Principles of Computer Systems: Layers

Katerina Argyraki

School of Computer & Communication Sciences

Definition

- Layer = group of modules
  - *Internet transport layer = UDP + TCP*
  - *Internet network layer = IP*
- A module communicates with modules in layer above/below, on the same stack instance, through API
  - *send/receive calls/notifications*
- A module communicates with modules in the same layer, on a different stack instance, through a protocol
  - *header semantics*

Layering violation

- When a module violates these rules
  - *relies on information about other layers that is not passed through the proper APIs*
  - *interprets a header that belongs to another layer*

- When a module makes assumptions about the operation of another module that belongs to a different layer

How to pick layers?

- Look for "natural" boundaries
  - *related functionality*
  - *developer expertise*
- Consider performance
  - *pick the finest layering that makes sense and provides the required performance*

Outline

- Data delivery
- Reliable data delivery
- Congestion control
- Address depletion
- Scaling content distribution

Outline

- Data delivery
- Reliable data delivery
- Congestion control
- Address depletion
- Scaling content distribution

Data delivery

- Deliver a packet from a source end-system to a destination end-system
- Partly solved at the link layer of local networks
  - *takes a packet across one physical link*
- Partly solved at the network layer of local networks (= link layer of the Internet)
  - *takes a packet across a sequence of physical links (= one local network)*
- Partly solved at the network layer of the Internet
  - *takes a packet across a sequence of local networks (= the Internet)*

Internet network layer (IP)

- Hierarchical addresses

- IP routing learns one route per IP prefix
- IP forwarding maintains state per IP prefix

- Hierarchy => scalability
- Two levels of aggregation
    - *intra-AS: addresses of each local network aggregated into one local prefix*
    - *inter-AS: multiple local prefixes of each AS aggregated into one externally visible prefix*

Internet link layer (Ethernet)

- Flat addresses

- L2 learning learns one route per active MAC address
- L2 forwarding maintains state per active MAC address

- Scales well enough for local networks

Outline

- Data delivery
- Reliable data delivery
- Congestion control
- Address depletion
- Scaling content distribution

Reliable data delivery

- Deliver a packet from a source end-system to a destination end-system with the capability to recover from corruption and loss

- Partly solved at the transport layer
  - *TCP offers reliable data delivery end-to-end*

- Partly solved at the link layer (of local networks)
  - *e.g., reliable data delivery across a wireless link*

The end-to-end argument

- Saltzer, Reed, and Clark, 1981
- If an upper layer must provide X anyway,
  don’t go out of your way to provide X at a lower layer
- It may make sense to also provide X at a lower layer
  as a performance optimization…
- …but consider the impact on upper-layer modules that do not need X.

A few more examples

- Acknowledgment of reception
- Duplicate suppression
- In-order delivery

Outline

- Data delivery
- Reliable data delivery
- Congestion control
- Address depletion
- Scaling content distribution

Congestion control

- Maximize throughput without creating network congestion

- Solved at the transport layer
  - *through the TCP congestion control algorithm*
  - *interprets a timeout as an indication of packet loss*
  - *interprets packet loss as an indication of network congestion*

(Fixed retransmission timeout)

- Bad idea: early retransmissions can lead to congestion collapse

- Layering violation: a module assumes that it knows when is the right time to retransmit, even though the relevant information belongs to the layers below

Packet loss as an indication of congestion

- There may be packet loss without congestion
  - e.g., a wireless link experiences loss due to fading
- There may be congestion without packet loss
  - e.g., a link with a very large queue experiences congestion
- Layering violation(?): TCP assumes that packet loss equals network congestion

Explicit Congestion Notification (ECN)

- ECN bits in both the IP and the TCP header
- A router sets an ECN bit in the IP header to signal congestion to the receiver
- The (network layer of the) receiver passes that information to TCP
- The TCP receiver sets an ECN bit in the TCP header
  to signal congestion to the TCP sender

- Network congestion detected at the network layer
- Information passed to TCP through the API

Outline

- Data delivery
- Reliable data delivery
- Congestion control
- Address depletion
- Scaling content distribution

Address depletion

- We are (were) running out of IPv4 addresses
- Solved by introducing a new technology at the network layer (IPv6)
- Solved through Network Address Translation (NAT)

Network Address Translation

- NAT gateway keeps state per TCP connection
- Maps each local {IP address, TCP port number} tuple to a different, externally visible TCP port number
- Embeds the local IP address in the TCP port number
- Layering violation: the NAT gateway assumes that it understands the semantics of the TCP header

Outline

- Data delivery
- Reliable data delivery
- Congestion control
- Address depletion
- Scaling content distribution

Scaling content distribution (late 00s)

- How can a content provider serve more clients
  without increasing the transmission rate of its link to the Internet?

- Partly solved through transparent caches
- Partly solved through Content Distribution Networks + dynamic DNS

- Reduce user-content distance
- Distribute load across content servers

These are “dirty” solutions

- Transparent caches hijack TCP connections
- CDNs duplicate network-layer functionality

- Both still need extra round-trip for DNS

- Layer duplication or violation + still limited by DNS

(TRIAD project) Clean solution: a new layer

- New content layer between transport and network

- Network layer: finds path between two end-systems
- Content layer: finds path between an end-system and a piece of content

- Content-layer header specifies target content by name

- Content-layer switches forward packets based on content name
- Content-layer switches may cache content and respond to client

Clean solution: a new layer

- Integrates DNS into the TCP/IP stack

- Subsumes transparent caches and CDNs
- Removes DNS round-trip

- Changes the layering to avoid violation and duplication

Clean solution: a new layer

- Forwarding based on content names
- They could be hierarchical, but could they enable efficient aggregation?

Deployment

- Not all switches need to be content-layer switches
  - *the same way not all switches are network-layer switches*
- Only the switches (routers) at domain boundaries
  - *the same ones that run e-BGP*
- Content-layer switches form an overlay over network-layer switches

IP addresses = transient routing tags

- Used for routing within each domain
- Do not need to be global

- Internet = network of NAT domains

(Also resolves source address spoofing)

- True domain-level path encoded in header
- Source cannot hide the origin domain
- Can spoof only its local neighbors

- Restricts spoofing within local domain

(Also resolves reflector attacks)

- True domain-level path encoded in header
- Source cannot hide the origin domain
- Can spoof only its local neighbors

- Restricts reflection within local domain

Scaling content distribution

- New content layer

- Content names = globally unique
- IP addresses = unique within their domain

- Content switches forward by content name and cache content

- Scalable content distribution
- No need for IPv6
- Limited spoofing and reflector attacks