Today
- Web Caching
  - Replica Placement
  - Update Propagation

Web Caching
- Example of geo-distributed large-scale replication
- Web server is primary replica
- Web pages are cached at:
  - Server replicas
  - Client browsers
  - Proxy caches
  - Content-distributions networks (CDNs)
- What consistency model is typically used?

Web Caching: Issues
- Where are the replicas placed?
- How are updates sent to different replicas?
Replica Placement

- Permanent replicas: Server replication, mirroring
- Server-initiated replicas: Push caching, CDNs
- Client-initiated replicas: Proxies, client caches

Permanent Replicas

- Server replication
  - Multiple servers hosting a Web site
  - Typically hosted on a LAN/cluster
  - Requests are distributed among the servers: Front-end load balancer, Dynamic DNS mapping
- Mirroring
  - Popular Websites have multiple fixed servers
  - Users select one of the mirror sites
  - E.g.: http://www.kernel.org/mirrors

Server-Initiated Replicas

- Servers are responsible for pushing updates to replicas
- Content-Distribution Networks (CDNs)
  - Statically placed geographically-distributed replicas
  - Each replica typically holds static content
  - Users are directed to closest replica
  - Example: Akamai
- Dynamic replication
  - Determine placement of server/files based on request traffic, location of clients

Client-Initiated Replicas

- Client caches:
  - Local cache maintained by the browser
- Proxy caches:
  - Cache shared by multiple browsers/users
  - Can be placed at multiple levels: On a LAN, department-wide, institute-wide
- Cooperative caches:
  - Each cache holds subset of Web pages
  - Share pages among each other
  - Can have hierarchy or overlay network of proxies
Update Propagation

- What happens when Web page changes?
- Who initiates the updates?
  - Push vs. pull
- What is sent in the updates?
  - Notification vs. new data

Push-based Updates

- Server sends updates to proxies whenever a Web page changes
- How to send updates?
  - Invalidation: Notification of change, actual data will be downloaded by the proxy
  - Updates: May be useful if frequency of updates is high and data needs to be fresh
  - Hybrid: Send updates for more time-sensitive data, invalidation for rest

Push-based Updates: Pros and Cons

- Pros:
  - Maintains high consistency
  - Proxies can be passive/dumb
  - Useful for time-sensitive data
- Cons:
  - Lot of state at the server
    - Negates the usefulness of HTTP (stateless)
    - Susceptible to server crashes
  - Too much network bandwidth for updates

Pull-based Updates

- Proxy pulls in updates from the server
- Proxy polls the server periodically
  - Get fresh data if changed
- What should be the polling frequency?
  - Client-initiated: Poll server whenever a client makes a request
  - Time-To-Live: Web Page may have a TTL, after which proxy needs to poll
  - Adaptive: Maintain a history of update times to estimate TTL for different pages
Pull-based Updates: Pros and Cons

Pros:
- Server remains stateless
- Can be implemented using HTTP (If-modified-since)
- Resilient to both server and proxy failures
- Demand-driven

Cons:
- Weaker consistency guarantees
- Active proxies required (not passive caches)
- High overhead if high polling frequency
- Response time higher on misses

Leases

- Hybrid approach between push and pull
- Lease: Limited time during which server pushes updates to proxy
- Lease expiry: Proxy has to pull updates or renew lease
- Generalization of push-pull approaches:
  - What is the lease duration for pure push and pure pull approaches?

Types of Leases

- Age-based
  - Larger the expected lifetime, longer the lease
  - Expected lifetime based on last modification time
- Renewal-frequency based
  - More frequently a proxy accesses a page, the longer its lease
- Server load based
  - Shorter leases during heavy load