Assignment #3: Reliable Data Transport Protocol

- Implementing the sending/receiving side of a RDT protocol
- Group of two
- Running in a simulated network environment

The Simulated Environment

- Data units:
  - message is passed between the upper layer and the rdt layer, which can be variable size
  - packet is passed between the rdt layer and the lower layer, which is of fixed size – 64-byte
  - fragmentation may be required
- One-way link latency - typically 100ms
- Message generation:
  - automatically generated by the upper layer at the sending side – 012345678901234567890 ...
  - automatically checked by the upper layer at the receiving side for corruptions, losses, and out-of-order deliveries
- Simulation settings:
  - sim_time, mean_msg_arrivalint, mean_msg_size, outoforder_rate, loss_rate, corrupt_rate, tracing_level
  - rdt_sim 1000 0.1 100 0.02 0 0 0

Helpful Hints

- RDT protocol – any variations of the sliding window protocol with a single timer
  - e.g. Go-Back-N, Selective Repeat
- Data buffer at the sender
- Error detection (never 100% accurate)
  - e.g. Internet checksum
- Packet format: header & payload (all within 64-byte packet)
  - e.g. header fields - payload size, sequence number, acknowledgment sequence number, and the checksum.
- Random numbers

My Simulation Code and Your Turn-in

- C/C++, makefile
- Simulation core (you don’t need to change)
  - rdt_receiver.h, rdt_sender.h, rdt_struct.h, and rdt_sim.cc
- RDT layer (you need to enhance)
  - rdt_sender.cc and rdt_receiver.cc
  - my current version works when the underlying link medium is loss-free, corruption-free, in-order.
- Your turn-in
  - enhanced rdt_sender.cc and rdt_receiver.cc
  - README
- Testing - step by step

Grading

- 20% each for error-free, loss-free, and in-order delivery of your RDT implementation
- 20% for clear documentation (in README file) of your design
- 20% for efficiency
- Extra credit for
  - most efficient implementation
  - finding my mistakes
Content Distribution

- Scalable Internet server systems (last class):
  - serve millions of user requests
  - host Terabytes of data
  - Internet load balancing
- The remaining question:
  - how to get your content to millions of users
  through the public Internet with limited bandwidth
  - Content distribution

Means of Content Distribution

- By content providers, e.g. www.yahoo.com
  - replicated, distributed Internet sites
- By content consumers (or clients), e.g. UR
  - reduce bandwidth consumption of an organization
  through caching – Web caching
- By a third-party
  - content distribution network

Replicated Internet Sites

- Content providers build a set of distributed sites on
  the Internet.
- DNS answers the IP address of the site close
to the client

Web caches (proxy server)

- Cache is installed and shared by a group of users (university,
  company, residential ISP)
- Goal: satisfy client requests without involving the original server.

A Quantitative Study: the Base Case

Assumptions
- 1.5Mbps wide-area access link, 10Mbps local-area network
- wide-area delay 2 seconds, local-area delay 10 milliseconds

Performance
- effective bandwidth to wide-area content - 1.5Mbps
- average access delay
  - Internet delay + LAN delay
  - 2.01 seconds

A Quantitative Study: Upgrading the Access Link

Upgrading the access link
- increase bandwidth of access link to, say, 10Mbps

Performance
- effective bandwidth to wide-area content - 10Mbps
- average access delay
  - Internet delay + LAN delay
  - 2.01 seconds

Not bad! But it is costly to upgrade the access link.
A Quantitative Study: Web Caching

Install a Web cache
- Suppose hit rate is 50%
- Performance
  - effective bandwidth to wide-area content = 3.0 Mbps
  - average access delay = cache miss delay * 50% + cache hit delay * 50%
  - = 2.02 * 50% + 0.01 * 50%
  - = 1.015 seconds

Last Words on Web Caching
- Web cache acts as both HTTP client and server
- Advantages of Web caching:
  - reduce traffic on an institution's access link
  - reduce average access delay for client requests
  - Internet dense with caches enables "poor" content providers to effectively deliver content
- Is the cached page up-to-date?
  - using If-modified-since HTTP header.
  - heuristics: remove pages that are too old.

Distribution through a third party - Content Distribution Network
- Motivation: limited scale for replication by content providers
- CDN is a business: content providers are the CDN customers.
- CDN company installs hundreds of CDN servers throughout Internet
- CDN replicates its customers' content in CDN servers.
- When provider updates content, CDN updates servers.

CDN in Action
- HTTP request for www.yahoo.com with embedded image www.cdn.com/yahoo/sports.gif
- DNS query for www.cdn.com
- CDN's authoritative DNS server
- Nearby CDN server
- CDN company
  - replicates content at CDN servers
  - uses its authoritative DNS server to redirect requests to a nearby CDN server

Determining Nearby CDN Server
- CDN creates a "map", indicating distances from ISPs and CDN nodes
- when a DNS query arrives at authoritative DNS server:
  - server determines ISP from which query originates
  - uses "map" to determine a nearby CDN server

Summary
- Three means:
  - By content providers - replicated, distributed Internet sites
  - By content consumers (or clients) - Web caching
  - By a third-party - content distribution network
- Scalability
  - rank them on scalability
- Content staleness
  - rank them on content staleness
- Transparency
  - rank them on transparency to clients