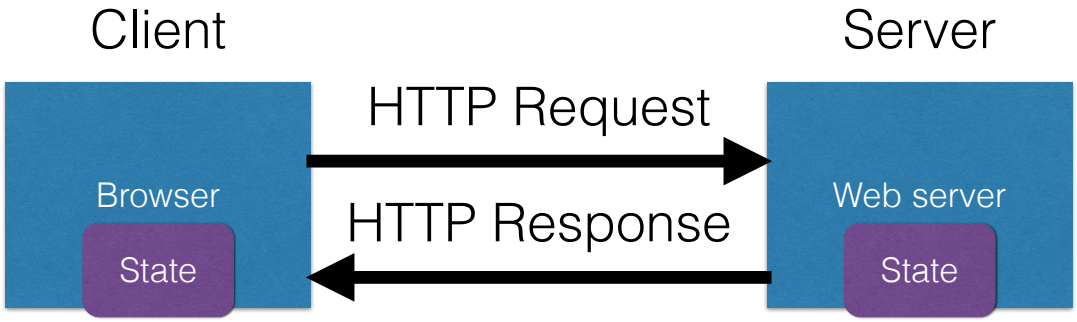
Web-based State using Hidden Fields and Cookies

HTTP is *stateless*

- The lifetime of an HTTP session is typically:
  - Client connects to the server
  - Client issues a request
  - Server responds
  - Client issues a request for something in the response
  - …. repeat ….
  - Client disconnects
- HTTP has no means of noting “oh this is the same client from that previous session”
  - *How is it you don’t have to log in at every page load?*

Maintaining State



- **Web application maintains *ephemeral* state**
  - Server processing often produces intermediate results
    - Not ACID, long-lived state
  - **Send** such **state to the client**
  - Client **returns the state** in subsequent **responses**

Two kinds of state: **hidden fields**, and **cookies**

Ex: Online ordering

socks.com/order.php

Order

$5.50

Order

socks.com/pay.php

Pay

The total cost is $5.50.
Confirm order?

Yes

No

Separate page

Ex: Online ordering

What's presented to the user

pay.php

```
<html>
<head> <title>Pay</title> </head>
<body>

<form action="submit_order" method="GET">
The total cost is $5.50. Confirm order?
<input type="hidden" name="price" value="5.50">
<input type="submit" name="pay" value="yes">
<input type="submit" name="pay" value="no">

</body>
</html>
```

Ex: Online ordering

The corresponding backend processing

```
if(pay == yes && price != NULL)
{
  bill_creditcard(price);
  deliver_socks();
}
else
  display_transaction_cancelled_page();
```

Ex: Online ordering

What's presented to the user

```
<html>
<head> <title>Pay</title> </head>
<body>

<form action="submit_order" method="GET">
The total cost is $5.50. Confirm order?
<input type="hidden" name="price" value="0.01">
<input type="submit" name="pay" value="yes">
<input type="submit" name="pay" value="no">

</body>
</html>
```

Client can change the value!

Solution: *Capabilities*

- **Server maintains *trusted* state** (while client maintains the rest)
  - Server stores intermediate state
  - Send a **capability** to access that state to the client
  - Client **references the capability** in subsequent responses
- **Capabilities should be large, random numbers**, so that they are hard to guess
  - To prevent illegal access to the state

Using capabilities

What's presented to the user

```
<html>
<head> <title>Pay</title> </head>
<body>

<form action="submit_order" method="GET">
The total cost is $5.50. Confirm order?
<input type="hidden" name="sid" value="781234">
<input type="submit" name="pay" value="yes">
<input type="submit" name="pay" value="no">

</body>
</html>
```

Capability; the system will detect a change and abort

Using capabilities

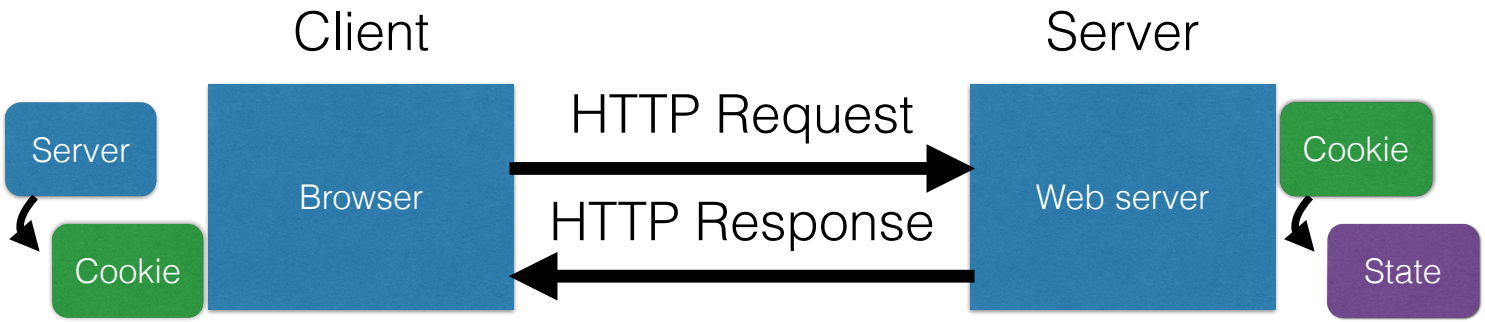
The corresponding backend processing

```
price = lookup(sid);
if(pay == yes && price != NULL)
{
  bill_creditcard(price);
  deliver_socks();
}
else
  display_transaction_cancelled_page();
```

But: we don't want to pass hidden fields around all the time

- Tedious to add/maintain on all the different pages
- Have to start all over on a return visit (after closing browser window)

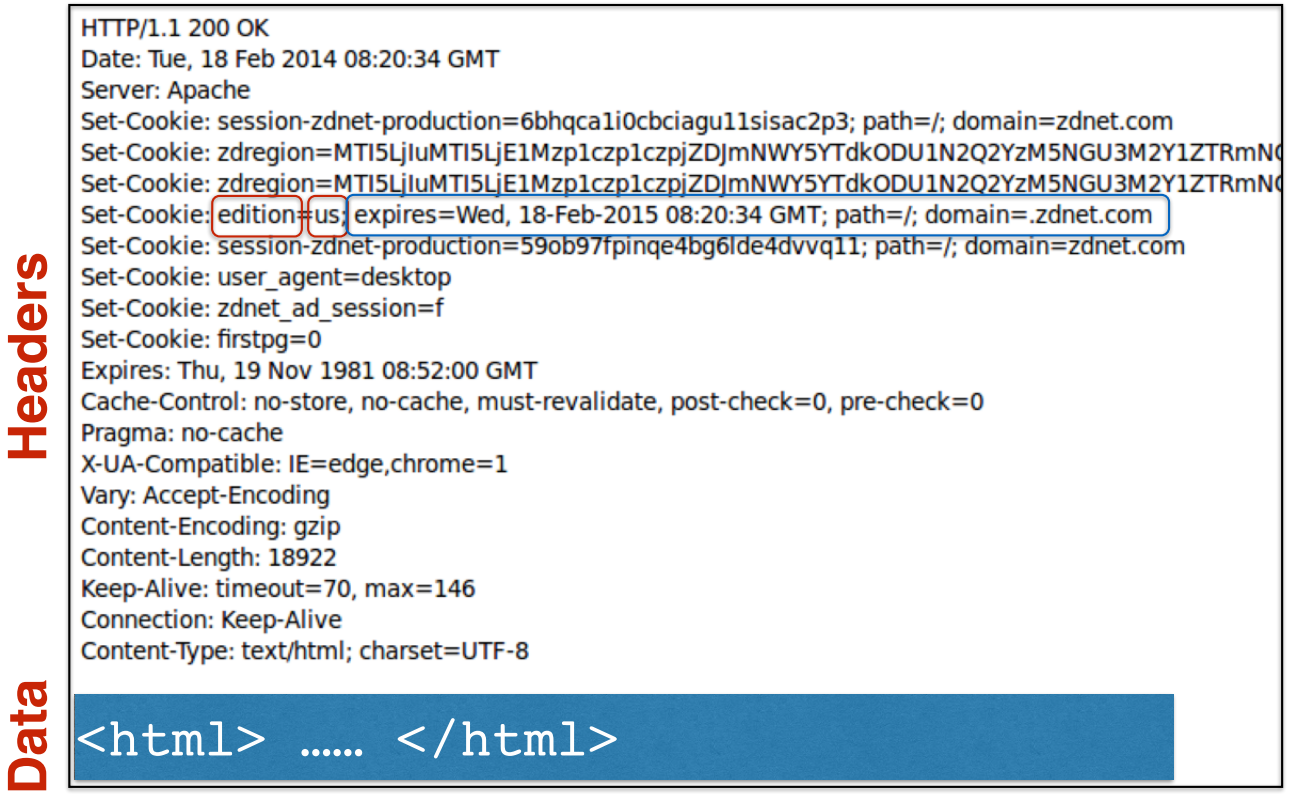
Statefulness with Cookies



- Server **maintains trusted state**
  - Server indexes/denotes state with a **cookie**
  - Sends cookie to the client, which stores it
  - Client returns it with subsequent queries to that same server

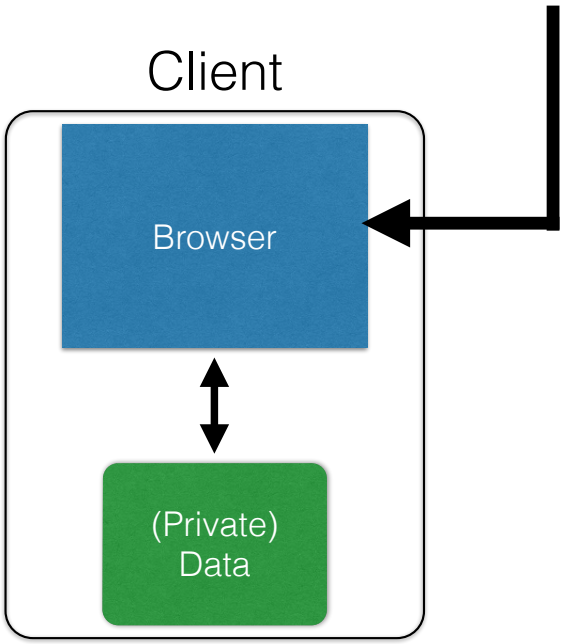
Cookies are key-value pairs

Set-Cookie:key=value; options; ….



Cookies

Set-Cookie: edition=us; expires=Wed, 18-Feb-2015 08:20:34 GMT; path=/; domain=.zdnet.com



Semantics

- Store “us” under the key “edition”
- This value is no good as of Wed Feb 18…
- This value should only be readable by any domain ending in `.zdnet.com`
- This should be available to any resource within a subdirectory of `/`
- Send the cookie with any future requests to `<domain>/<path>`

Requests with cookies

```
HTTP/1.1 200 OK
Date: Tue, 18 Feb 2014 08:20:34 GMT
Server: Apache
Set-Cookie: session-zdnet-production=6bhqca1i0cbciagu11sisac2p3; path=/; domain=zdnet.com
Set-Cookie: zdregion=MTI5LjIuMTI5LjE1Mzp1czp1czpjZDJmNWY5YTdkODU1N2Q2YzM5NGU3M2Y1ZTRmN
Set-Cookie: zdregion=MTI5LjIuMTI5LjE1Mzp1czp1czpjZDJmNWY5YTdkODU1N2Q2YzM5NGU3M2Y1ZTRmN
Set-Cookie: edition=us; expires=Wed, 18-Feb-2015 08:20:34 GMT; path=/; domain=.zdnet.com
Set-Cookie: session-zdnet-production=59ob97fpinqe4bg6lde4dvvq11; path=/; domain=zdnet.com
```

Subsequent visit

HTTP Headers

```
http://zdnet.com/

GET / HTTP/1.1
Host: zdnet.com
User-Agent: Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.9.2.11) Gecko/20101013 Ubuntu/9.04 (jaunty) Firefox/3.6.11
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Connection: keep-alive
Cookie: session-zdnet-production=59ob97fpinqe4bg6lde4dvvq11; zdregion=MTI5LjIuMTI5LjE1Mzp1czp1czpjZDJmNW ...
```

Why use cookies?

- **Session identifier**
  - After a user has authenticated, subsequent actions provide a cookie
  - So the user does not have to authenticate each time
- **Personalization**
  - Let an anonymous user customize your site
  - Store font choice, etc., in the cookie

Why use cookies?

- **Tracking users**
  - Advertisers want to know your behavior
  - Ideally build a profile *across different websites*
    - Visit the Apple Store, then see iPad ads on Amazon?!
  - How can site B know what you did on site A?

A shows you an ad from B and B scrapes the referrer URL

Option 1: B maintains a DB, indexed by your IP address

Problem: IP addrs change

Option 2: B maintains a DB indexed by a *cookie*

- **"Third-party cookie"**
- **Commonly used by large ad networks (doubleclick)**

http://live.wsj.com/video/how-advertisers-use-internet-cookies-to-track-you