Cookies and Sessions

Lecture Notes for CS 142
Fall 2010
John Ousterhout

- Readings for this topic: none.

Stateless applications

- Web application servers are generally "stateless":
    - Each HTTP request is independent; server can't tell if 2 requests came from the same browser or user.
    - Web server applications maintain no information in memory from request to request (only information on disk survives from one request to another).
- Statelessness not always convenient for application developers: need to tie together a series of requests from the same user.

Browser cookies

- Cookie basics:
    - The first time a browser connects with a particular server, there are no cookies.
    - When the server responds it includes a `Set-Cookie:` header that defines a cookie.
    - Each cookie is just a name-value pair.
    - In the future whenever the browser connects with the same server, it includes a `Cookie:` header containing the name and value, which the server can use to connect related requests.
- What's in a cookie?
    - Name and data.
        - Data size limited by browsers (typically < 4 KB).
        - A server can define multiple cookies with different names, but browsers limit the number of cookies per server (around 50).
    - Domain for this cookie: server, port (optional), URL prefix (optional). The cookie is only included in requests matching its domain.
    - Expiration date: browser can delete old cookies.

Sessions

- Cookies are used by the server to implement *sessions*:
    - A pool of data related to an active connection (one browser instance).
- Typically the cookie for an application contains an identifier for a session.

- Web frameworks like Rails do most of the work of managing sessions and cookies:
    - Rails provides `session`, a hash-like object in which you can store anything you like
        - Data will be available in all future requests from the same browser.
    - Rails automatically checks for a session cookie at the start of each request:
        - Cookie exists? use it to find session data

        - No cookie? Create new session, new cookie
    - End of each request: save session data where it can be found by future requests.

- Managing session state:
    - Approach #1: just keep state in main memory
    - Approach #2: store session state in files on disk
    - Approach #3: store session state in a database
    - Most frameworks allow you to control session storage:
        - Provide an object that saves and restores session data.
- Server must eventually delete stale session data.

- Sessions have numerous security issues, which we will discuss later.