Content Delivery Networks

Content in today's Internet

- Most flows are HTTP
  - Web is at least 80% of traffic
  - Median object size is 2.2MB (as of 2022)
- HTTP uses TCP, so it will
  - Be ACK clocked
  - For Web, likely never leave slow start
- Is the Internet designed for this common case?

Evolution of Serving Web Content

- In the beginning…
  - …there was a single server
  - How to serve video stream to large number of users
- Issues with this model
  - Site reliability
    - Unplugging cable, hardware failure, natural disaster
  - Network congestion along long network path
  - Scalability
    - Flash crowds

Replicated Web service

- Use multiple servers

- Advantages
  - Better scalability
  - Better reliability
- Disadvantages
  - How do you decide which server to use?
  - How to do synchronize state among servers?

Load Balancers

- Device that multiplexes requests across a collection of servers
  - All servers share one public IP
  - Balancer transparently directs requests to different servers
- How should the balancer assign clients to servers?
  - Random / round-robin
  - Load-based
- Challenges
  - Scalability (must support traffic for n hosts)
  - State (must keep track of previous decisions)

Load balancing: Are we done?

- Advantages
  - Allows scaling of hardware independent of IPs
  - Relatively easy to maintain
- Disadvantages
  - Expensive
  - Still a single point of failure
  - Location!

Popping up: HTTP performance

- For Web pages
  - RTT matters most
  - Where should the server go?
- For video
  - Available bandwidth matters most
  - Where should the server go?
- Is there one location that is best for everyone?

Why speed matters

- Impact on user experience
  - Users navigating away from pages
  - Video startup delay
- Impact on revenue
  - Amazon: increased revenue 1.5% for every 100ms reduction in PLT
  - Google:21% increase in revenue by reducing PLT from 6 seconds to 1.2 seconds
- Why can't Internet help?
  - Peering point congestion
  - Inefficient routing
  - Unreliable networks
  - Inefficient communication protocols

Strawman solution: Web caches

- ISP uses a middlebox that caches Web content
  - Better performance – content is closer to users
  - Lower cost – content traverses network boundary once
  - Does this solve the problem?

- No!
  - Size of all Web content is too large
    - Zipf distribution limits cache hit rate
  - Web content is **dynamic** and **customized**
    - Can't cache banking content
    - What does it mean to cache search results?

Distance vs Throughput

Distance (Server to User)	Network RTT	Typical Packet Loss	Throughput	4GB DVD Download Time
Local: <100 mi.	1.6 ms	0.6%	44 Mbps (high quality HDTV)	12 min.
Regional: 500–1,000 mi.	16 ms	0.7%	4 Mbps (basic HDTV)	2.2 hrs.
Cross-continent: ~3,000 mi.	48 ms	1.0%	1 Mbps (SD TV)	8.2 hrs.
Multi-continent: ~6,000 mi.	96 ms	1.4%	0.4 Mbps (poor)	20 hrs

Goals of a CDN

- Primary Goals
  - Create replicas of content throughout the Internet
  - Ensure that replicas are always available
  - Direct clients to replicas that will give good performance

Examples of CDNs

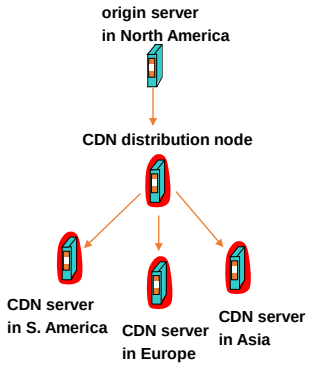
- Akamai
  - 360K servers, 1300+ networks, 135 countries
- Limelight
  - Well provisioned delivery centers, interconnected via a private fiber-optic connected to 700+ access networks
- Edgecast
  - 30+ PoPs, 5 continents, 2000+ direct connections
- Others
  - Google, Facebook, AWS, AT&T, Level3, Brokers

Akamai Statistics

- Distributed servers
  - Servers: ~151,000
  - Networks: ~1,200
  - Countries: ~92
- Many customers
  - Apple, BBC, FOX, GM IBM, MTV, NASA, NBC, NFL, NPR, Puma, Red Bull, Rutgers, SAP, …
- Client requests
  - Hundreds of billions per day
  - Half in the top 45 networks
  - 15-20% of all Web traffic worldwide
- Applications
  - Live streaming
  - Social networks
  - Protection against DDoS attack
  - Handle flash crowd

Content Delivery Network

- Proactive content replication
  - Content provider (e.g., CNN) contracts with a CDN
- CDN replicates the content
  - On many servers spread throughout the Internet
- Updating the replicas
  - Updates pushed to replicas when the content changes



Key Components of a CDN

- Distributed servers
  - Usually located inside of other ISPs
  - Often located in IXPs
- High-speed overlay network connecting them
- Clients
  - Can be located anywhere in the world
  - They want fast content delivery
- Glue
  - Something that binds clients to “nearby” replica servers

Elements of CDN

- Caching and Replacement of Content (Resilient Overlay)
- Content/Object Location (Consistent Hashing-based)
- Request Redirection (DNS-based, HTTP-based)
- Server Selection (Load, Nearest, Availability)
- Monitoring Performance and Logging

Inside a CDN

- Servers are deployed in clusters for reliability
  - Some may be offline
    - Could be due to failure
    - Also could be "suspended" (e.g., to save power or for upgrade)
- Could be multiple clusters per location (e.g., in multiple racks)
- Server locations
  - Well-connected points of presence (PoPs)

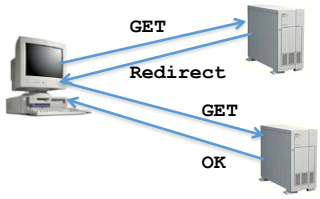
Server Selection Policy

- Live server
  - For availability
- Lowest load
  - To balance load across the servers
- Closest
  - Nearest geographically, or in round-trip time
- Best performance
  - Throughput, latency, …
- Cheapest bandwidth, electricity, …

Requires continuous monitoring of liveness, load, and performance

Server Selection Mechanism

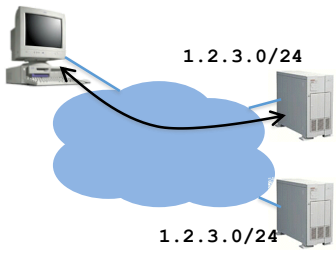
- Application
  - HTTP redirection



- Advantages
  - Fine-grained control
  - Selection based on client IP address
- Disadvantages
  - Extra round-trips for TCP connection to server
  - Overhead on the server
  - Implementation issues in browsers

Server Selection Mechanism

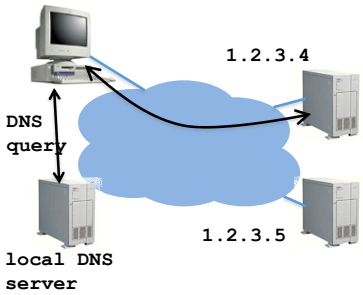
- Routing
  - Anycast routing



- Advantages
  - No extra round trips
  - Route to nearby server
- Disadvantages
  - Does not consider network or server load
  - Different packets may go to different servers
  - Used only for simple request-response apps

Server Selection Mechanism

- Naming
  - DNS-based server selection



- Advantages
  - Avoid TCP set-up delay
  - DNS caching reduces overhead
  - Relatively fine control
- Disadvantage
  - Based on IP address of local DNS server
  - "Hidden load" effect
  - DNS TTL limits adaptation

How Akamai Works

- Clients fetch html document from primary server
  - E.g. fetch index.html from cnn.com
- URLs for replicated content are replaced in html
  - E.g. <img src="http://cnn.com/af/x.gif"> replaced by <img src="http://a73.g.akamaitech.net/7/23/cnn.com/af/x.gif">
- Client is forced to resolve aXYZ.g.akamaitech.net hostname

How Akamai Works

- Akamai only replicates static content
- Modified name contains original file name
- Akamai server is asked for content
  - First checks local cache
  - If not in cache, requests file from primary server and caches file

How Akamai Works

- Root server gives NS record for akamai.net
- Akamai.net name server returns NS record for g.akamaitech.net
  - Name server chosen to be in region of client's name server
  - TTL is large
- g.akamaitech.net nameserver chooses server in region
  - Should try to chose server that has file in cache - How to choose?
  - Uses aXYZ name and hash
  - TTL is small → why?

Mapping System

- Equivalence classes of IP addresses
  - IP addresses experiencing similar performance
  - Quantify how well they connect to each other
- Collect and combine measurements
  - Ping, traceroute, BGP routes, server logs
    - E.g., over 100 TB of logs per days
  - Network latency, loss, and connectivity

Mapping System

- Map each IP class to a preferred server cluster
  - Based on performance, cluster health, etc.
  - Updated roughly every minute
- Map client request to a server in the cluster
  - Load balancer selects a specific server
  - E.g., to maximize the cache hit rate

Adapting to Failures

- Failing hard drive on a server
  - Suspends after finishing “in progress” requests
- Failed server
  - Another server takes over for the IP address
  - Low-level map updated quickly
- Failed cluster
  - High-level map updated quickly
- Failed path to customer’s origin server
  - Route packets through an intermediate node

Akamai Transport Optimizations

- Bad Internet routes
  - Overlay routing through an intermediate server
- Packet loss
  - Sending redundant data over multiple paths
- TCP connection set-up/teardown
  - Pools of persistent connections
- TCP congestion window and round-trip time
  - Estimates based on network latency measurements

Akamai Application Optimizations

- Slow download of embedded objects
  - Prefetch when HTML page is requested
- Large objects
  - Content compression
- Slow applications
  - Moving applications to edge servers
  - E.g., content aggregation and transformation
  - E.g., static databases (e.g., product catalogs)
  - E.g. batching and validating input on Web forms