Saving State on the WWW

The Issue

➢ Connections on the WWW are stateless

➢ Every time a link is followed is like the first time to the server — it has no memory for connections

---

Why Bother To Fix This?

By saving state we can…

- Save configuration information between sessions
- Make adaptive websites (change themselves to suit user's behaviour)
- Enable e-commerce applications (shopping carts)

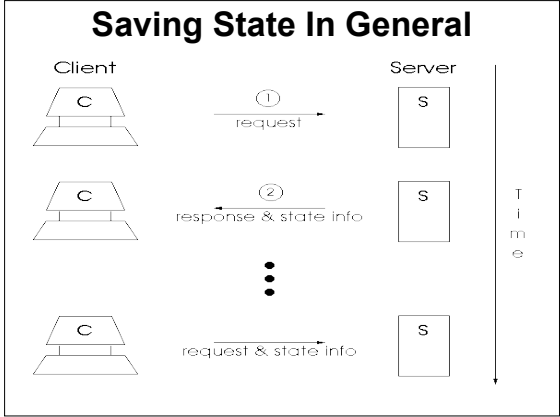- Violate users' privacy by tracking which websites and webpages they visit

Saving State In General

Methods of Saving State

➢ Cookies
➢ Session-level authentication
➢ `form`s
➢ URL Rewriting

Method 1: Cookies

➢ Basic idea
  • Client stores data for the server
  • Client sends data to server with each request
➢ Details (Version 0)
  • Required fields: *name=value*
  • Optional fields: domain, path, secure, expires
  • Size: maximum 4 kilobytes
  • Number: maximum 1024 cookies

Aside: Cookie Concept

- Cookie is a computing term from long ago.
- According to *The New Hacker's Dictionary*:
  - Something passed between subroutines or programs that enables the receiver to do something useful
  - The thing being passed is opaque to the sender (e.g. `time_t` type C libraries use)
  - Cookies are also small

- 'The phrase "it hands you a magic cookie" means it returns a result whose contents are not defined but which can be passed back to the same or some other program later.' [source for quote at end]

Cookie Examples

Examples at course website

- `time1.cgi` vs. `time2.cgi`
  Compare form method with cookie method

- `cookie-colour`
  - One program to write cookies
  - One program to read cookies
  - Use `env.cgi` to see cookies in headers

Method 2:
Session-level Authentication

- See §12.2 (Basic Authentication) in *HTTP: The Definitive Guide* by David Gourley & Brian Totty, © 2002 by O'Reilly & Associates, Inc. (ISBN: 1-56592-509-2)

- Session (from ISO Reference Model)
  - Logical communication between two network end points
  - Sessions are composed of requests and responses that occur between applications in different network hosts.
  - In browser terms a session is the longevity of the O/S process

The Steps of Basic Authentication

1. Browser requests resource from server application
   - usually with GET protocol

2. Server replies with code 401 (authorization required)

3. Browser prompts user
   - for name & password

4. Browser resends request including the name & password (in the network header)
   - Every time the browser makes a request for that resource it will send the name & password, until the end of the session
   - The name & password are like a cookie that is stored in RAM
   - Because they are in RAM they will be forgotten when the browser quits (i.e., at the end of the session)

Method 3:
`forms` with hidden fields

➤ We usually pull webpages in from a server

➤ Forms are for pushing data to the server

➤ To use forms we need to use CGI protocol

- CGI = common gateway interface
- An application layer protocol that allows client to send data to the server

---

Two `form` Methods

`method="get"`	`method="post"`
➤Data is part of URL	➤Data is <u>not</u> part of URL
➤Only used for simple requests (e.g. search engine queries)	➤Can be used for file upload
➤Conceptually: a query of a database	➤Conceptually: an alteration to a database

See <u>`form` examples</u> online

Saving State With `forms`

- Hidden post & simple get
- Did you see the hidden field?
- Did you see the hidden data?

- That's one way of saving state:

  Placing the data in a `form` so that every time the form is submitted (sent to the server) the data is sent too

  - Examples using CGI program to generate a `form`
    - Loan.cgi and multi-page.cgi

Essence of State Saving Using `forms`

- There must be an uninterrupted sequence of request/responses pairs from the browser to a CGI program (or programs)
- The state must be
  - represented in the `form`, and
  - recognized by the CGI program(s)
- The CGI program(s) must encode the state in the `form`

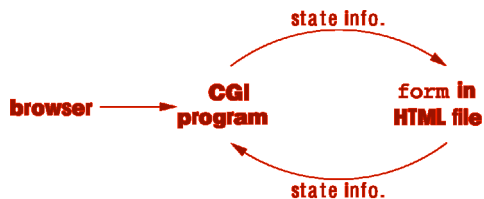## Essence of State Saving Using `forms`

state info.

browser → CGI program → form in HTML file

state info.

This diagram is for a single CGI program and a single form, but the same thing could be done with multiple programs & forms

Method 4:
Servlets & URL Rewriting

➢ Recall that `method="get"` `form`s pass their data in the URL

➢ These URLS are designed to be cached
  • You don't need a browser that can understand `form`s to use them
  • You can just type them in like any other web address

URL Rewriting Explained

So why not put the state information from `method="get"` `form`s in the `href` of every anchor

```
Instead of
<a href="foo.html"
   >click here</a>
do
<a href="foo.html?session=…"
   >click here</a>
```

Servlets (1 of 2)

➢ Many users dislike long URLs — they are hard to mail to friends & look ugly
➢ Some browser software don't support cookies — and many users have such support disabled
➢ Wouldn't it be great if your server would use cookies when the client supported them, and URL rewriting when it didn't?

Servlets (2 of 2)

➤ Servlets (and other server-side software) do that!
➤ Servlets are server-side programs written in Java
➤ Other server-side technologies work the same way but are implemented in other languages

See also
Servlets
lecture

What Data Do We Pass?

➤ But isn't that a lot of state information to send back and forth?

➤ Not really, because we don't have to pass all of the data back and forth
➤ We can pass a user or session ID and the server will maintain a database keyed by those IDs

Resources

➤ Cookie resources at course website
➤ *HTTP: The Definitive Guide* in the e-book collection
➤ Cookie examples at course website
➤ SessionTrack servlet example at course website
   ☛ Note that servlets are not always running at FCS

Bibliography

1. CS4173 Resource List
2. *Web Protocols and Practice*
   Balachander Krishnamurthy & Jennifer Rexford
   Addison-Wesley May 2001
   © 2001 by AT&T Corp. (ISBN 0-201-71088-9)
3. *HTTP: The Definitive Guide*
   David Gourley & Brian Totty, with others
   O'Reilly & Assoc. Sept. 2001
   (ISBN 1-56592-509-2)
4. The New Hacker's Dictionary
   Eric S. Raymond (editor)
   Online version used at
   <URL:http://www.logophilia.com/jargon/jargon_28.html#TAG1093>