Design Principles / Protocol Functions

Goals:

- Identify, study common architectural components, protocol mechanisms, approaches do we find in network architectures?
- *Synthesis:* Big picture

Principles / protocol functions:

- Signaling
  - Protocols
  - Separation of data, control
  - Hard state versus soft state
- Randomization
- Indirection
- Network virtualization: overlays
- Multiplexing
- Design for scale

1

1: Separation of Control and Data

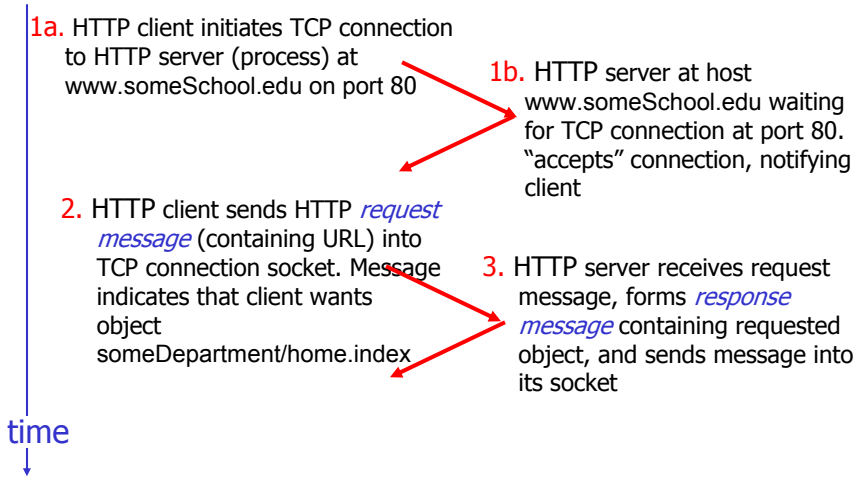
- Internet:
  - HTTP: in-band signaling; ftp: out-of-band signaling
  - RSVP (signaling) separate from routing, forwarding.
- PSTN (public switched telephone network):
  - SS7 (packets-switched control network) separate from (circuit-switched) call trunk lines
  - Earlier tone-based (in-band signaling)

2

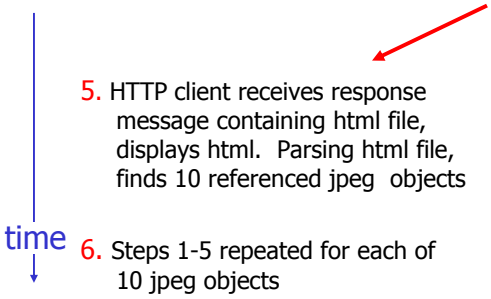
Internet: HTTP – inband signaling

Suppose user enters URL

`www.someSchool.edu/someDepartment/home.index`

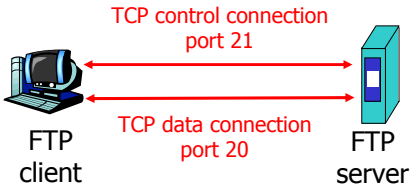


Nonpersistent HTTP (cont.)



FTP: Separate Control, Data Connections

- FTP client contacts FTP server at port 21
- Client obtains authorization over control connection
- Client browses remote directory via commands sent over control connection.
- When server receives file transfer commend server opens new TCP data connection to client
- After transferring one file, server closes connection.



- Server opens 2nd TCP data connection to transfer another file.
- Control connection: "out of band" signaling
- FTP server maintains "state": current directory, earlier authentication

5

Separate Control, Data: Why (or Why Not)?

Why?

- Allows concurrent control + data
- Allows perform authentication at control level
- Simplifies processing of data/control streams – higher throughput
- Provide QoS appropriate for control/data streams

Why not?

- Separate channels complicate management, increases resource requirements
- Can increase latency, e.g., http – two top connections vs. one.

6

2: 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:
  - TCP sequence numbers, timer values, RTT estimates
  - RSVP router maintain lists of upstream sender IDs, downstream receiver reservations

7

State: Senders, Receivers

- Sender: network node that (re)generates signaling (control) msgs to install, keep-alive, remove state from other nodes
- Receiver: node that creates, maintains, removes state based on signaling msgs received from sender

8

Hard-state

- State *installed* by receiver on receipt of *setup msg* from sender
- State *removed* by receiver on receipt of *teardown msg* from sender
- *Default assumption:* state valid unless told otherwise
  - In practice: failsafe-mechanisms (to remove orphaned state) in case of sender: e.g., receiver-to-sender "heartbeat": Is this state still valid?
- Examples:
  - TCP
  - ST-II (Internet hard-state signaling)

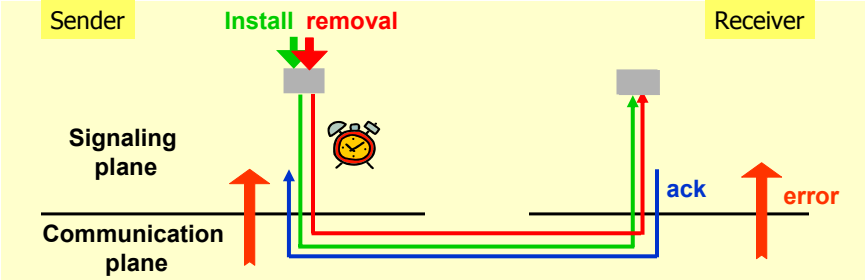
9

Soft-state

- State *installed* by receiver on receipt of setup (trigger) msg from sender (typically, an endpoint)
  - sender also sends periodic *refresh msg*: indicating receiver should continue to maintain state
- State *removed* by receiver via timeout, in absence of refresh msg from sender
- Default assumption: state becomes invalid unless refreshed
  - in practice: explicit state removal (*teardown*) msgs also used
- Examples:
  - RSVP
  - RTP

10

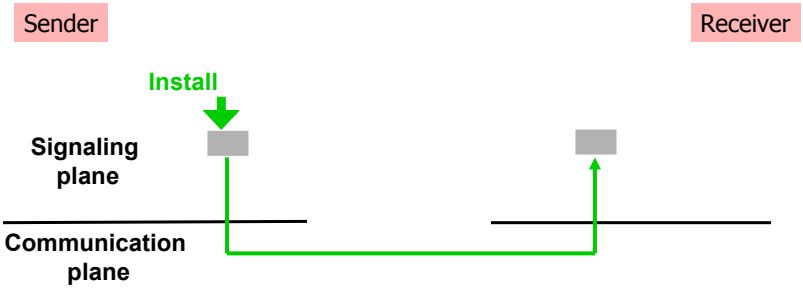
Hard-state Signaling



- Reliable signaling
- State removal by request
- Requires additional error handling
  - E.g., sender failure

11

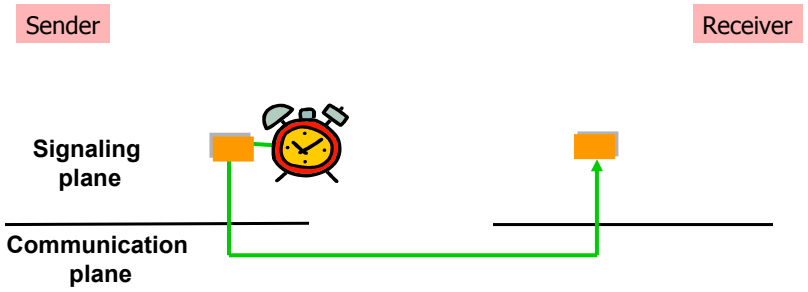
Soft-state Signaling



- Best effort signaling

12

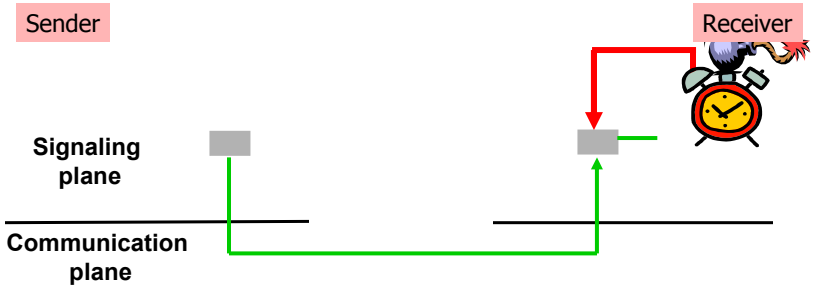
Soft-state Signaling



- Best effort signaling
- Refresh timer, periodic refresh

13

Soft-state Signaling



- Best effort signaling
- Refresh timer, periodic refresh
- State time-out timer, state removal only by time-out

14

Soft-state: Claims

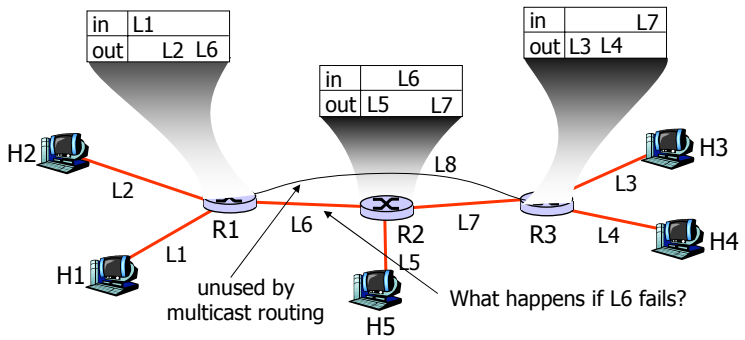
- "Systems built on soft-state are robust" [Raman 99]
- "Soft-state protocols provide … greater robustness to changes in the underlying network conditions …" [Sharma 97]
- "Obviates the need for complex error handling software" [Balakrishnan 99]

What does this mean?

15

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



16

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

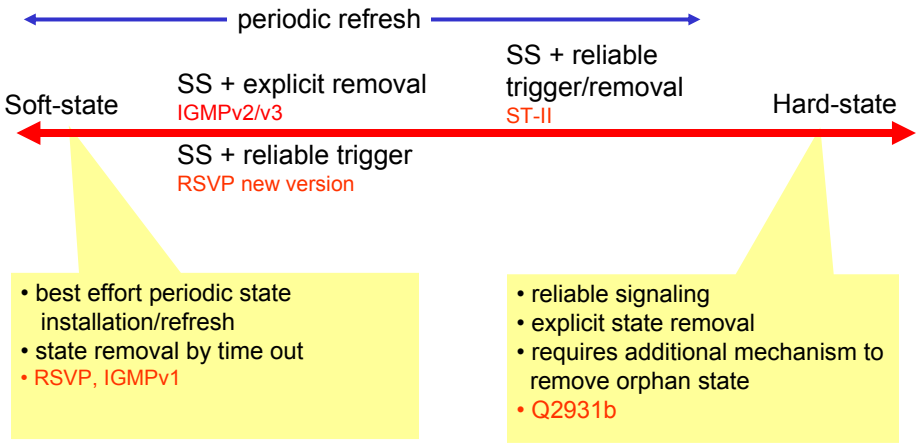
17

Soft-state: Refreshes

- Refresh msgs 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 msgs unreliable
  - would like triggers to result in state-installation asap
  - enhancement: add receiver-to-sender refresh_ACK for triggers
  - e.g., see "Staged Refresh Timers for RSVP"

18

Signaling Spectrum



19

Hard-state Versus Soft-state: Discussion

Q: which is preferable and why?

hard state:

- better if message OH really high
- potentially greater consistency
- system wide coupling → difficult to analyze

soft state:

- robustness, shorter convergence times
- easily decomposed → simpler analysis

20

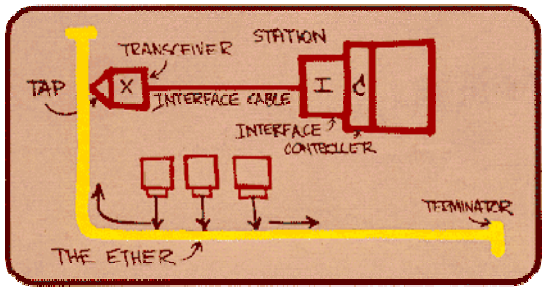
3. Randomization

- Randomization used in many protocols
- Examples:
  - Ethernet multiple access protocol
  - Randomization in Router Queue Management RED
  - Router (de)synchronization
  - Switch scheduling

21

Ethernet

- Single shared broadcast channel
- 2+ simultaneous transmissions by nodes: interference
  - only one node can send successfully at a time
- *Multiple access protocol:* distributed algorithm that determines how nodes share channel, i.e., determine when node can transmit



Metcalfe's Ethernet sketch

22

Ethernet's CSMA/CD

Jam Signal: make sure all other transmitters are aware of collision; 48 bits;

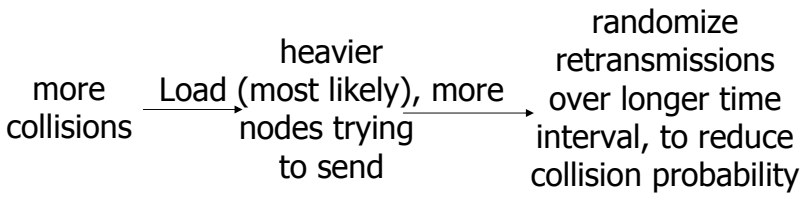
Exponential Backoff:

- First collision for given packet: choose K randomly from {0,1}; delay is K x 512 bit transmission times
- After second collision: choose K randomly from {0,1,2,3}…
- After ten or more collisions, choose K randomly from {0,1,2,3,4,…,1023}

23

Ethernet's Use of Randomization

- *Resulting behavior:* probability of retransmission attempt (equivalently length of randomization interval) adapted to current load
  - Simple, load-adaptive, multiple access



24

The Bottom Line

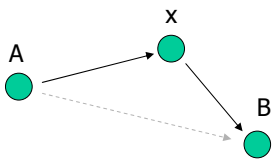
- Why does Ethernet use randomization:
  to desynchronize:

  A distributed adaptive algorithm to spread out load over time when there is contention for multiple access channel

25

4: Indirection

Indirection: rather than reference an entity directly, reference it ("indirectly) via another entity, which in turn can or will access the original entity



"Every problem in computer science can be solved by adding another level of indirection"
— Butler Lampson

26

Mobility and Indirection

How do *you* contact a mobile friend?

Consider friend frequently changing addresses, how do you find her?

- Search all phone books?
- Call her parents?
- Expect her to let you know where he/she is?

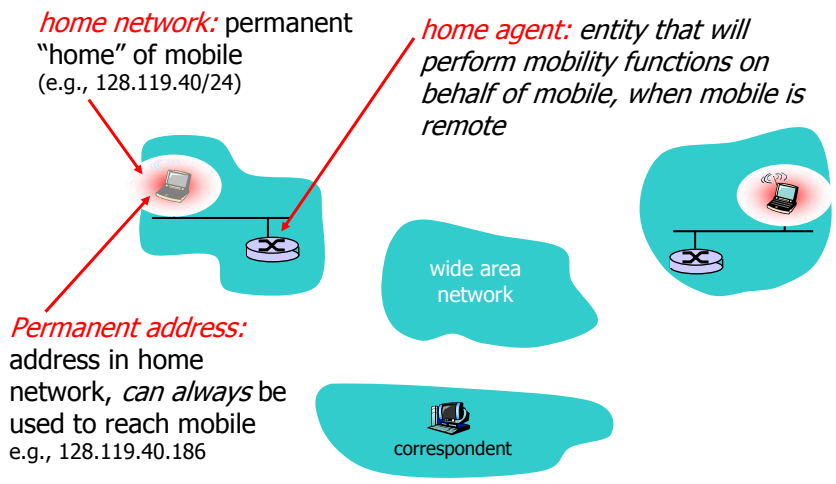
27

Mobility and Indirection:

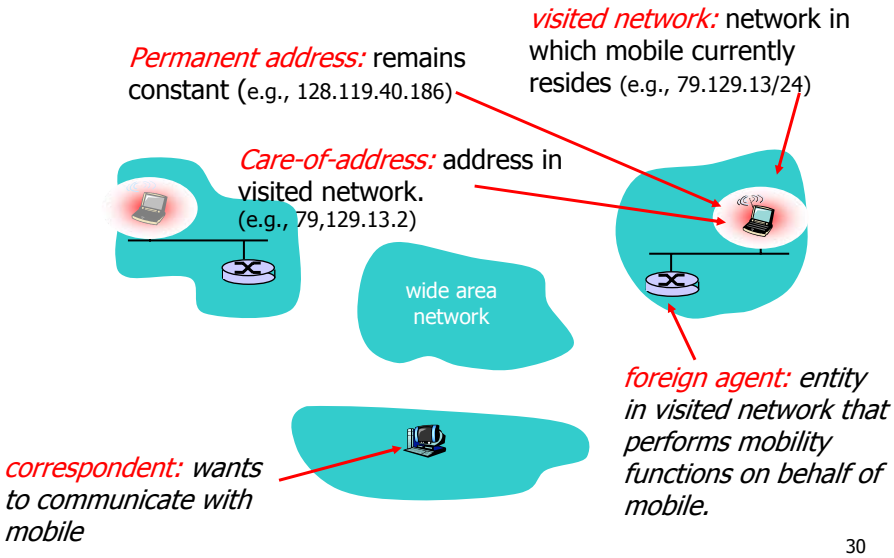
- Mobile node moves from network to network
- Correspondents want to send packets to mobile node
- Two approaches:
  - *Indirect routing:* communication from correspondent to mobile goes through home agent, then forwarded to remote
  - *Direct routing:* correspondent gets foreign address of mobile, sends directly to mobile

28

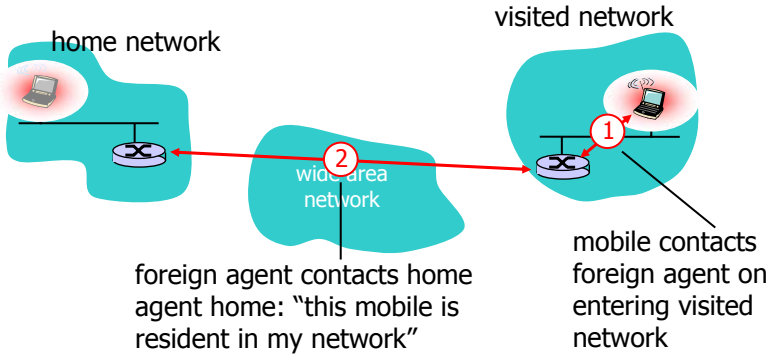
Mobility: Vocabulary



Mobility: More Vocabulary



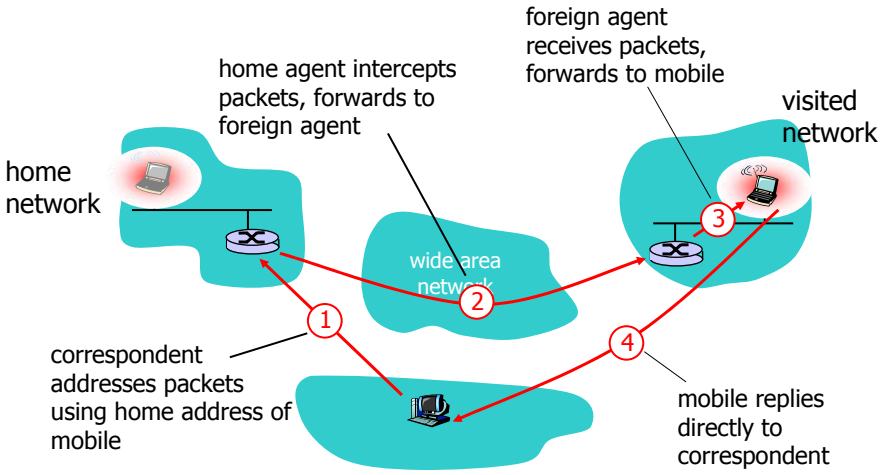
Mobility: Registration



End result:

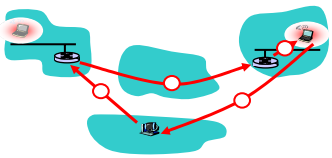
- Foreign agent knows about mobile
- Home agent knows location of mobile

Mobility Via Indirect Routing



Indirect Routing: Comments

- Mobile uses two addresses:
  - Permanent address: used by correspondent (hence mobile location is *transparent* to correspondent)
  - Care-of-address: used by home agent to forward datagrams to mobile
- Foreign agent functions may be done by mobile itself
- Triangle routing: correspondent-home-network-mobile
  - Inefficient when correspondent, mobile are in same network



33

Mobility Via Indirection: Why Indirection?

- Transparency to correspondent
- "Mostly" transparent to mobile (except that mobile must register with foreign agent)
  - Transparent to routers, rest of infrastructure
  - Potential concerns if egress filtering is in place in origin networks (since source IP address of mobile is its home address): spoofing?

38

Indirection: Summary

We've seen indirection used in many ways:

- Mobility
- Multicast
- Internet indirection

The uses of indirection:

- Sender does not need to know receiver ID – do not *want* sender to know intermediary identities
- Beauty, grace, elegance
- Transparency of indirection is important
- Performance: is it more efficient?

39

5. Multiplexing

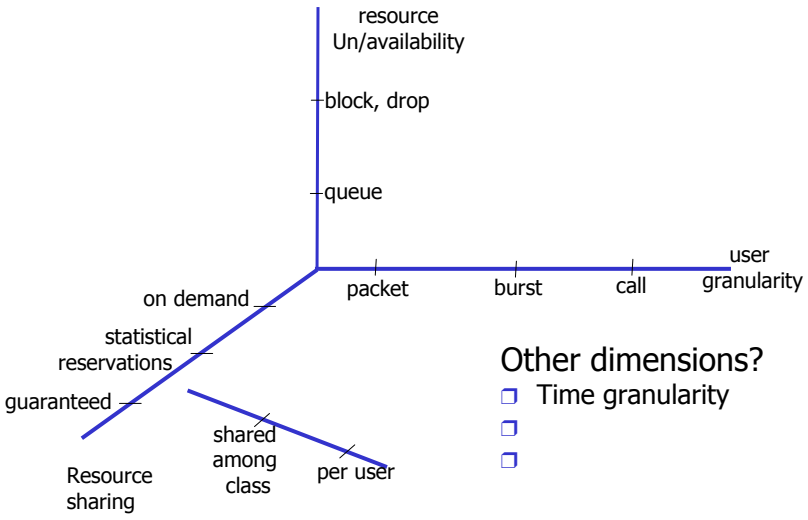
Multiplexing: *sharing* resource(s) among users of the resource.

Multiplexed human resources:

- Roadways, traffic intersections
- Bathrooms (except for the rich)
- Reserve reading
- …

40

Many Dimensions of Multiplexing

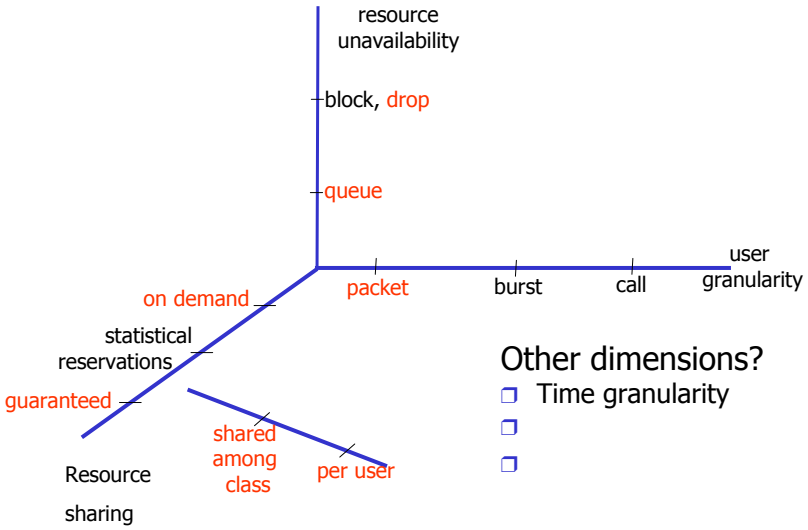


Other dimensions?

- Time granularity
-
-

41

Packet-level Multiplexing



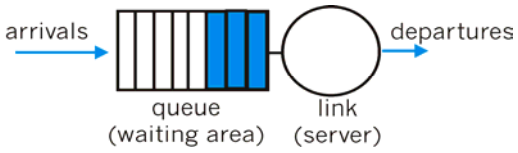
Other dimensions?

- Time granularity
-
-

42

Scheduling and Policing *Packets*

- Scheduling: choose next packet to send on link
- E.g.:
  - FIFO (first in first out) scheduling
  - Round Robin



43

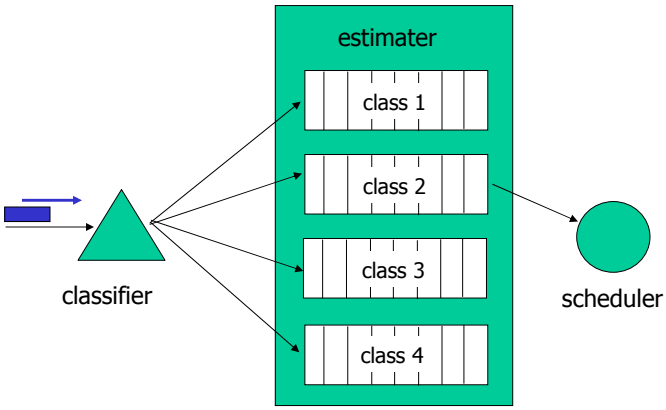
Policing Mechanisms

Policing: limit traffic to not exceed declared parameters

- E.g.:
  - Token bucket

44

General Model of Class-based Link Scheduling



45

Multiplexing: What Has One Learned?

- Predictable traffic makes allocating resources easier
- Lots of mechanism
- Admission control: deterministic performance by non-statistical allocation of resources

46

6. Virtualization of Networks

Virtualization of resources:

powerful abstraction in systems engineering

- Computing examples: virtual memory, virtual devices
  - Virtual machines: e.g., Java
  - IBM VM OS from 1960's/70's
- Layering of abstractions: don't sweat the details of the lower layer, only deal with lower layers abstractly

47

The Internet: Virtualizing Local Networks

1974: multiple unconnected networks

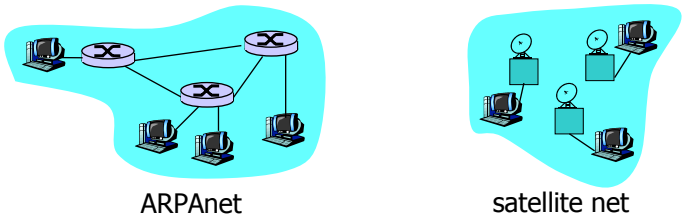
- ARPAnet
- Data-over-cable networks
- Packet satellite network (Aloha)
- Packet radio network

… differing in:

- Addressing conventions
- Packet formats
- Error recovery
- Routing

48

Cerf & Kahn: Interconnecting Two Networks



- "…interconnection must preserve intact the internal operation of each network."
- " ..the interface between networks must play a central role in the development of any network interconnection strategy. We give a special name to this interface that performs these functions and call it a GATEWAY."
- ".. prefer that the interface be as simple and reliable as possible, and deal primarily with passing data between networks that use different packet-switching strategies
- "…address formats is a problem between networks because the local network addresses of TCP's may vary substantially in format and size. A uniform internetwork TCP address space, understood by each GATEWAY and TCP, is essential to routing and delivery of internetwork packets."

49

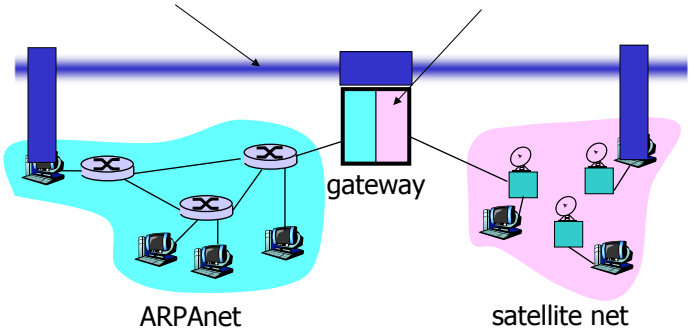
Cerf & Kahn: Interconnecting Two Networks

Internetwork layer:

- Addressing: internetwork appears as a single, uniform entity, despite underlying local network heterogeneity
- Network of networks

Gateway:

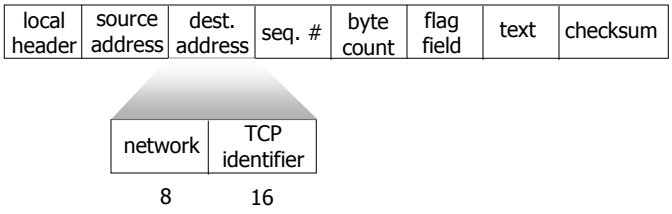
- "Embed internetwork packets in local packet format or extract them"
- Route (at internetwork level) to next gateway



50

Historical Aside

Proposed Internetwork packet in 1974:



51

Cerf & Kahn's Internetwork Architecture

What is virtualized?

- Two layers of addressing:
  internetwork and local network
- New layer makes everything homogeneous
- Underlying local network technology (cable, satellite, 56K modem) is "invisible" at internetwork layer

52

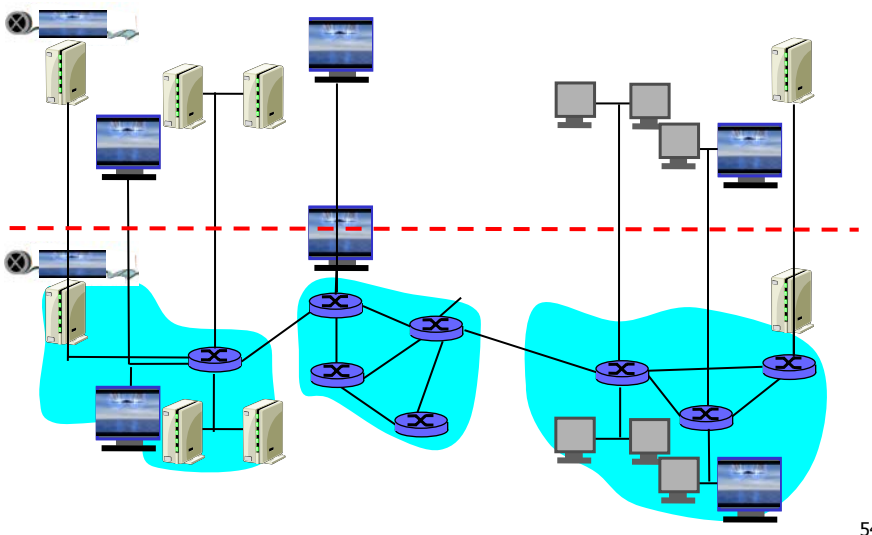
Resilient Overlay Networks

Overlay network:

- Applications, running at various sites as "nodes" on an application-level network
- Create "logical" links (e.g., TCP or UDP connections) pairwise between each other
- Each logical link: multiple physical links, routing defined by native Internet routing

53

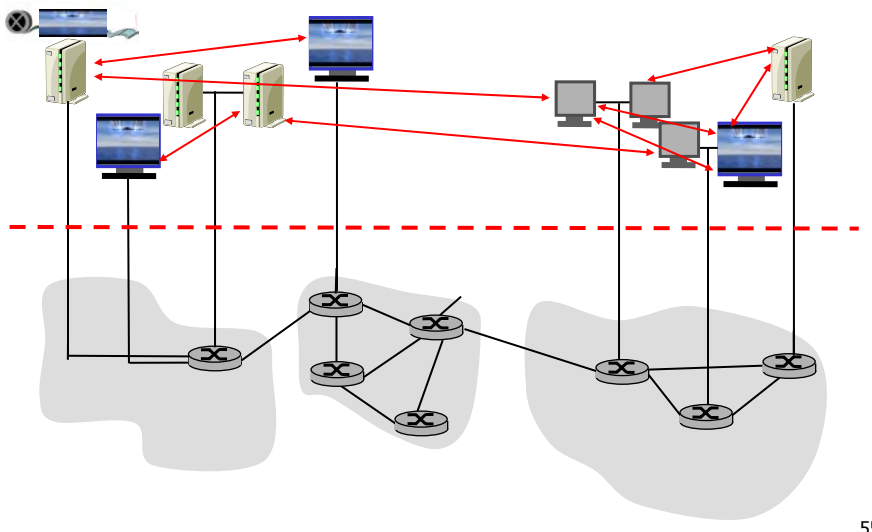
Overlay Network



54

Overlay Network (2.)

Focus at the application level



55

Virtual Private Networks (VPN)

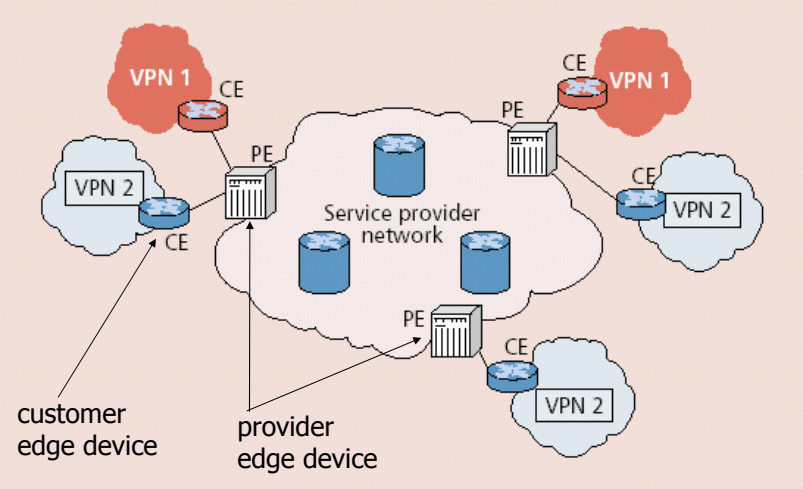
VPNs

Networks perceived as being private networks by customers using them, but built over shared infrastructure owned by service provider (SP)

- SP infrastructure:
  - Backbone
  - Provider edge devices
- Customer:
  - Customer edge devices (communicating over shared backbone)

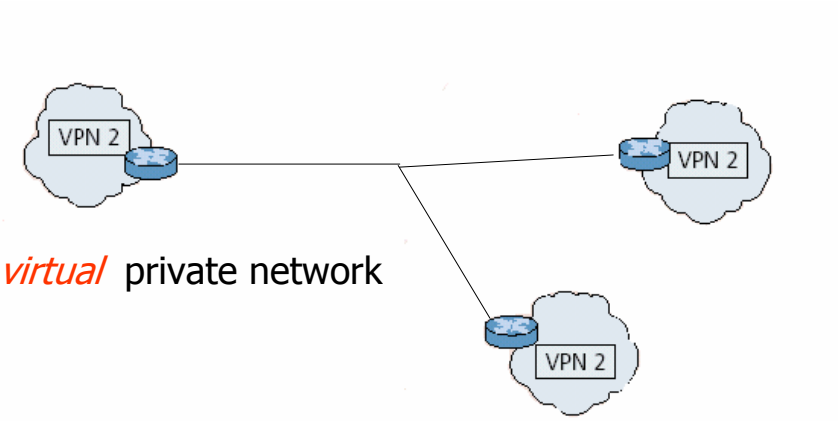
56

VPN Reference Architecture



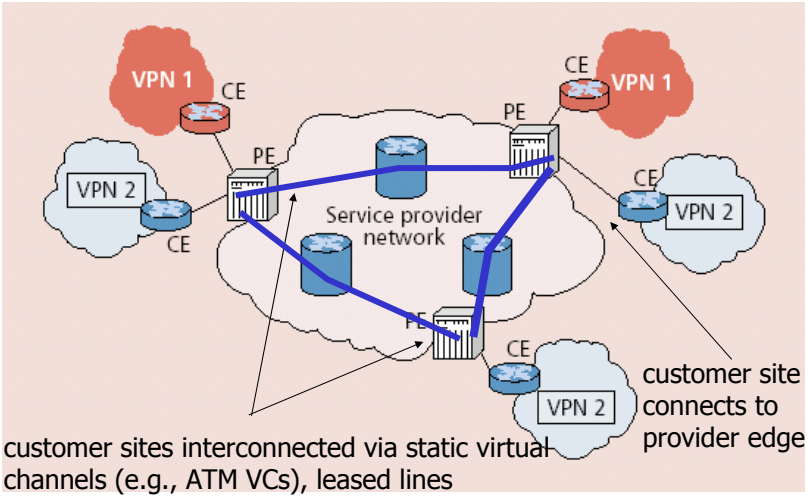
57

VPN: Logical View



58

Leased-line VPN

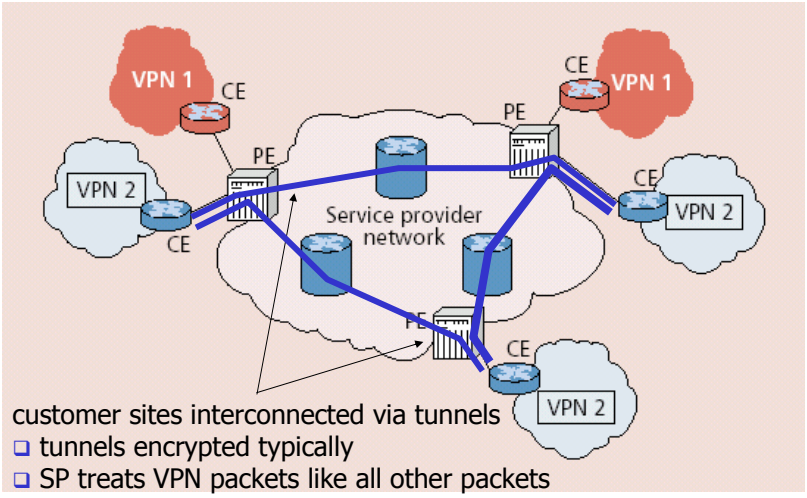


customer sites interconnected via static virtual channels (e.g., ATM VCs), leased lines

59

Customer Premise VPN

- All VPN functions implemented by customer



customer sites interconnected via tunnels

- tunnels encrypted typically
- SP treats VPN packets like all other packets

60

Drawbacks

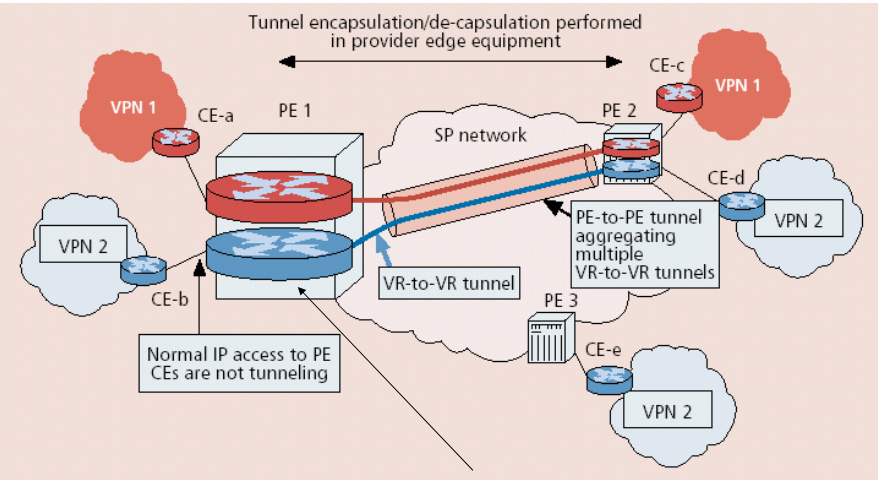
- Leased-line VPN: configuration costs, maintainence by SP: long time, much manpower
- CPE-based VPN: expertise by customer to acquire, configure, manage VPN

Network-based VPN

- Customer's routers connect to SP routers
- SP routers maintain separate (independent) IP contexts for each VPN
  - Sites can use private addressing
  - Traffic from one vpn can not be injected into another

61

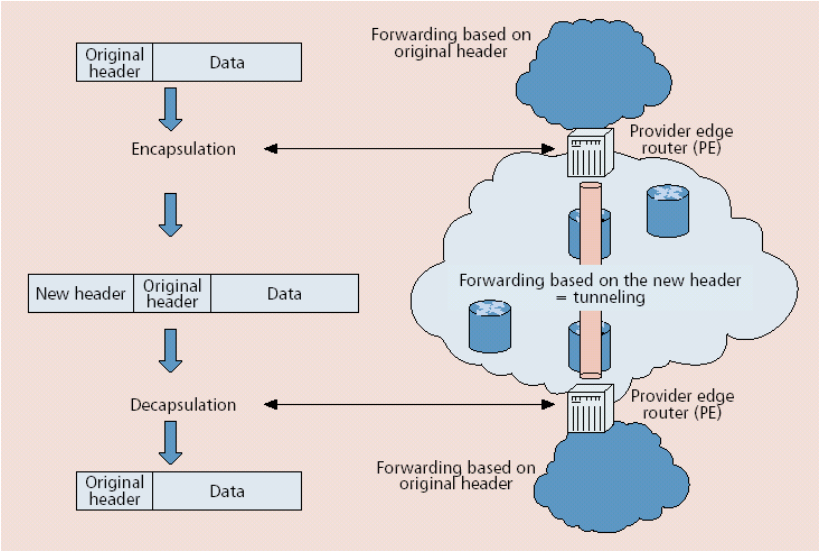
Network-based Layer 3 VPNs



multiple virtual routers
in single provider edge device

62

Tunneling



63

VPNs: Why?

- Privacy
- Security
- Works well with mobility:
  - looks like you are always at home
- Cost:
  - Many forms of newer VPNs are cheaper than leased line VPN's
  - Ability to share at lower layers
  - Exploit multiple paths, redundancy, fault-recovery (lower layers)
  - Need isolation mechanisms to ensure appropriate resources sharing
- Abstraction and manageability:
  - All machines with addresses that are "in" are trusted no matter where they are

64

7. Designs for Scale

How to deal with large numbers (millions) of entities in a system?

- IP devices in the internet (0.5 billion)
- Users in P2P network (millions)

More generally:

- Are there advantages to large scale?
- "For every type of animal there is a most convenient size, and a large change in size *inevitably* carries with it a change of form."
  True for networks?

65

Dealing With Scale: Hierarchical Routing

Scale: with 500 million destinations:

- Can't store all dest's in routing tables!
- Routing table exchange would swamp links!

Administrative autonomy

- Internet = network of networks
- Each network admin may want to control routing in its own network

66

Dealing With Scale

Question: What are the *advantages* of large scale?

- Take advantage of having to do similar things for others (caching)
- Fault tolerance:
  - Large number of servers
  - We have redundancy; multiple routes between sites
- Metcalfe's law:
  - "Value" of a network is proportional to square of number of things connected (bigger is better)
- Law of large numbers:
  - Allocation of resources based on average usage rather than peak
- Amortizing upgrade maintenance over a large population:
  - Popular network and services likely to be upgraded/improved
- Denial of service:
  - Size/replication makes it harder to attack
  - More generally, a system with replicated components is more survivable.

67

Dealing With Scale

Discussion: "For every type of animal there is a most convenient size, and a large change in size inevitably carries with it a change of form."

Question: True for networks? Why? How so? Examples?

- Ethernet doesn't scale up: geographical distance, speed of light delays degrade performance of random access protocols. (geographic scaling). Maybe scale with # users in geographically narrow net if bandwidth scales with users.
- As number of communicants scales, need to change/improve manner in which to access communication channel
  - Example: small number of students, versus 500-class lecture. Keeping bandwidth fixed as # users scales.

68

Dealing With Scale

Discussion: "For every type of animal there is a most convenient size, and a large change in size inevitably carries with it a change of form."

Question: True for networks? Why? How so? Examples?

- Routing:
  - Large number of users and optimal routes => requires lots of info to compute routes, etc...
  - Doesn't scale
- Certain services become necessary when you get big
  - Name storage/translation: dns, phone books
- A single centralized site eventually breaks
  - Need replication or other form of distribution
- Switched vs. routed networks
  - Change from layer 2 switched networks to layer 3 routed networks as # users gets bigger

69

End of "Design Principles"!

Goals

- Review
- Framework for covering "advanced topics"
- Material that is timely, timeless: hot now but also long shelf life
- Synthesis: deeper understanding; "see the forest for the trees"

70