CDNs (Content Delivery Networks) (§7.5.3)
Topic

- CDNs (Content Delivery Networks)
  - Efficient distribution of popular content; faster delivery for clients
Context

• As the web took off in the 90s, traffic volumes grew and grew. This:
  1. Concentrated load on popular servers
  2. Led to congested networks and need to provision more bandwidth
  3. Gave a poor user experience

• Idea:
  – Place popular content near clients
  – Helps with all three issues above
Before CDNs

- Sending content from the source to 4 users takes $4 \times 3 = 12$ “network hops” in the example.
After CDNs

- Sending content via replicas takes only $4 + 2 = 6$ “network hops”
After CDNs (2)

- Benefits assuming popular content:
  - Reduces server, network load
  - Improves user experience (PLT)
Popularity of Content

- Zipf’s Law: few popular items, many unpopular ones; both matter

Zipf popularity ($k$th item is $1/k$)


George Zipf (1902-1950)
How to place content near clients?

• Use browser and proxy caches
  – Helps, but limited to one client or clients in one organization

• Want to place replicas across the Internet for use by all nearby clients
  – Done by clever use of DNS
Content Delivery Network

CDN origin server

CDN node

Page fetch

Sydney

Boston

Amsterdam

Distribution to CDN nodes

Worldwide clients
Content Delivery Network (2)

- DNS resolution of site gives different answers to clients
  - Tell each client the site is the nearest replica (map client IP)
Business Model

- Clever model pioneered by Akamai
  - Placing site replica at an ISP is win-win
  - Improves site experience and reduces bandwidth usage of ISP
What are companies doing with networking?
QUIC

• Why?

• What does it do?
QUIC: Why?

- Changing TCP is a pain and unlikely to happen

- If you can control the client/server then implement your own reliability over UDP
QUIC: What?

- Google controls the Chrome browser, Search/Youtube server

- QUIC is a transport layer protocol implemented over UDP
QUIC: What does it do?

- lower-latency connection establishment
- improved congestion control and better loss recovery
Zero-latency connection establishment

**TCP**
- Sender
- Receiver
- 100 ms

**TCP + TLS**
- Sender
- Receiver
- 200 ms
- 300 ms

**QUIC** (equivalent to TCP + TLS)
- Sender
- Receiver
- 0 ms
- 100 ms

1. Repeat connection
2. Never talked to server before
Improved congestion control and better loss recovery

• Packet sequence numbers are never reused → No confusion

• Use NACK to explicitly tell the sender which packets are lost

• Retransmit lost packets faster at the sender
Improved congestion control and better loss recovery

- QUIC outshines TCP under poor network conditions, shaving a full second off the Google Search page load time for the slowest 1% of connections.
- Users report 30% fewer rebuffers when watching videos over QUIC.
- In 2015, roughly half of all requests from Chrome to Google servers are served over QUIC.