WEB SECURITY:
WEB BACKGROUND

CMSC 414
FEB 20 2018
A very basic web architecture

DB is a separate entity, logically (and often physically)
A very basic web architecture

Browser

Web server

Database

(Private) Data

(Much) user data is part of the browser

DB is a separate entity, logically (and often physically)
Interacting with web servers

Get and put **resources** which are identified by a URL

http://www.cs.umd.edu/~dml/home.html
Interacting with web servers

Get and put *resources* which are identified by a URL

```
http://www.cs.umd.edu/~dml/home.html
```

**Protocol**

- ftp
- https
- tor
Interacting with web servers

Get and put resources which are identified by a URL

http://www.cs.umd.edu/~dml/home.html
Interacting with web servers

Get and put resources which are identified by a URL

http://www.cs.umd.edu/~dml/home.html

Hostname/server

Translated to an IP address by DNS
(more on this later)
Interacting with web servers

Get and put *resources* which are identified by a URL

http://www.cs.umd.edu/~dml/home.html
Interacting with web servers

Get and put *resources* which are identified by a URL

http://www.cs.umd.edu/~dml/home.html

Path to a resource

Here, the file home.html is *static content* i.e., a fixed file returned by the server
Interacting with web servers

Get and put resources which are identified by a URL

http://www.cs.umd.edu/~dml/home.html

Path to a resource

Here, the file home.html is static content i.e., a fixed file returned by the server

http://facebook.com/delete.php
Interacting with web servers

Get and put *resources* which are identified by a URL

http://www.cs.umd.edu/~dml/home.html

Path to a resource

Here, the file home.html is *static content* i.e., a fixed file returned by the server

http://facebook.com/delete.php

Path to a resource

Here, the file home.html is *dynamic content* i.e., the server generates the content on the fly
Interacting with web servers

Get and put *resources* which are identified by a URL

http://www.cs.umd.edu/~dml/home.html

Path to a resource

Here, the file home.html is *static content*
i.e., a fixed file returned by the server

http://facebook.com/delete.php

Here, the file home.html is *dynamic content*
i.e., the server generates the content on the fly
Interacting with web servers

Get and put resources which are identified by a URL

http://www.cs.umd.edu/~dml/home.html

Path to a resource

Here, the file home.html is static content i.e., a fixed file returned by the server

http://facebook.com/delete.php?f=joe123&w=16

Here, the file home.html is dynamic content i.e., the server generates the content on the fly
Interacting with web servers

Get and put resources which are identified by a URL

http://www.cs.umd.edu/~dml/home.html

Path to a resource

Here, the file home.html is static content
i.e., a fixed file returned by the server

http://facebook.com/delete.php?f=joe123&w=16

Arguments

Here, the file home.html is dynamic content
i.e., the server generates the content on the fly
Basic structure of web traffic

Client

Browser

(Private) Data

Server

Web server

Database
Basic structure of web traffic

Client

Browser

Server

Web server
Basic structure of web traffic

Client

Browser

HTTP

Web server

Server
Basic structure of web traffic

- HyperText Transfer Protocol (HTTP)
  - An “application-layer” protocol for exchanging collections of data
Basic structure of web traffic

Client

Browser

Server

Web server
Basic structure of web traffic

Client

Browser

Server

Web server

User clicks
Basic structure of web traffic

Client

Browser

HTTP Request

Server

Web server

User clicks
Basic structure of web traffic

- Requests contain:
  - The URL of the resource the client wishes to obtain
  - Headers describing what the browser can do

- Requests be GET or POST
  - **GET**: all data is in the URL itself (supposed to have no side-effects)
  - **POST**: includes the data as separate fields (can have side-effects)
HTTP GET requests

http://www.reddit.com/r/security

<table>
<thead>
<tr>
<th>HTTP Headers</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://www.reddit.com/r/security">http://www.reddit.com/r/security</a></td>
</tr>
<tr>
<td>GET /r/security HTTP/1.1</td>
</tr>
<tr>
<td>Host: <a href="http://www.reddit.com">www.reddit.com</a></td>
</tr>
<tr>
<td>User-Agent: Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.9.2.11) Gecko/20100113 Ubuntu/9.04 (jaunty) Firefox/3.6.11</td>
</tr>
<tr>
<td>Accept: text/html,application/xhtml+xml,application/xml;q=0.9,<em>/</em>;q=0.8</td>
</tr>
<tr>
<td>Accept-Language: en-us,en;q=0.5</td>
</tr>
<tr>
<td>Accept-Encoding: gzip, deflate</td>
</tr>
<tr>
<td>Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7</td>
</tr>
<tr>
<td>Keep-Alive: 115</td>
</tr>
<tr>
<td>Connection: keep-alive</td>
</tr>
</tbody>
</table>
HTTP GET requests

http://www.reddit.com/r/security
HTTP GET requests

http://www.reddit.com/r/security

User-Agent is typically a browser but it can be wget, JDK, etc.
Hacker Claims Feds Hit Him With 44 Felonies When He Refused to Be an FBI Spy

Lenovo Installed Adware on Computers that allows for MITM (SSL Cert Replacement)

Google Chrome Recorded the Highest Number of Vulnerabilities in January 2015

Chips under the skin: Biohacking, the connected body is 'here to stay'

IT Security career dilemma (self.security)
Hacker Claims Feds Hit Him With 44 Felonies When He Refused to Be an FBI Spy

Lenovo Installed Adware on Computers that allows for MITM (SSL Cert Replacement)

Google Chrome Recorded the Highest Number of Vulnerabilities in January 2015

Chips under the skin: Biohacking, the connected body is 'here to stay'

IT Security career dilemma (self.security)
Referrer URL: the site from which this request was issued.
HTTP POST requests

Posting on Piazza

Session cookie (more on this later). Not something you want to share!
HTTP POST requests

Posting on Piazza

<table>
<thead>
<tr>
<th>HTTP Headers</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="https://piazza.com/logic/api?method=content.create&amp;aid=i6ceq3skno48">https://piazza.com/logic/api?method=content.create&amp;aid=i6ceq3skno48</a></td>
</tr>
<tr>
<td>POST</td>
</tr>
<tr>
<td>User-Agent: Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.9.2.11) Gecko/20100113 Ubuntu/9.04 (jaunty) Firefox/3.6.11</td>
</tr>
<tr>
<td>Accept: application/json, text/javascript, <em>/</em>; q=0.01</td>
</tr>
<tr>
<td>Accept-Language: en-us,en;q=0.5</td>
</tr>
<tr>
<td>Accept-Encoding: gzip, deflate</td>
</tr>
<tr>
<td>Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7</td>
</tr>
<tr>
<td>Keep-Alive: 115</td>
</tr>
<tr>
<td>Connection: keep-alive</td>
</tr>
<tr>
<td>Content-Type: application/x-www-form-urlencoded; charset=UTF-8</td>
</tr>
<tr>
<td>X-Requested-With: XMLHttpRequest</td>
</tr>
<tr>
<td>Referer: <a href="https://piazza.com/class?id=i55texo54nv3eh">https://piazza.com/class?id=i55texo54nv3eh</a></td>
</tr>
<tr>
<td>Content-Length: 640</td>
</tr>
<tr>
<td>Cookie: piazza_session=</td>
</tr>
<tr>
<td>Pragma: no-cache</td>
</tr>
<tr>
<td>Cache-Control: no-cache</td>
</tr>
</tbody>
</table>

Session cookie (more on this later). Not something you want to share!
HTTP POST requests

Posting on Piazza

Implicitly includes data as a part of the URL

Session cookie (more on this later). Not something you want to share!
HTTP POST requests

Posting on Piazza

Explicitly includes data as a part of the request’s content

Implicitly includes data as a part of the URL

Session cookie (more on this later). Not something you want to share!
Basic structure of web traffic

Client

Browser

HTTP Request

User clicks

Server

Web server
Basic structure of web traffic

Client

Browser

User clicks

Server

Web server
Basic structure of web traffic

Client

Browser

HTTP Response

Server

Web server

User clicks
Basic structure of web traffic

User clicks

- Responses contain:
  - Status code
  - Headers describing what the server provides
  - Data
  - Cookies
    - State it would like the browser to store on the site’s behalf
HTTP responses

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=MTI5LjIuMTI5LjE1Mzp1czp1czp1ZDJmNY5YTdkODU1N2Q2YzM5NGU3M2Y1ZTRmNQ=
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
Set-Cookie: user_agent=desktop
Set-Cookie: zdnet_ad_session=f
Set-Cookie: firstpg=0
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
X-UA-Compatible: IE=edge,chrome=1
Vary: Accept-Encoding
Content-Encoding: gzip
Content-Length: 18922
Keep-Alive: timeout=70, max=146
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8

<html> ...... </html>
HTTP responses

HTTP version
Status code
Reason phrase

HTTP/1.1 200 OK

Date: Tue, 18 Feb 2014 08:20:34 GMT
Server: Apache
Set-Cookie: session-zdnet-production=6bhqca1i0cbbc1agu11sisac2p3; path=/; domain=zdnet.com
Set-Cookie: zdregion=MTI5LjEuMTI5LjE1Mzptc3pzc3pZDJmNWY5YTdkODU1N2Q2YzM5NGU3M2Y1ZTRmNQ
Set-Cookie: zdregion=MTI5LjEuMTI5LjE1Mzptc3pzc3pZDJmNWY5YTdkODU1N2Q2YzM5NGU3M2Y1ZTRmNQ
Set-Cookie: edition=us; expires=Wed, 18-Feb-2015 08:20:34 GMT; path=/; domain=.zdnet.com
Set-Cookie: session-zdnet-production=59ob97fpinq4e4bg6ide4dvq11; path=/; domain=zdnet.com
Set-Cookie: user_agent=desktop
Set-Cookie: zdnet_ad_session=f
Set-Cookie: firstpg=0
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
X-UA-Compatible: IE=edge,chrome=1
Vary: Accept-Encoding
Content-Encoding: gzip
Content-Length: 18922
Keep-Alive: timeout=70, max=146
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8

<html> ...... </html>
X-nananana: Batcache
X-hacker: If you're reading this, you should visit automattic.com/jobs and apply to join the fun, mention this header.

X-nananana: Batcache
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”

- *With this alone, you’d have to log in at every page load*
Maintaining state across HTTP sessions

- Server processing results in intermediate state
- Send the state to the client in *hidden fields*
- Client returns the state in subsequent responses
Maintaining state across HTTP sessions

- Server processing results in intermediate state
- Send the state to the client in *hidden fields*
- Client returns the state in subsequent responses
Maintaining state across HTTP sessions

- Server processing results in intermediate state
- Send the state to the client in *hidden fields*
- Client returns the state in subsequent responses
Maintaining state across HTTP sessions

- Server processing results in intermediate state
- Send the state to the client in *hidden fields*
- Client returns the state in subsequent responses
Maintaining state across HTTP sessions

- Server processing results in intermediate state
- Send the state to the client in *hidden fields*
- Client returns the state in subsequent responses
Maintaining state across HTTP sessions

- Server processing results in intermediate state
- Send the state to the client in hidden fields
- Client returns the state in subsequent responses
Maintaining state across HTTP sessions

- Server processing results in intermediate state
- Send the state to the client in *hidden fields*
- Client returns the state in subsequent responses
Maintaining state across HTTP sessions

- Server processing results in intermediate state
- Send the state to the client in *hidden fields*
- Client returns the state in subsequent responses
Maintaining state across HTTP sessions

- Server processing results in intermediate state
- Send the state to the client in *hidden fields*
- Client returns the state in subsequent responses
Maintaining state across HTTP sessions

- Server processing results in intermediate state
- Send the state to the client in *hidden fields*
- Client returns the state in subsequent responses
Online ordering

socks.com

Order

$5.50
Online ordering

socks.com

Order

$5.50

socks.com

Pay

The total cost is $5.50. Confirm order?

Yes  No

Separate page
Online ordering

What’s presented to the user

```html
<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>
```
Online ordering

What’s presented to the user

```html
<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>
```
Online ordering

The corresponding backend processing

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

The corresponding backend processing

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

What’s presented to the user

```html
<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">  

</form>  
</body>  
</html>
```
Online ordering

What’s presented to the user

```html
<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>
```
Minimizing trust in the client

What’s presented to the user

```html
<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">

</form>
</body>
</html>
```
Minimizing trust in the client

What’s presented to the user

```html
<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">

</form>
</body>
</html>
```
The corresponding backend processing

```plaintext
price = lookup(sid);
if(pay == yes && price != NULL)
{
    bill_creditcard(price);
    deliver_socks();
}
else
    display_transaction_cancelled_page();
```
Minimizing trust in the client

The corresponding backend processing

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

We don’t want to pass hidden fields around all the time
Statefulness with Cookies

- Server stores state, indexes it with a cookie
- Send this cookie to the client
- Client stores the cookie and returns it with subsequent queries to that same server
Statefulness with Cookies

- Server stores state, indexes it with a cookie
- Send this cookie to the client
- Client stores the cookie and returns it with subsequent queries to that same server
Statefulness with Cookies

- Server stores state, indexes it with a cookie
- Send this cookie to the client
- Client stores the cookie and returns it with subsequent queries to that same server
Statefulness with Cookies

- Server stores state, indexes it with a cookie
- Send this cookie to the client
- Client stores the cookie and returns it with subsequent queries to that same server
Statefulness with Cookies

- Server stores state, indexes it with a cookie
- Send this cookie to the client
- Client stores the cookie and returns it with subsequent queries to that same server
Statefulness with Cookies

- Server stores state, indexes it with a cookie
- Send this cookie to the client
- Client stores the cookie and returns it with subsequent queries to that same server
Statefulness with Cookies

- Server stores state, indexes it with a cookie
- Send this cookie to the client
- Client stores the cookie and returns it with subsequent queries to that same server
Statefulness with Cookies

- Server stores state, indexes it with a cookie
- Send this cookie to the client
- Client stores the cookie and returns it with subsequent queries to that same server
Statefulness with Cookies

- Server stores state, indexes it with a cookie
- Send this cookie to the client
- Client stores the cookie and returns it with subsequent queries to that same server
Statefulness with Cookies

- Server stores state, indexes it with a cookie
- Send this cookie to the client
- Client stores the cookie and returns it with subsequent queries to that same server
Statefulness with Cookies

- Server stores state, indexes it with a cookie
- Send this cookie to the client
- Client stores the cookie and returns it with subsequent queries to that same server
Cookies are key-value pairs

Set-Cookie: key=value; options; ....

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=MTI5LjIuMTI5LjE1Mzp1czp1czjpZDJmNWY5YTkOUD1N2Q2YzM5NGU3M2Y1ZTRmNc
Set-Cookie: zdregion=MTI5LjIuMTI5LjE1Mzp1czp1czjpZDJmNWY5YTkOUD1N2Q2YzM5NGU3M2Y1ZTRmNc
Set-Cookie: edition=us; expires=Wed, 18-Feb-2015 08:20:34 GMT; path=/; domain=.zdnet.com
Set-Cookie: session-zdnet-production=590b97fpinqe4bg6de4dvq11; path=/; domain=zdnet.com
Set-Cookie: user_agent=desktop
Set-Cookie: zdnet_ad_session=f
Set-Cookie: firstpg=0
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
X-UA-Compatible: IE=edge,chrome=1
Vary: Accept-Encoding
Content-Encoding: gzip
Content-Length: 18922
Keep-Alive: timeout=70, max=146
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8

<html> ...... </html>
Cookies are key-value pairs

Set-Cookie: key=value; options; ....

HTTP/1.1 200 OK
Date: Tue, 18 Feb 2014 08:20:34 GMT
Server: Apache

Set-Cookie: session-znet-production=6bhqca1lobciaigu11sisac2p3; path=/; domain=znet.com
Set-Cookie: zdregion=MTI5LjIuMTI5LjE1Mzp1czp1czp1zdJmNYW5YTdkODU1N2Q2YzMsNGU3M2Y1ZTRmN0
Set-Cookie: zdregion=MTI5LjIuMTI5LjE1Mzp1czp1czp1zdJmNYW5YTdkODU1N2Q2YzMsNGU3M2Y1ZTRmN0
Set-Cookie: edition=fusl expires=Wed, 18-Feb-2015 08:20:34 GMT; path=/; domain=znet.com
Set-Cookie: session-znet-production=590b97fipinque4bg6ide4dvvq11; path=/; domain=znet.com
Set-Cookie: user_agent=desktop
Set-Cookie: znet_ad_session=f
Set-Cookie: firstpg=0
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
X-UA-Compatible: IE=edge,chrome=1
Vary: Accept-Encoding
Content-Encoding: gzip
Content-Length: 18922
Keep-Alive: timeout=70, max=146
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8

<html> ...... </html>
Cookies

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

Client

Browser

(Private) Data

Semantics
Cookies

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

Client

Browser

(Private) Data

Semantics

- Store “us” under the key “edition” (think of it like one big hash table)
Cookies

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

Client

Browser

(Private) Data

Semantics

• Store “us” under the key “edition” (think of it like one big hash table)

• This value is no good as of Wed Feb 18…
Cookies

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

Client

Browser

(Private) Data

Semantics

- Store “us” under the key “edition” (think of it like one big hash table)
- This value is no good as of Wed Feb 18…
- This value should only be readable by any domain ending in .zdnet.com
Cookies

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

Client

Browser

(Private) Data

Semantics

- Store “us” under the key “edition” (think of it like one big hash table)
- 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 /
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” (think of it like one big hash table)
- 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 to any future requests to <domain>/<path>
Cookies

Semantics

- Store “us” under the key “edition” (think of it like one big hash table)

- 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 to 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=6bhqca1i0cbciagu1sisac2p3; path=/; domain=zdnet.com
Set-Cookie: zdregion=MTI5LjIuMTI5LjE1Mzp1czp1czp1ZDJmNjQ5YTk0ODU1N2Q2YzM5NGU3M2Y1ZTRmNQ
Set-Cookie: zdregion=MTI5LjIuMTI5LjE1Mzp1czp1czp1ZDJmNjQ5YTk0ODU1N2Q2YzM5NGU3M2Y1ZTRmNQ
Set-Cookie: edition=us; expires=Wed, 18-Feb-2015 08:20:34 GMT; path=/; domain=.zdnet.com
Set-Cookie: session-zdnet-production=59ob97fpinge4bg6lde4dvwq11; path=/; domain=zdnet.com

Subsequent visit
Requests with cookies

Response

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=MTI5LjUuMTI5LjE1Mzp1czp1czp1ZDJmNWY5YTkODU1N2Q2YZM5NGU3M2Y1ZTRmN
Set-Cookie: zdregion=MTI5LjUuMTI5LjE1Mzp1czp1czp1ZDJmNWY5YTkODU1N2Q2YZM5NGU3M2Y1ZTRmN
Set-Cookie: edition=us; expires=Wed, 18-Feb-2015 08:20:34 GMT; path=/; domain=.zdnet.com
Set-Cookie: session-zdnet-production=59ob97fpinge4bg6lde4dvwq11; path=/; domain=zdnet.com

Subsequent visit
Requests with cookies

HTTP/1.1 200 OK
Date: Tue, 18 Feb 2014 08:20:34 GMT
Server: Apache
Set-Cookie: session-zdnet-production=6bhqcaii0cbciagu11sisac2p3; path=/; domain=zdnet.com
Set-Cookie: zdregion=MTI5LjIuMTI5LjE1Mzp1czp1czp1ZDJmNWY5YTdkODU1N2Q2YzM5NGU3M2Y1ZTRmNQ==
Set-Cookie: zdregion=MTI5LjIuMTI5LjE1Mzp1czp1czp1ZDJmNWY5YTdkODU1N2Q2YzM5NGU3M2Y1ZTRmNQ==
Set-Cookie: edition=us; expires=Wed, 18-Feb-2015 08:20:34 GMT; path=/; domain=.zdnet.com
Set-Cookie: session-zdnet-production=59ob97fpinge4bq6lde4dvq11; path=/; domain=zdnet.com

---

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/20100113 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=59ob97fpinge4bq6lde4dvq11; zdregion=MTI5LjIuMTI5LjE1Mzp1czp1czp1ZDJmNWY5YTdkODU1N2Q2YzM5NGU3M2Y1ZTRmNQ==
Why use cookies?

• 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*
    - Read about iPad on CNN, then see ads on Amazon?!
  - How can an advertiser (A) know what you did on another site (S)?
Why use cookies?

- Tracking users
  - Advertisers want to know your behavior
  - Ideally build a profile across different websites
    - Read about iPad on CNN, then see ads on Amazon?!
  - How can an advertiser (A) know what you did on another site (S)?

S shows you an ad from A; A scrapes the referrer URL
Why use cookies?

• **Tracking users**
  • Advertisers want to know your behavior
  • Ideally build a profile *across different websites*
    - Read about iPad on CNN, then see ads on Amazon?!
  • How can an advertiser (A) know what you did on another site (S)?

  S shows you an ad from A; A scrapes the referrer URL

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

**Problem:** IP addr add change
Why use cookies?

• Tracking users
  • Advertisers want to know your behavior
  • Ideally build a profile across different websites
    - Read about iPad on CNN, then see ads on Amazon?!
  • How can an advertiser (A) know what you did on another site (S)?

S shows you an ad from A; A scrapes the referrer URL

Option 1: A maintains a DB, indexed by your IP address
Problem: IP addrs change

Option 2: A maintains a DB indexed by a cookie
- “Third-party cookie”
- Commonly used by large ad networks (doubleclick)
They should put a tiny message at the end of chapstick tubes congratulating you for not losing the damn thing.

Meet Biddy, The Traveling Hedgehog

Mt. Fuji overlooking Yokohama

RIP in peace

[Image] Stop Letting People

Hacker Claims Feds Hit Him With 44 Felonies When He Refused to Be an FBI Spy
Snippet of reddit.com source
Snippet of reddit.com source

Our first time accessing adzerk.net
I visit reddit.com
I visit reddit.com
I visit reddit.com
I visit reddit.com

Later, I go to reddit.com/r/security
I visit reddit.com

Later, I go to reddit.com/r/security
I visit reddit.com

Later, I go to reddit.com/r/security
I visit reddit.com

Later, I go to reddit.com/r/security

We are only sharing this cookie with *.adzerk.net; but we are telling them about where we just came from

Later, I go to reddit.com/r/security

Cookie: cfduid=dc3a93cd30ca47b76600d63cde283e9b81424367471
Cookies and web authentication

• An extremely common use of cookies is to track users who have already authenticated.

• If the user already visited
  
  http://website.com/login.html?user=alice&pass=secret
  
  with the correct password, then the server associates a “session cookie” with the logged-in user’s info.

• Subsequent requests (GET and POST) include the cookie in the request headers and/or as one of the fields:
  
  http://website.com/doStuff.html?sid=81asf98as8eak

• The idea is for the server to be able to say “I am talking to the same browser that authenticated Alice earlier.”
Cookies and web authentication

- An extremely common use of cookies is to track users who have already authenticated

- If the user already visited `http://website.com/login.html?user=alice&pass=secret` with the correct password, then the server associates a “session cookie” with the logged-in user’s info

- Subsequent requests (GET and POST) include the cookie in the request headers and/or as one of the fields: `http://website.com/doStuff.html?sid=81asf98as8eak`

- The idea is for the server to be able to say “I am talking to the same browser that authenticated Alice earlier.”

**Attacks?**