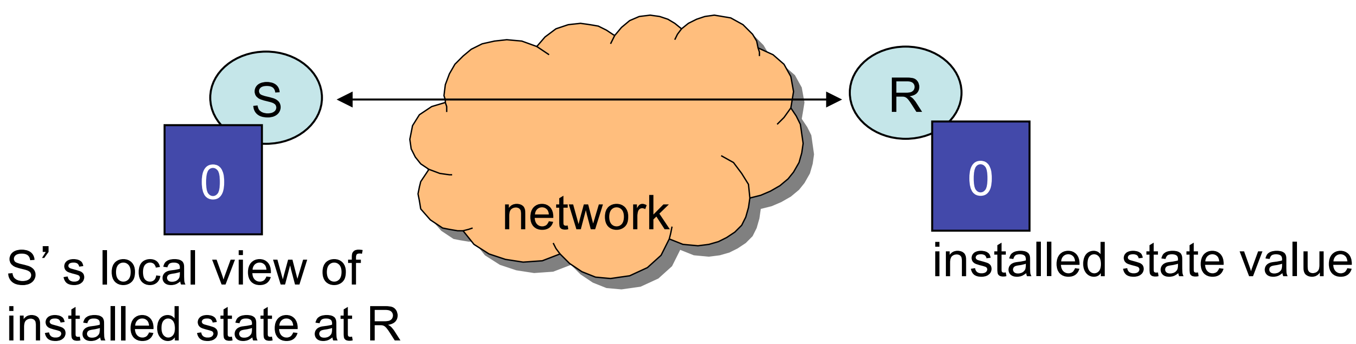
State: senders, receivers

- sender: network node that *(re)generates* signaling (control) messages to install, keep-alive, remove state from other nodes

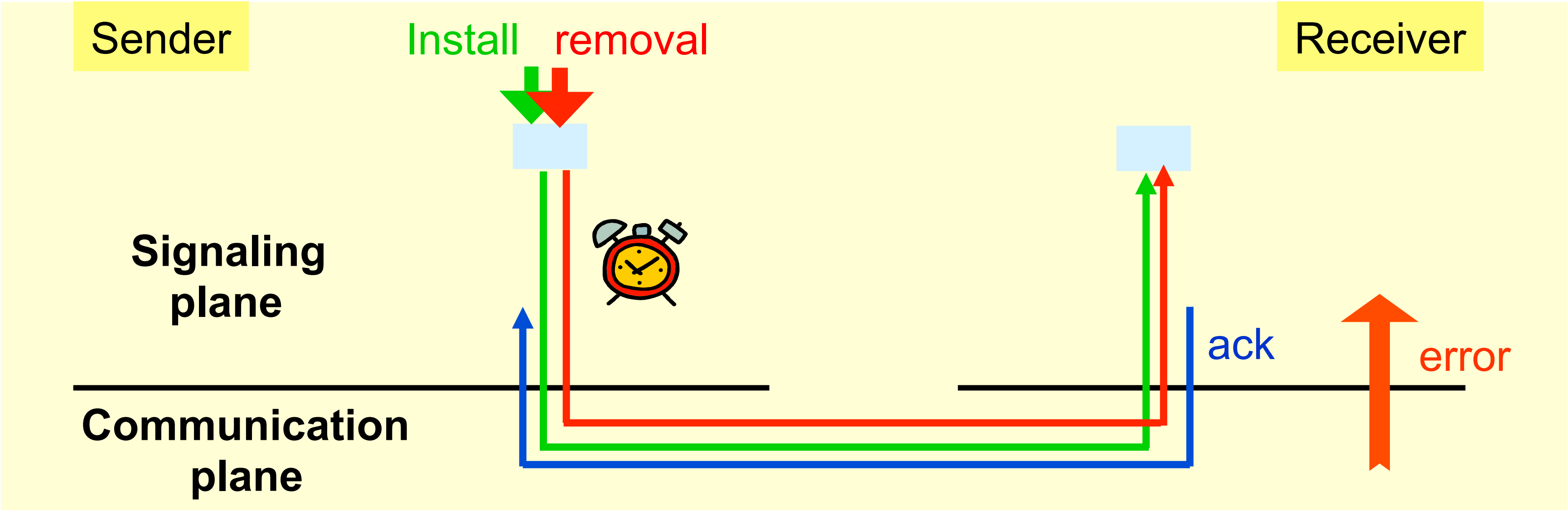
- receiver: node that creates, maintains, removes state based on signaling messages *received* from sender

Let's build a signaling protocol

- *S:* state *S*ender (state installer)
- *R:* state *R*eceiver (state holder)
- desired functionality:
  - S: set values in R to 1 when state "installed", set to 0 when state "not installed"
  - if other side is down, state is not installed (0)
  - initial condition: state not installed

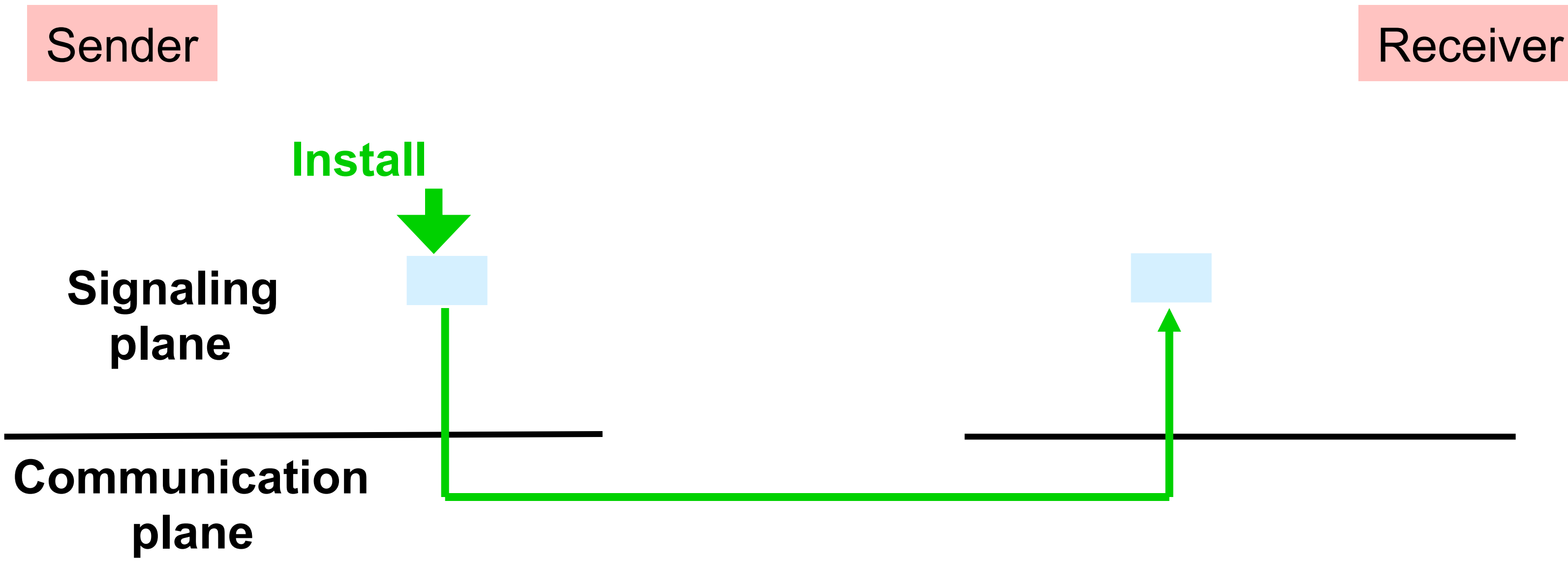


Hard-state signaling



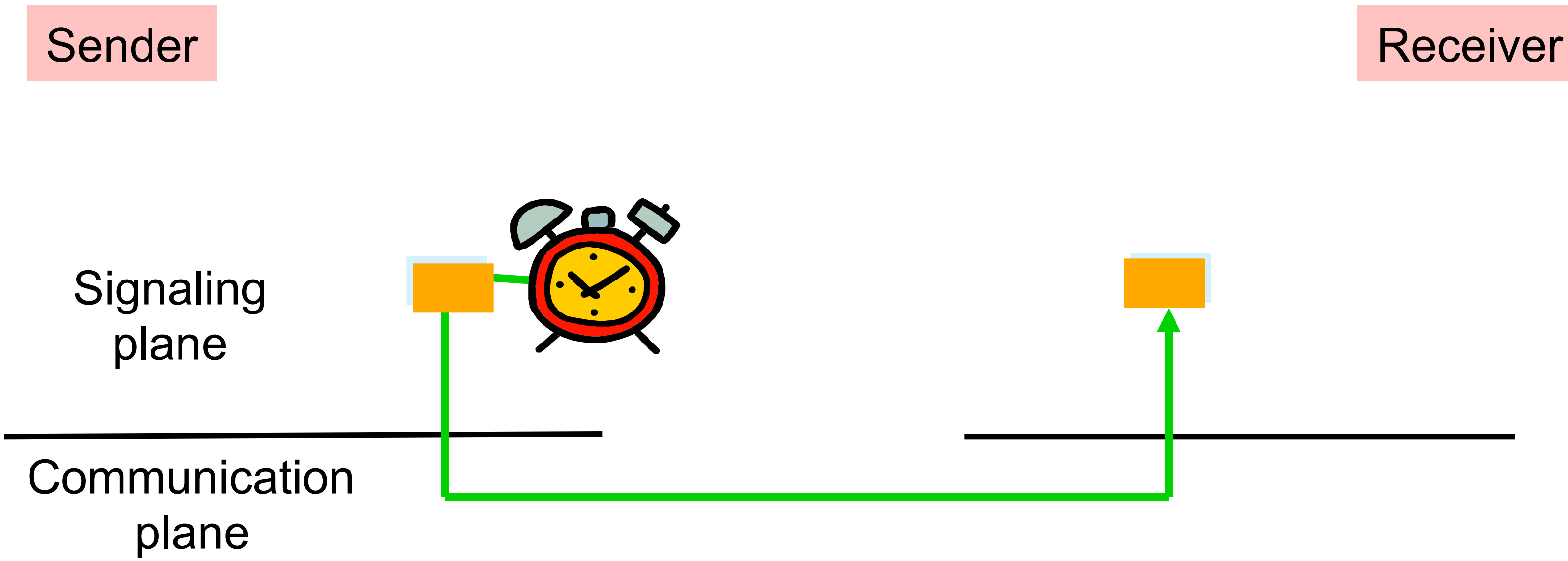
- reliable signaling
- state removal by request
- requires additional error handling
  - e.g., sender failure

Soft-state signaling



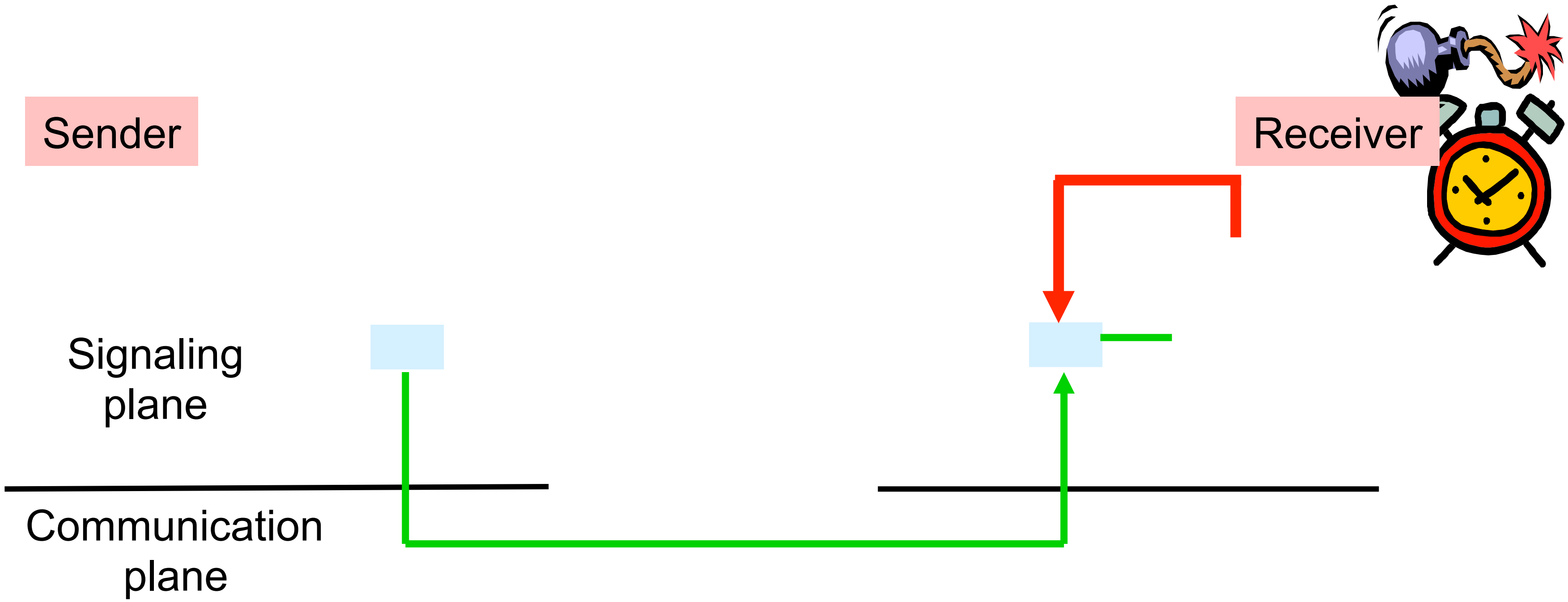
- best effort signaling

Soft-state signaling



- best effort signaling
- refresh timer, periodic refresh

Soft-state signaling



- best effort signaling
- refresh timer, periodic refresh
- state time-out timer, state removal only by time-out

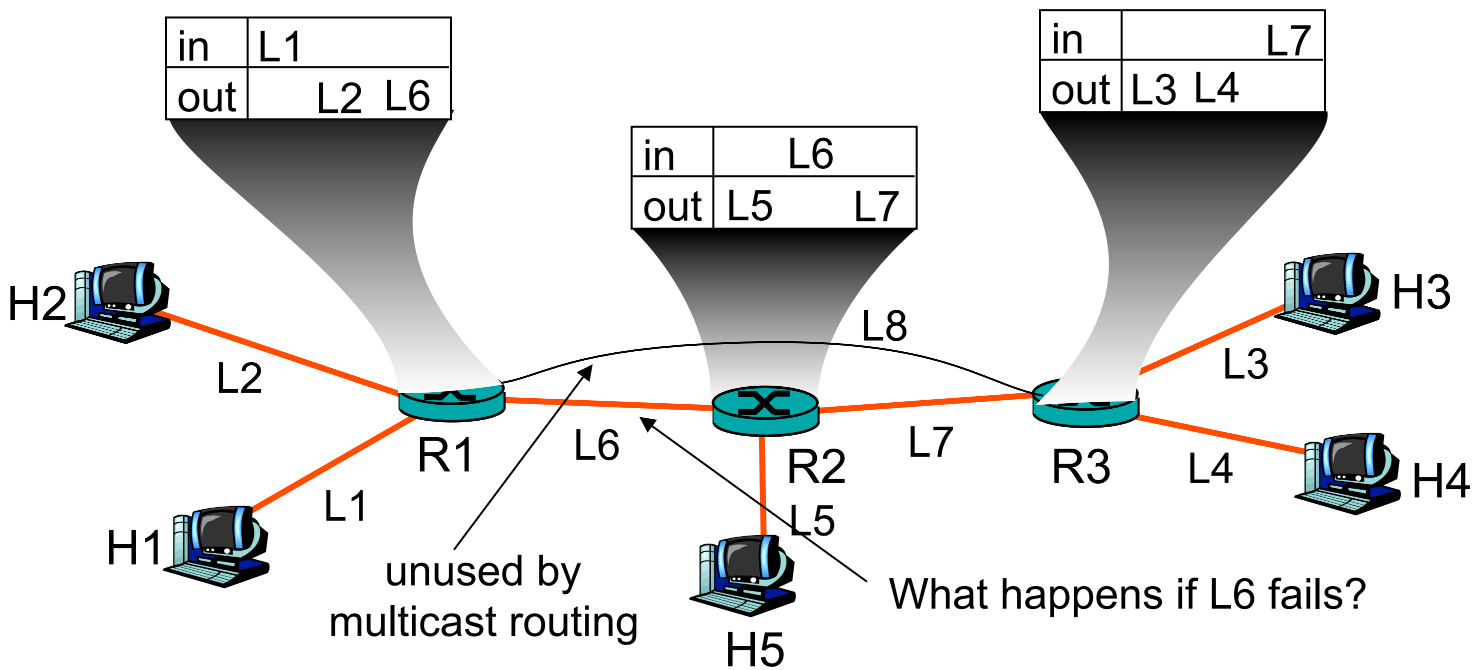
Soft-state: claims

- "Systems built on soft-state are robust" [Raman 99]
- "Soft-state protocols provide .. greater robustness to changes in the underlying network conditions…" [Sharma 97]
- "obviates the need for complex error handling software" [Balakrishnan 99]

What does this mean?

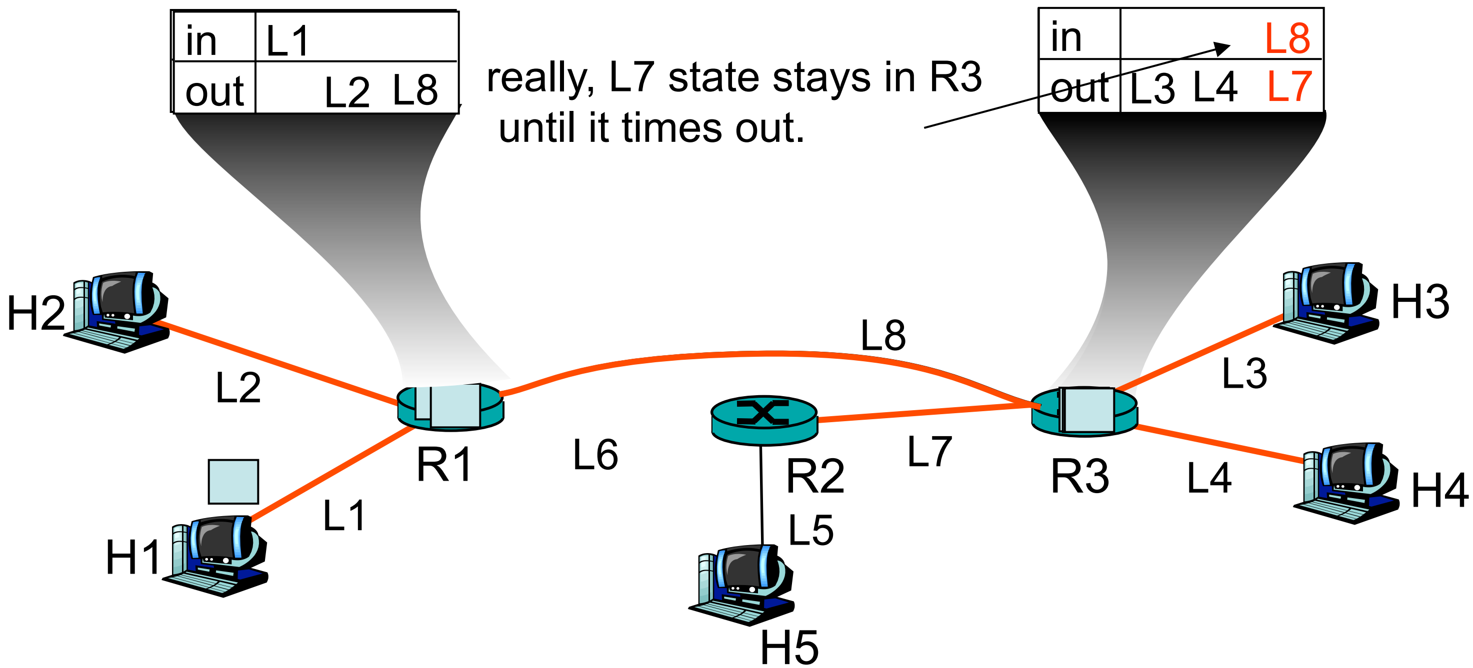
Soft-state: "easy" handling of changes

- Periodic refresh: if network "conditions" change, refresh will re-establish state under new conditions
- example: RSVP/routing interaction: if routes change (nodes fail) RSVP PATH refresh will *re-establish* state along new path



Soft-state: “easy” handling of changes

- L6 goes down, multicast routing reconfigures but…
- H1 data no longer reaches H3, H4, H5 (no sender or receiver state for L8)
- H1 refreshes PATH, establishes *new* state for L8 in R1, R3
- H4 refreshes RESV, propagates upstream to H1, establishes new receiver state for H4 in R1, R3



Soft-state: "easy" handling of changes

- "recovery" performed transparently to end-system by normal refresh procedures
- no need for network to signal failure/change to end system, or end system to respond to specific error
- less signaling (volume, types of messages) than hard-state from network to end-system but…
- more signaling (volume) than hard-state from end-system to network for refreshes

Soft-state: refreshes

- refresh messages serve many purposes:
  - trigger: first time state-installation
  - refresh: refresh state known to exist ("I am still here")
  - <lack of refresh>: remove state ("I am gone")
- challenge: all refresh messages unreliable
  - problem: what happens if first PATH message gets lost?
    - copy of PATH message only sent after refresh interval
  - would like triggers to result in state-installation a.s.a.p.
  - enhancement: add receiver-to-sender refresh_ACK for triggers
  - sender initiates retransmission if no refresh_ACK is received after short timeout
  - e.g., see paper "Staged Refresh Timers for RSVP" by Ping Pan and Henning Schulzrinne
  - approach also applicable to other soft-state protocols

Signaling Spectrum

periodic refresh

Soft-state (SS)

SS + explicit removal
IGMPv2/v3

SS + reliable trigger
RSVP new version

SS + reliable trigger/removal
ST-II

Hard-state

- best effort periodic state installation/refresh
- state removal by time out
- RSVP, IGMPv1

- reliable signaling
- explicit state removal
- requires additional mechanism to remove orphan state
- Q2931b