EPL606

Topic 1
Introduction
Part B - Design Considerations

The majority of the slides in this course are adapted from the accompanying slides to the books by Larry Peterson and Bruce Davie and by Jim Kurose and Keith Ross. Additional slides and/or figures from other sources and from Vasos Vassiliou are also included in this presentation.

Design Considerations

- How to determine split of functionality
  - Across protocol layers
  - Across network nodes
- Assigned Reading
  - [SRC84] End-to-end Arguments in System Design
  - [Cla88] Design Philosophy of the DARPA Internet Protocols

Goals [Clark88]

- Connect existing networks
  - initially ARPANET and ARPA packet radio network
- Survivability
  - ensure communication service even in the presence of network and router failures
- Support multiple types of services
- Must accommodate a variety of networks
- Allow distributed management
- Allow host attachment with a low level of effort
- Be cost effective
- Allow resource accountability

Challenge

- Many differences between networks
  - Address formats
  - Performance – bandwidth/latency
  - Packet size
  - Loss rate/pattern/handling
  - Routing
- How to internetwork various network technologies

Challenge 1: Address Formats

- Map one address format to another. Why not?
- Provide one common format
  - map lower level addresses to common format

Challenge 2: Different Packet Sizes

- Define a maximum packet size over all networks. Why not?
- Implement fragmentation/re-assembly
  - who is doing fragmentation?
  - who is doing re-assembly?

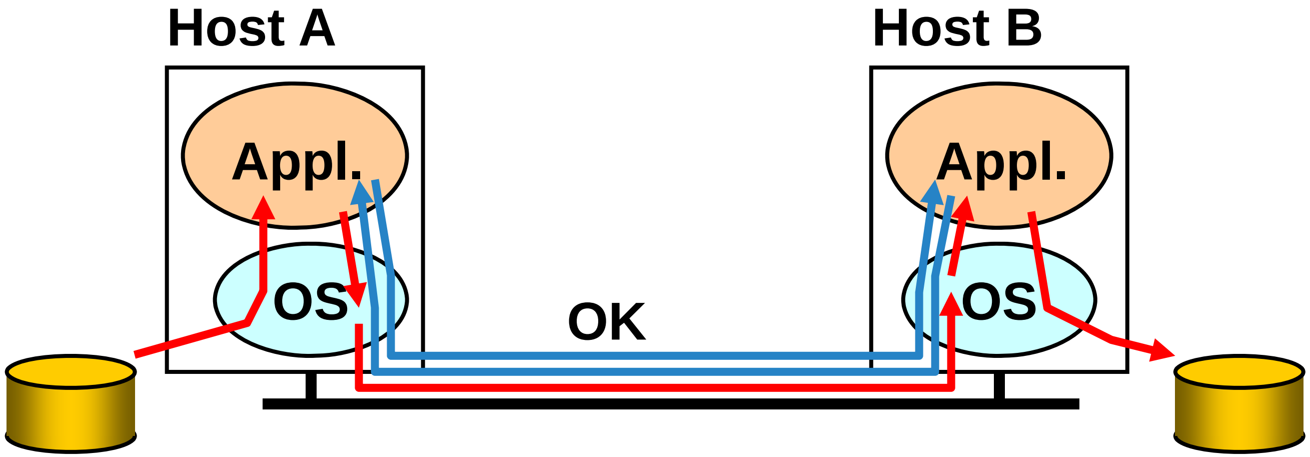
Gateway Alternatives

- Translation
  - Difficulty in dealing with different features supported by networks
  - Scales poorly with number of network types (N^2 conversions)
- Standardization
  - “IP over everything” (Design Principle 1)
  - Minimal assumptions about network
  - Hourglass design

End-to-End Argument (Principle 2)

- Deals with where to place functionality
  - Inside the network (in switching elements)
  - At the edges
- Argument
  - There are functions that can only be correctly implemented by the endpoints – do not try to completely implement these elsewhere
  - Caveat: can provide a partial form as performance enhancement
  - Guideline not a law

Example: Reliable File Transfer



- Solution 1: make each step reliable, and then concatenate them
- Solution 2: end-to-end check and retry

E2E Example: File Transfer

- Even if network guaranteed reliable delivery
  - Need to provide end-to-end checks
  - E.g., network card may malfunction
  - The receiver has to do the check anyway!
- Full functionality can only be entirely implemented at application layer; no need for reliability from lower layers
- Is there any need to implement reliability at lower layers?

Discussion

- Yes, but only to improve performance
- If network is highly unreliable
  - Adding some level of reliability helps performance, not correctness
  - Don’t try to achieve perfect reliability!
  - Implementing a functionality at a lower level should have minimum performance impact on the application that do not use the functionality

Examples

- What should be done at the end points, and what by the network?
  - Reliable/sequenced delivery?
  - Addressing/routing?
  - Security?
  - What about Ethernet collision detection?
  - Multicast?
  - Real-time guarantees?

Internet & End-to-End Argument

- At network layer provides one simple service: best effort datagram (packet) delivery
- Only one higher level service implemented at transport layer: reliable data delivery (TCP)
  - Performance enhancement; used by a large variety of applications (Telnet, FTP, HTTP)
  - Does not impact other applications (can use UDP)
  - Original TCP/IP were integrated – Reed successfully argued for separation
- Everything else implemented at application level
- Does FTP look like E2E file transfer?
  - TCP provides reliability between kernels not disks

Principle 3

- Best effort delivery
- All packets are treated the same
- Relatively simple core network elements
- Building block from which other services (such as reliable data stream) can be built
- Contributes to scalability of network

Principle 4

- Fate sharing
- Critical state only at endpoints
- Only endpoint failure disrupts communication
- Helps survivability

Principle 5

- Soft-state
  - Announce state
  - Refresh state
  - Timeout state
- Penalty for timeout – poor performance
- Robust way to identify communication flows
  - Possible mechanism to provide non-best effort service
- Helps survivability

Principle 6

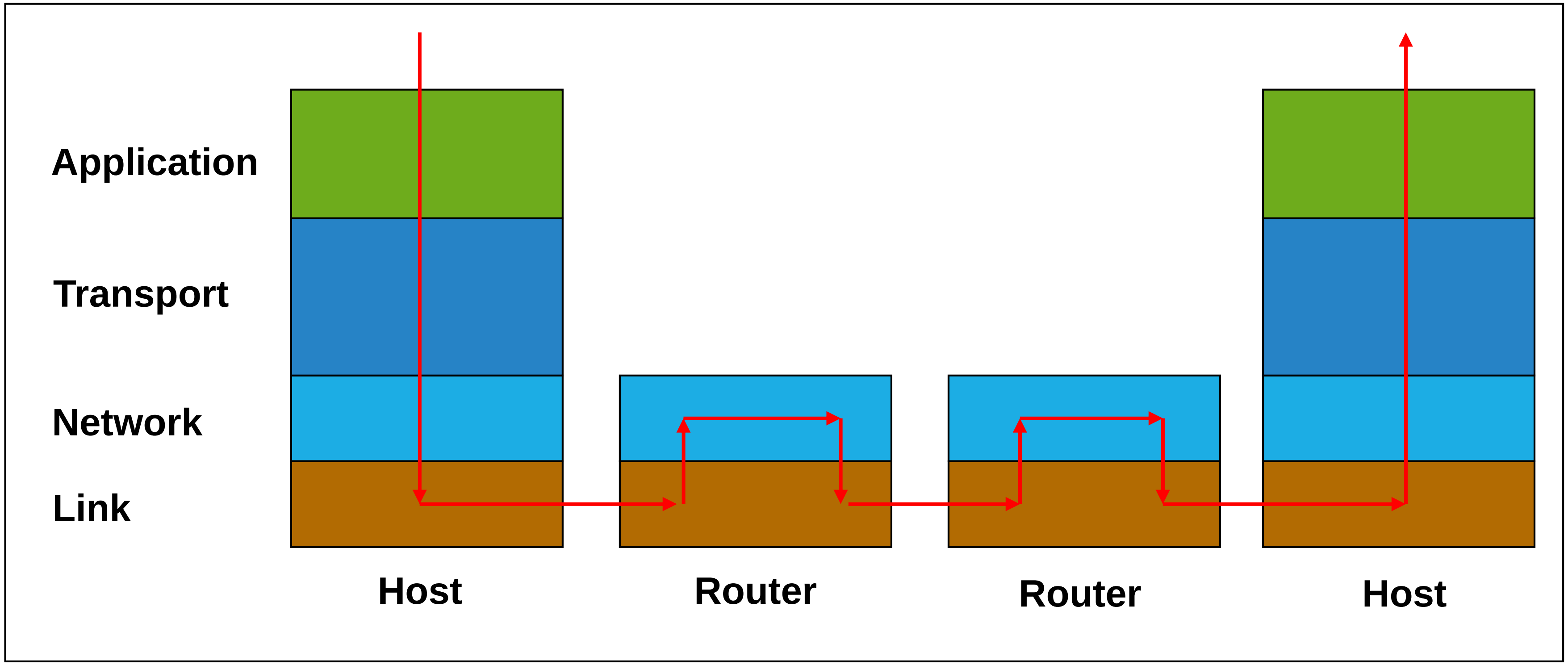
- Decentralization
- Each network owned and managed separately
- Will see this in BGP routing especially

Principle 7

- Be conservative in what you send and liberal in what you accept
  - Unwritten rule
- Especially useful since many protocol specifications are ambiguous
- E.g. TCP will accept and ignore bogus acknowledgements

IP Layering (Principle 8)

- Relatively simple
- Sometimes taken too far

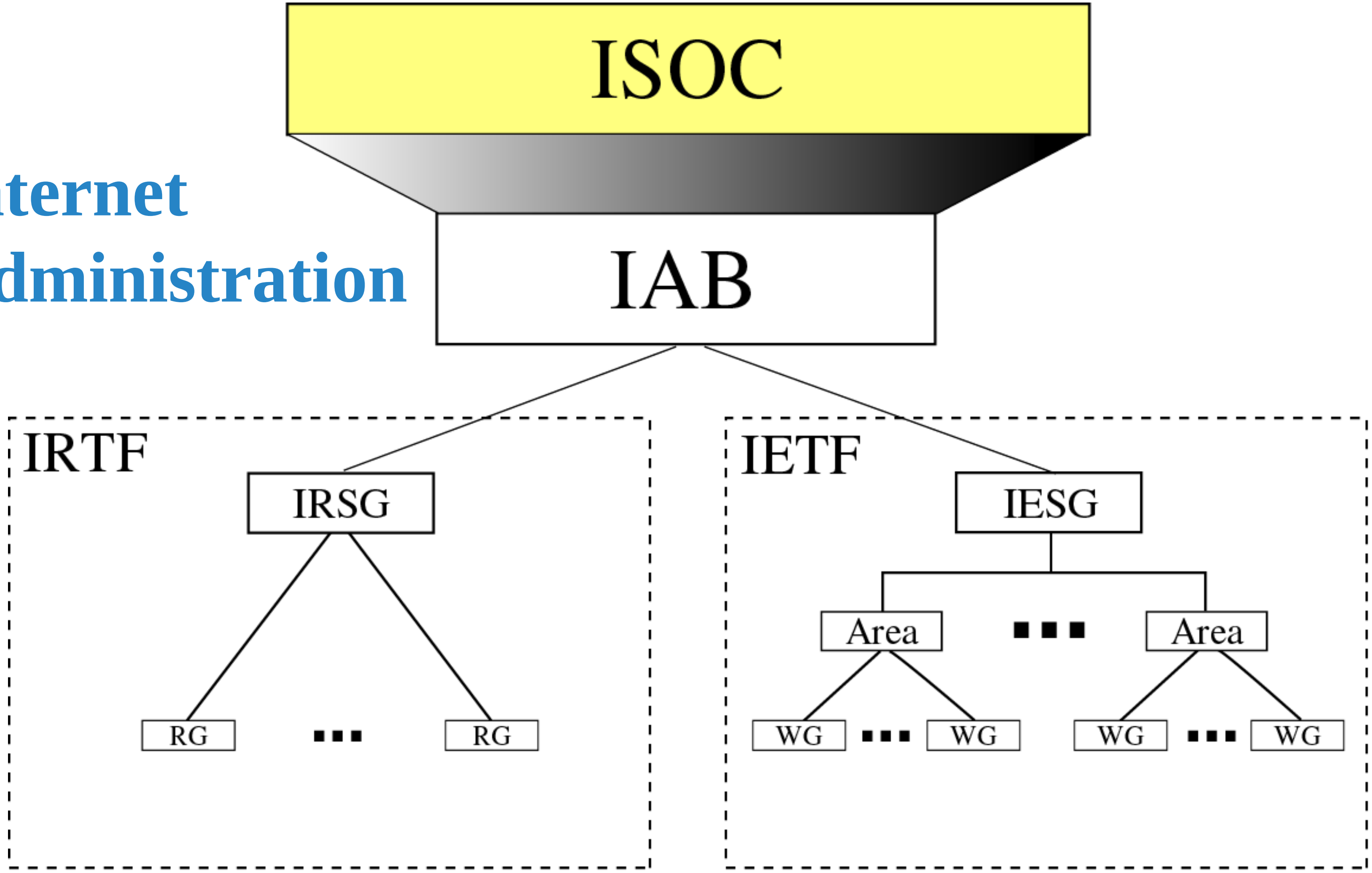


Integrated Layer Processing (ILP)

- Layering is convenient for architecture but not for implementations
- Combining data manipulation operations across layers provides gains
  - E.g. copy and checksum combined provides 90Mbps vs. 60Mbps separated

How is IP Design Standardized?

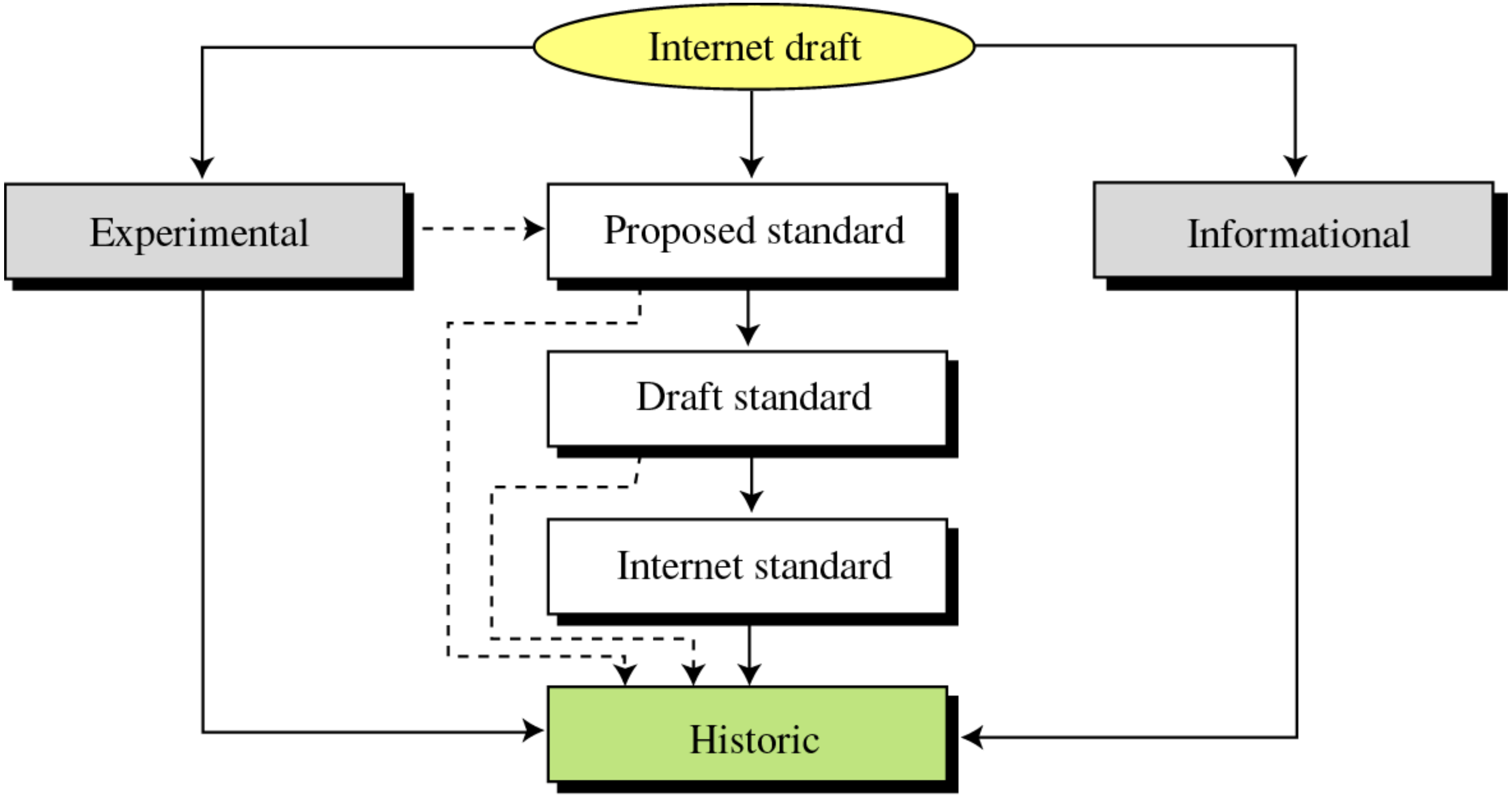
Internet Administration



How is IP Design Standardized?

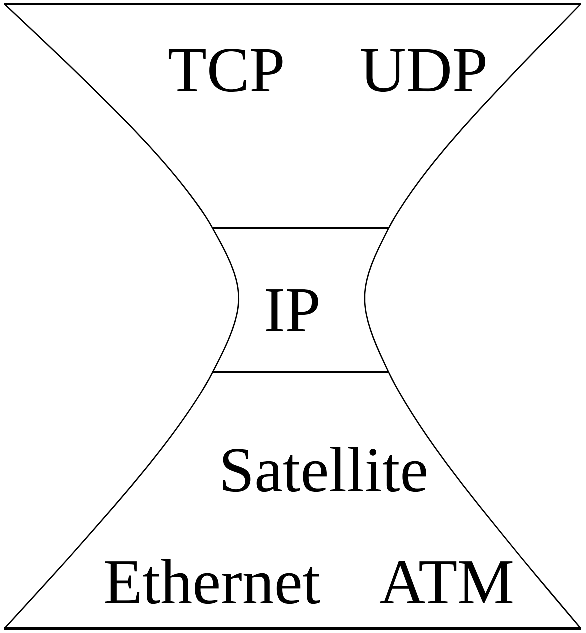
- IETF
  - Voluntary organization
  - Meeting every 4 months
  - Working groups and email discussions
- “We reject kings, presidents, and voting; we believe in rough consensus and running code” (Dave Clark 1992)
  - Need 2 independent, interoperable implementations for standard
- IRTF
  - End2End
  - Reliable Multicast, etc..

Maturity levels of an RFC



Summary: Internet Architecture

- Packet-switched datagram network
- IP is the "compatibility layer"
  - Hourglass architecture
  - All hosts and routers run IP
- Stateless architecture
  - no per flow state inside network



Summary: Minimalist Approach

- Dumb network
  - IP provide minimal functionalities to support connectivity
    - Addressing, forwarding, routing
- Smart end system
  - Transport layer or application performs more sophisticated functionalities
    - Flow control, error control, congestion control
- Advantages
  - Accommodate heterogeneous technologies (Ethernet, modem, satellite, wireless)
  - Support diverse applications (telnet, ftp, Web, X windows)
  - Decentralized network administration