This time

Continuing with

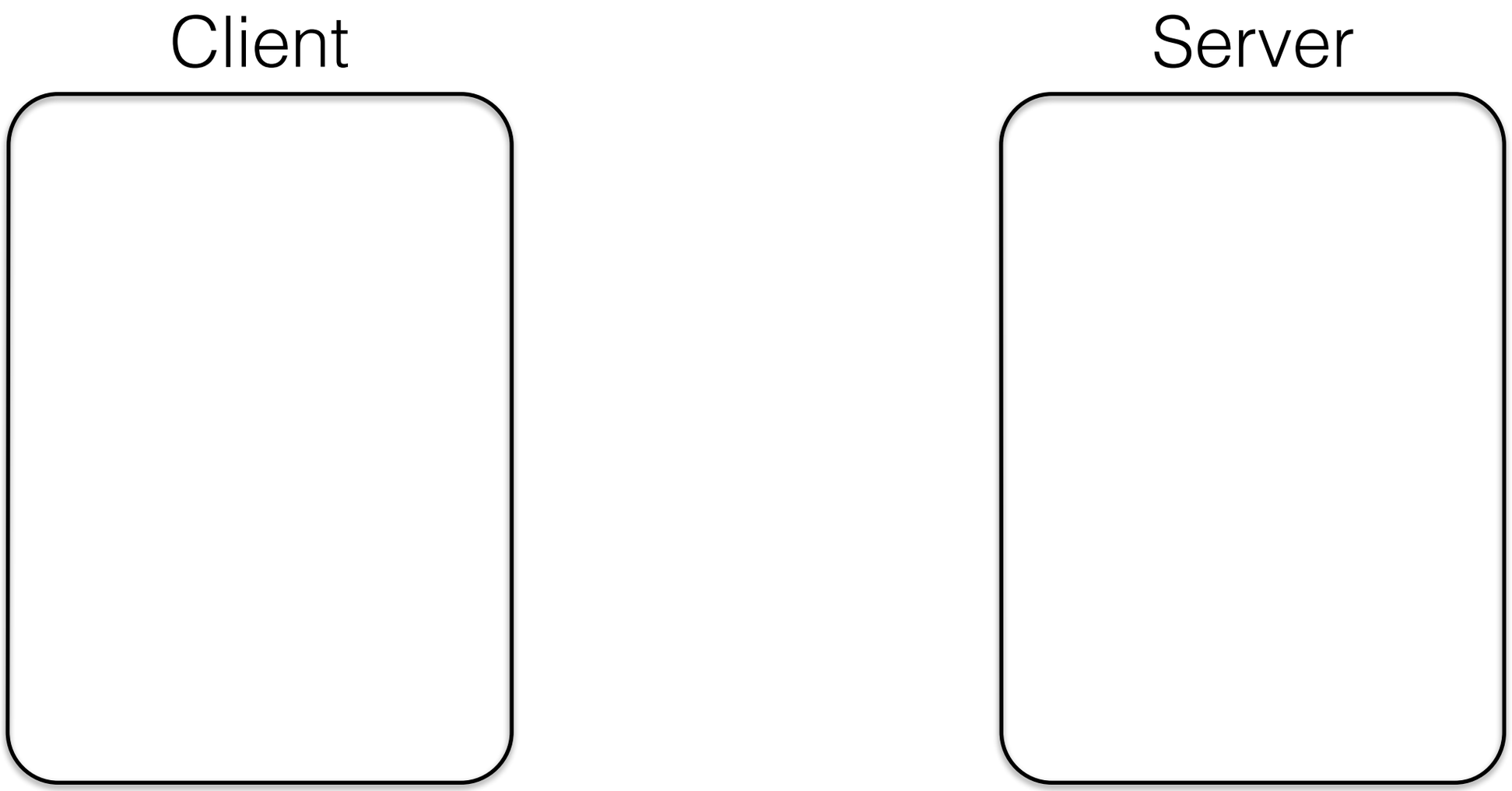
Software Security

Getting insane with

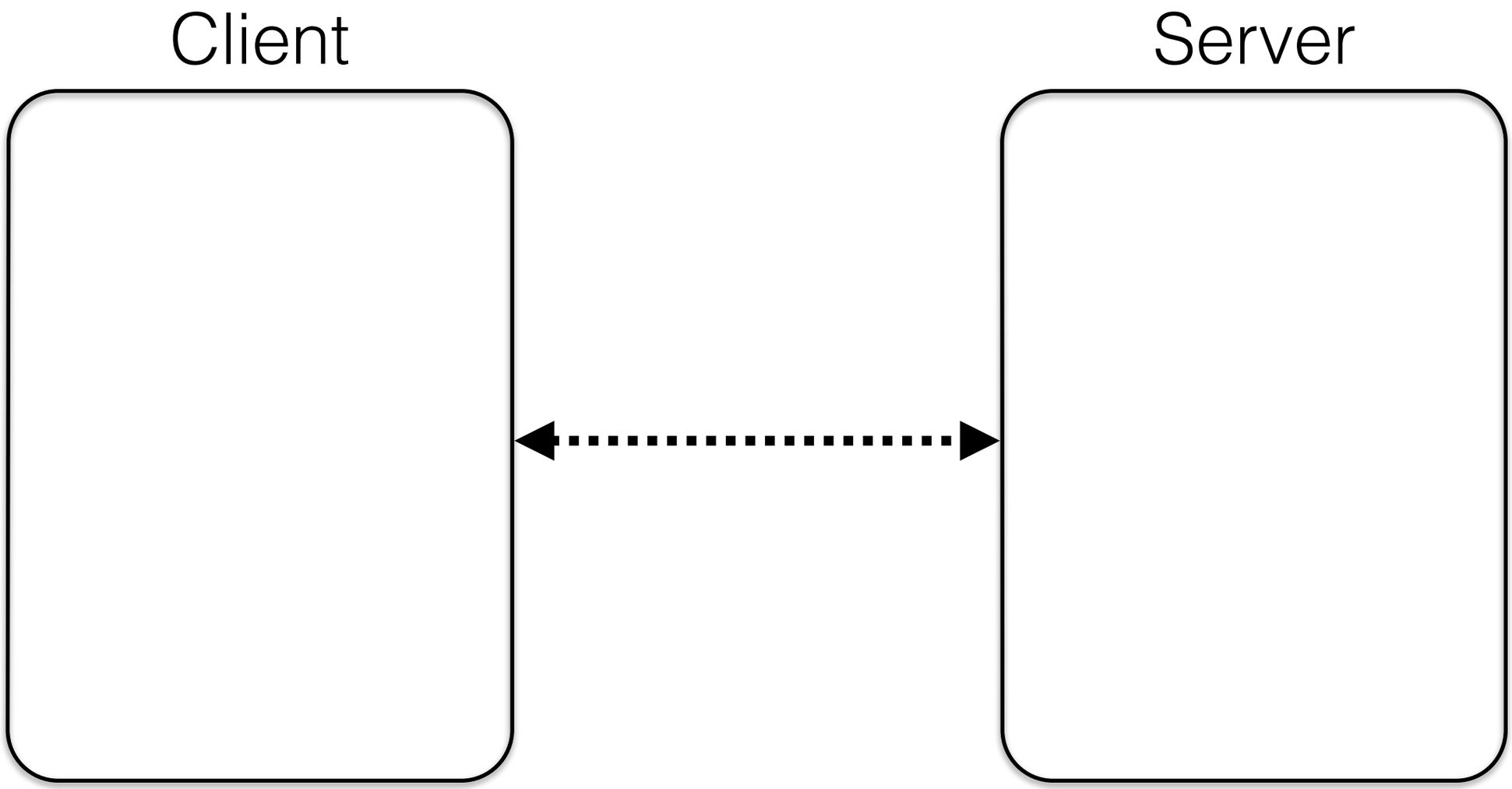
Input sanitization’); drop table slides

- New attacks and countermeasures:
  - SQL injection
  - Background on web architectures

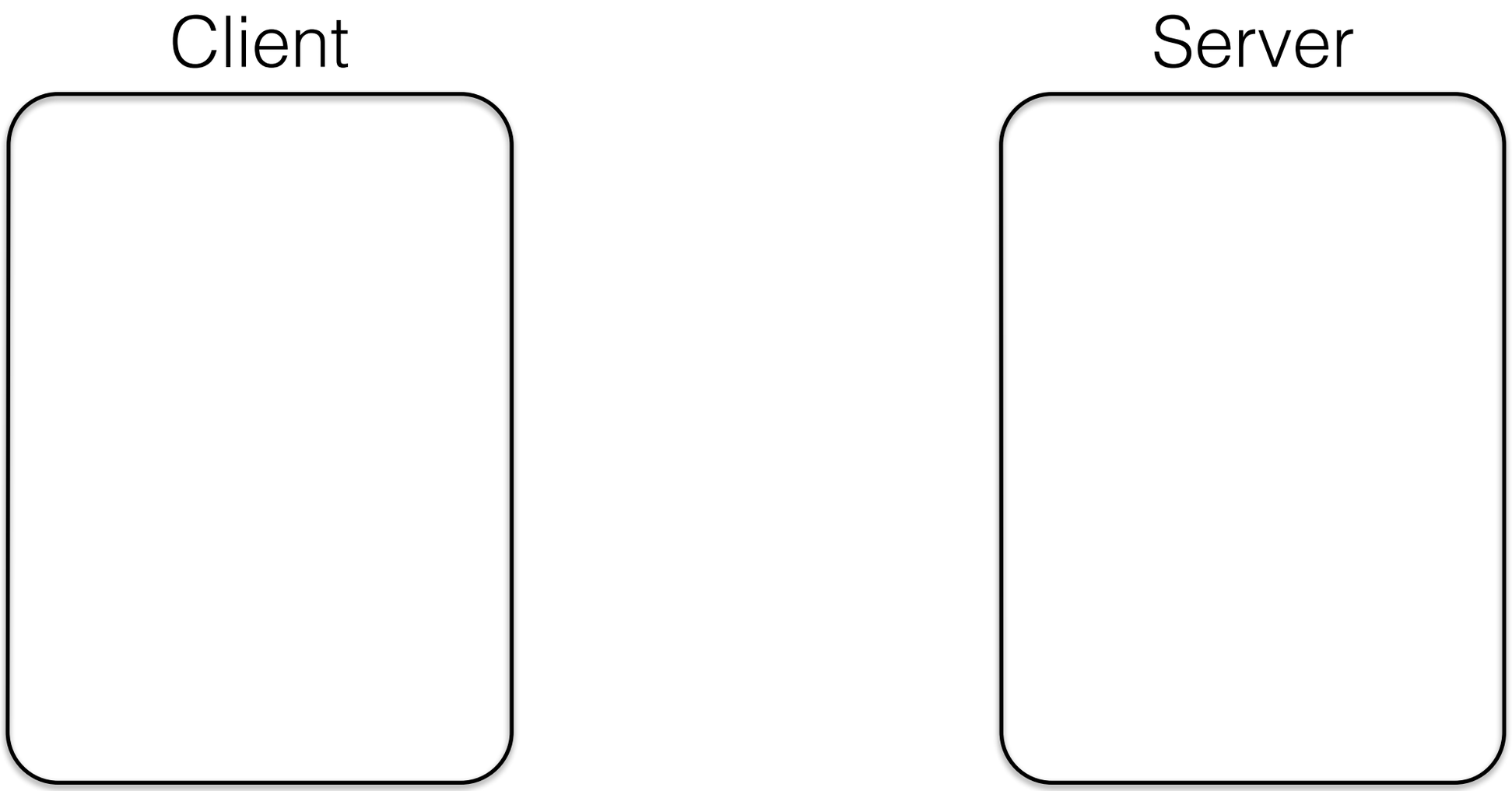
A very basic web architecture



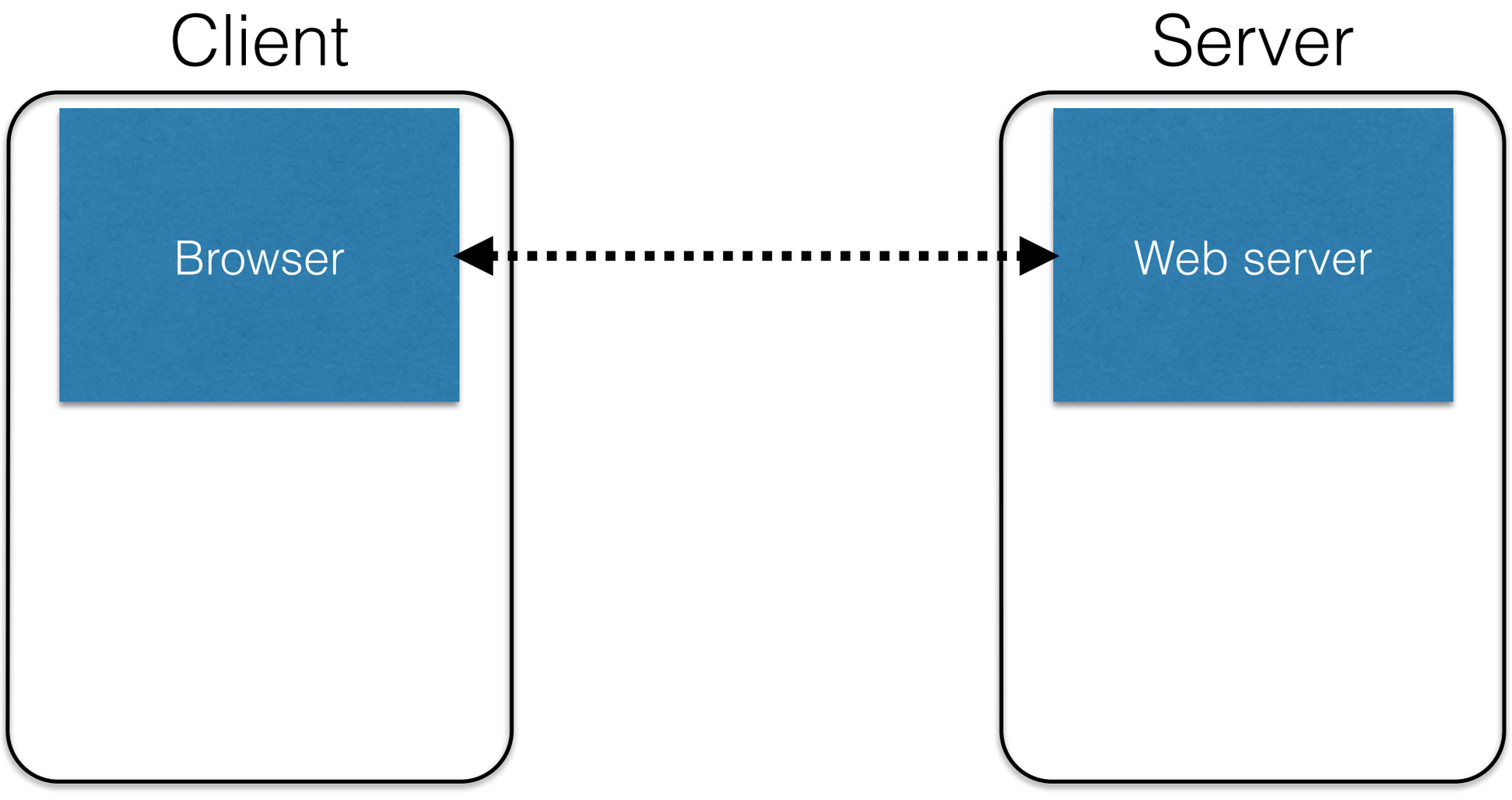
A very basic web architecture



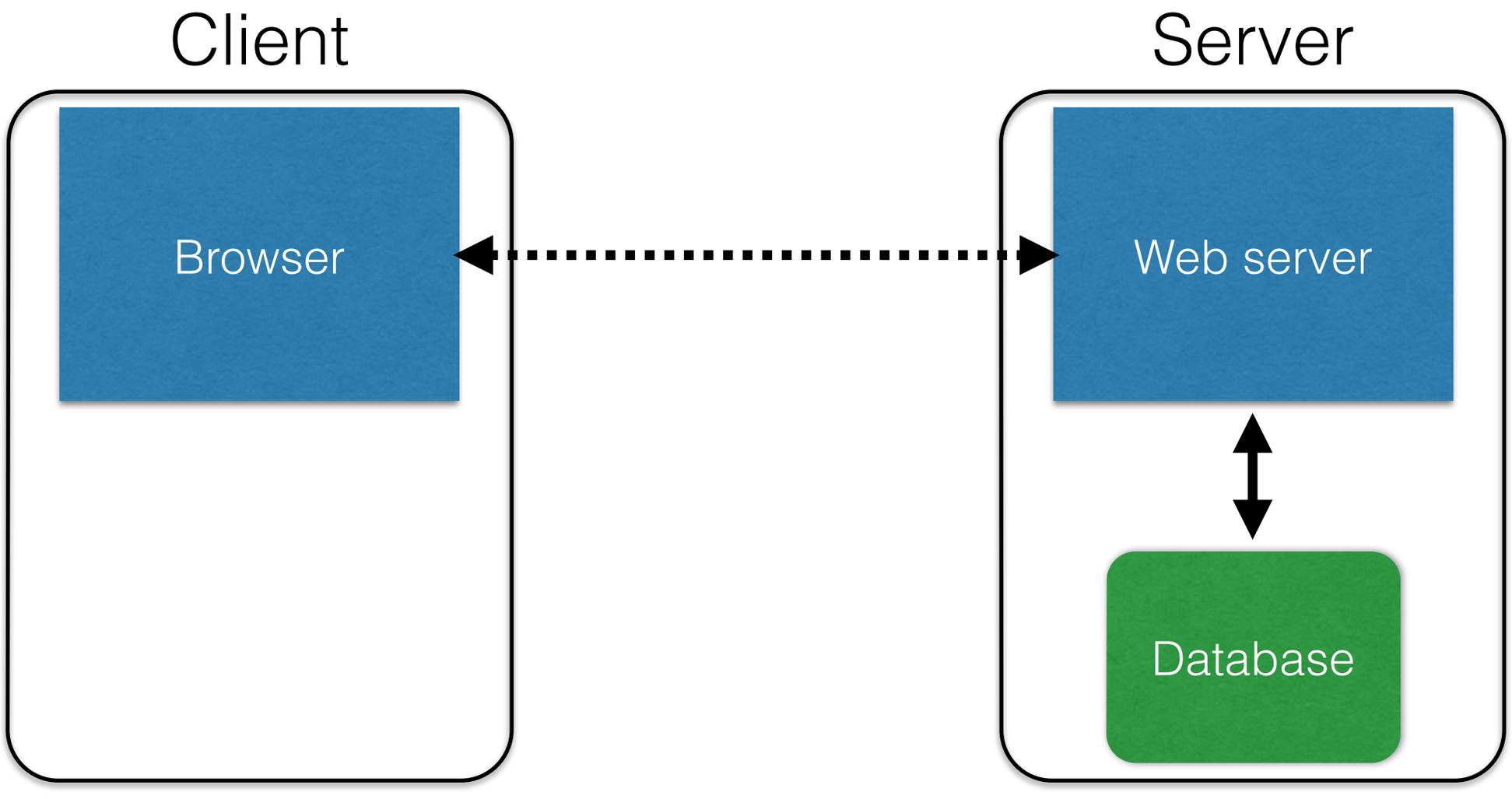
A very basic web architecture



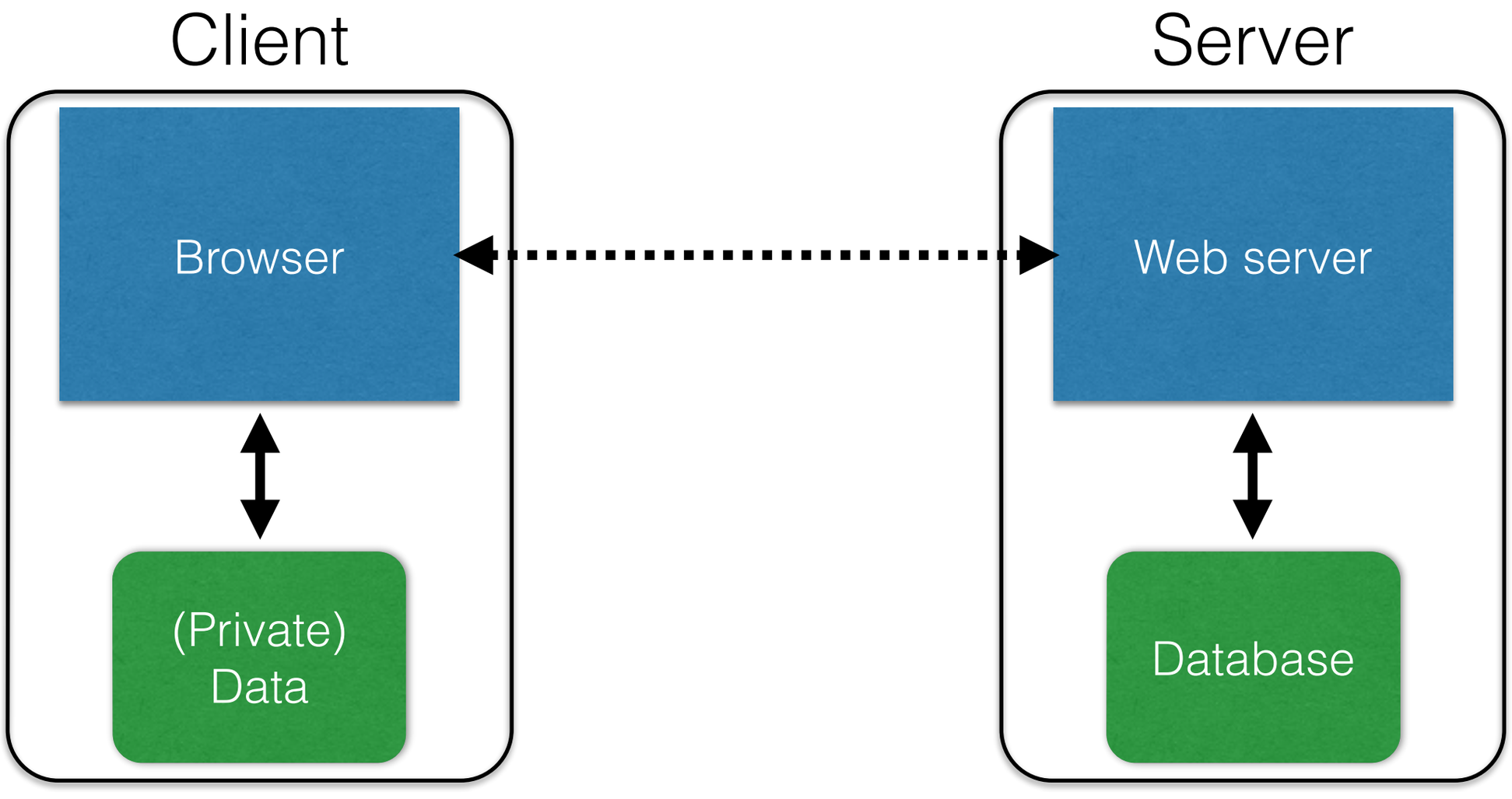
A very basic web architecture



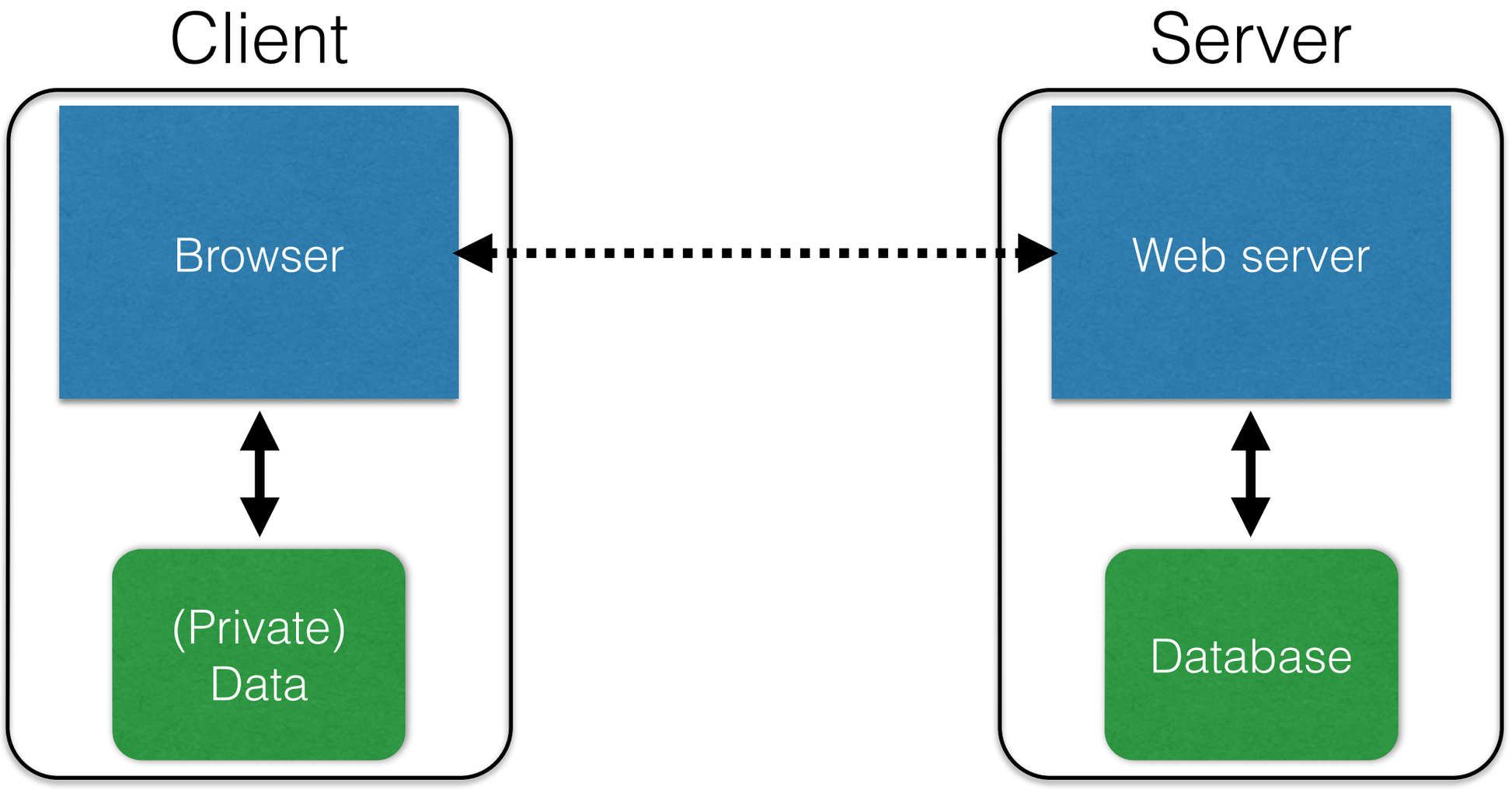
A very basic web architecture



A very basic web architecture



A very basic web architecture



DB is a separate entity, logically (and often physically)

SQL security

Databases

- Provide data storage & data manipulation
- Database designer lays out the data into tables
- Programmers query the database
- Database Management Systems (DBMSes) provide
  - semantics for how to organize data
  - transactions for manipulating data sanely
  - a **language** for creating & querying data
    - and APIs to interoperate with other languages
  - management via users & permissions

Databases: basics

Users

Name	Gender	Age	Email	Password
Dee	F	28	dee@pp.com	j3i8g8ha
Mac	M	7	bouncer@pp.com	a0u23bt
Charlie	M	32	aneifjask@pp.com	0aergja
Dennis	M	28	imagod@pp.com	1bjb9a93

Databases: basics

Table

Users

Name	Gender	Age	Email	Password
Dee	F	28	dee@pp.com	j3i8g8ha
Mac	M	7	bouncer@pp.com	a0u23bt
Charlie	M	32	aneifjask@pp.com	0aergja
Dennis	M	28	imagod@pp.com	1bjb9a93

Databases: basics

Users

Name	Gender	Age	Email	Password
Dee	F	28	dee@pp.com	j3i8g8ha
Mac	M	7	bouncer@pp.com	a0u23bt
Charlie	M	32	aneifjask@pp.com	0aergja
Dennis	M	28	imagod@pp.com	1bjb9a93

Databases: basics

Users **Table name**

Name	Gender	Age	Email	Password
Dee	F	28	dee@pp.com	j3i8g8ha
Mac	M	7	bouncer@pp.com	a0u23bt
Charlie	M	32	aneifjask@pp.com	0aergja
Dennis	M	28	imagod@pp.com	1bjb9a93

Databases: basics

Users

Name	Gender	Age	Email	Password
Dee	F	28	dee@pp.com	j3i8g8ha
Mac	M	7	bouncer@pp.com	a0u23bt
Charlie	M	32	aneifjask@pp.com	0aergja
Dennis	M	28	imagod@pp.com	1bjb9a93

Databases: basics

Users

Name	Gender	Age	Email	Password
Dee	F	28	dee@pp.com	j3i8g8ha
Mac	M	7	bouncer@pp.com	a0u23bt
Charlie	M	32	aneifjask@pp.com	0aergja
Dennis	M	28	imagod@pp.com	1bjb9a93

Column

Databases: basics

Users

Name	Gender	Age	Email	Password
Dee	F	28	dee@pp.com	j3i8g8ha
Mac	M	7	bouncer@pp.com	a0u23bt
Charlie	M	32	aneifjask@pp.com	0aergja
Dennis	M	28	imagod@pp.com	1bjb9a93

Databases: basics

Users

Name	Gender	Age	Email	Password
Dee	F	28	dee@pp.com	j3i8g8ha
Mac	M	7	bouncer@pp.com	a0u23bt
Charlie	M	32	aneifjask@pp.com	0aergja
Dennis	M	28	imagod@pp.com	1bjb9a93

Row (Record)

Databases: basics

Users

Name	Gender	Age	Email	Password
Dee	F	28	dee@pp.com	j3i8g8ha
Mac	M	7	bouncer@pp.com	a0u23bt
Charlie	M	32	aneifjask@pp.com	0aergja
Dennis	M	28	imagod@pp.com	1bjb9a93

Database transactions

Transactions are the unit of work on a database

Database transactions

Transactions are the unit of work on a database

"Give me everyone in the User table who is listed as taking CMSC414 in the Classes table"

"Deduct $100 from Alice; Add $100 to Bob"

Database transactions

Transactions are the unit of work on a database

"Give me everyone in the User table who is listed as taking CMSC414 in the Classes table" 2 reads

"Deduct $100 from Alice; Add $100 to Bob" 2 writes

Database transactions

Transactions are the unit of work on a database

"Give me everyone in the User table who is listed as taking CMSC414 in the Classes table" 2 reads

1 transaction

"Deduct $100 from Alice; Add $100 to Bob" 2 writes

Database transactions

Transactions are the unit of work on a database

"Give me everyone in the User table who is listed as taking CMSC414 in the Classes table" 2 reads

1 transaction

"Deduct $100 from Alice; Add $100 to Bob" 2 writes

- Typically want ACID transactions
  - **A**tomicity: Transactions complete entirely or not at all
  - **C**onsistency: The database is always in a *valid* state (but not necessarily *correct*)
  - **I**solation: Results from a transaction aren't visible until it is complete
  - **D**urability: Once a transaction is committed, it remains, despite, e.g., power failures

SQL (Standard Query Language)

Users

Name	Gender	Age	Email	Password
Dee	F	28	dee@pp.com	j3i8g8ha
Mac	M	7	bouncer@pp.com	a0u23bt
Charlie	M	32	aneifjask@pp.com	0aergja
Dennis	M	28	imagod@pp.com	1bjb9a93

SQL (Standard Query Language)

Users

Name	Gender	Age	Email	Password
Dee	F	28	dee@pp.com	j3i8g8ha
Mac	M	7	bouncer@pp.com	a0u23bt
Charlie	M	32	aneifjask@pp.com	0aergja
Dennis	M	28	imagod@pp.com	1bjb9a93

```
SELECT Age FROM Users WHERE Name='Dee';
```

SQL (Standard Query Language)

Users

Name	Gender	Age	Email	Password
Dee	F	28	dee@pp.com	j3i8g8ha
Mac	M	7	bouncer@pp.com	a0u23bt
Charlie	M	32	aneifjask@pp.com	0aergja
Dennis	M	28	imagod@pp.com	1bjb9a93

```
SELECT Age FROM Users WHERE Name='Dee';
```

28

SQL (Standard Query Language)

Users

Name	Gender	Age	Email	Password
Dee	F	28	dee@pp.com	j3i8g8ha
Mac	M	7	bouncer@pp.com	a0u23bt
Charlie	M	32	aneifjask@pp.com	0aergja
Dennis	M	28	imagod@pp.com	1bjb9a93

```
SELECT Age FROM Users WHERE Name='Dee';
```

28

```
UPDATE Users SET email='readgood@pp.com'
    WHERE Age=32; -- this is a comment
```

SQL (Standard Query Language)

Users

Name	Gender	Age	Email	Password
Dee	F	28	dee@pp.com	j3i8g8ha
Mac	M	7	bouncer@pp.com	a0u23bt
Charlie	M	32	readgood@pp.com	0aergja
Dennis	M	28	imagod@pp.com	1bjb9a93

```
SELECT Age FROM Users WHERE Name='Dee';
```

28

```
UPDATE Users SET email='readgood@pp.com'
    WHERE Age=32; -- this is a comment
```

SQL (Standard Query Language)

Users

Name	Gender	Age	Email	Password
Dee	F	28	dee@pp.com	j3i8g8ha
Mac	M	7	bouncer@pp.com	a0u23bt
Charlie	M	32	readgood@pp.com	0aergja
Dennis	M	28	imagod@pp.com	1bjb9a93

```
SELECT Age FROM Users WHERE Name='Dee';
```
28

```
UPDATE Users SET email='readgood@pp.com'
    WHERE Age=32; -- this is a comment
INSERT INTO Users Values('Frank', 'M', 57, ...);
```

SQL (Standard Query Language)

Users

Name	Gender	Age	Email	Password
Dee	F	28	dee@pp.com	j3i8g8ha
Mac	M	7	bouncer@pp.com	a0u23bt
Charlie	M	32	readgood@pp.com	0aergja
Dennis	M	28	imagod@pp.com	1bjb9a93
Frank	M	57	armed@pp.com	ziog9gga

```
SELECT Age FROM Users WHERE Name='Dee';
```
28

```
UPDATE Users SET email='readgood@pp.com'
    WHERE Age=32; -- this is a comment
INSERT INTO Users Values('Frank', 'M', 57, ...);
```

SQL (Standard Query Language)

Users

Name	Gender	Age	Email	Password
Dee	F	28	dee@pp.com	j3i8g8ha
Mac	M	7	bouncer@pp.com	a0u23bt
Charlie	M	32	readgood@pp.com	0aergja
Dennis	M	28	imagod@pp.com	1bjb9a93
Frank	M	57	armed@pp.com	ziog9gga

```
SELECT Age FROM Users WHERE Name='Dee';
```
28

```
UPDATE Users SET email='readgood@pp.com'
    WHERE Age=32; -- this is a comment
INSERT INTO Users Values('Frank', 'M', 57, ...);
DROP TABLE Users;
```

SQL (Standard Query Language)

```
SELECT Age FROM Users WHERE Name='Dee';    28
UPDATE Users SET email='readgood@pp.com'
    WHERE Age=32; -- this is a comment
INSERT INTO Users Values('Frank', 'M', 57, ...);
DROP TABLE Users;
```

Server-side code

Website

“Login code” (php)

```
$result = mysql_query(“select * from Users
        where(name=‘$user’ and password=‘$pass’);”);
```

Suppose you successfully log in as $user
if this query returns any rows whatsoever

Server-side code

Website

“Login code” (php)

```
$result = mysql_query(“select * from Users
        where(name=‘$user’ and password=‘$pass’);”);
```

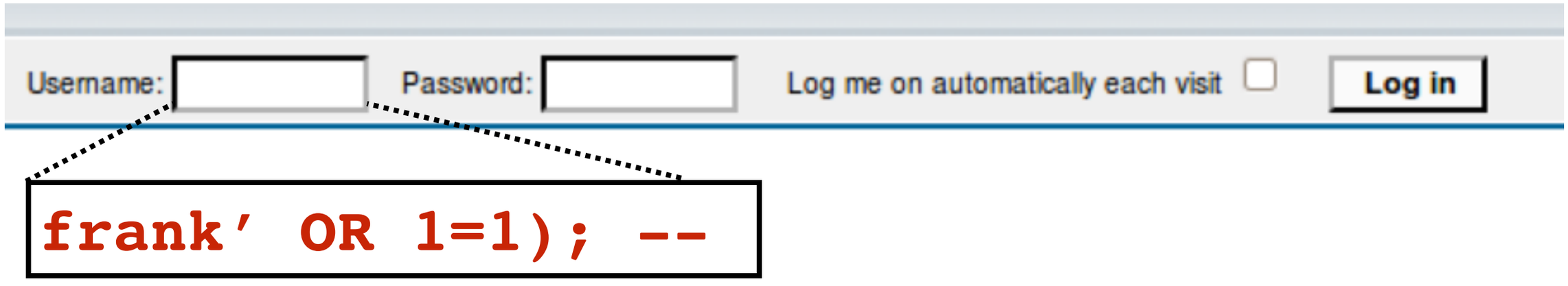
Suppose you successfully log in as $user
if this query returns any rows whatsoever

How could you exploit this?

SQL injection

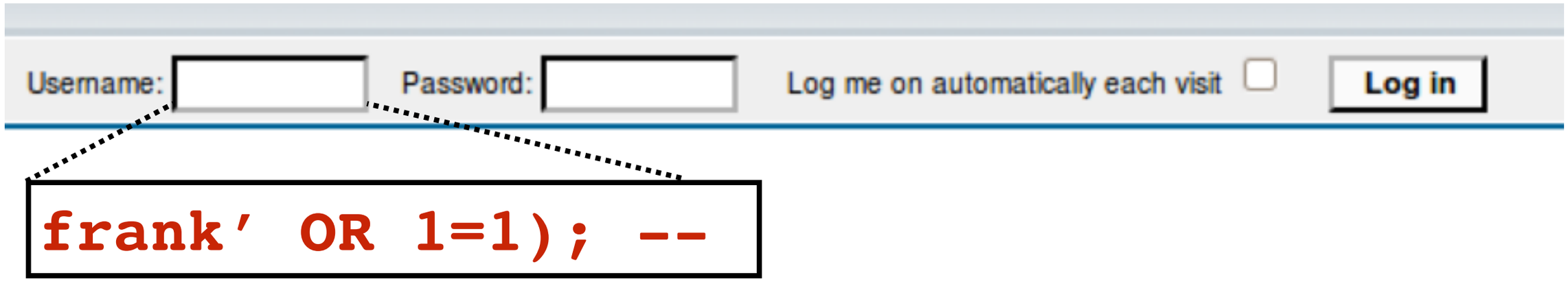

```
$result = mysql_query(“select * from Users
        where(name=‘$user’ and password=‘$pass’);”);
```

SQL injection




```
$result = mysql_query("select * from Users
        where(name='$user' and password='$pass');");
```

SQL injection




```
$result = mysql_query("select * from Users
        where(name='$user' and password='$pass');");


$result = mysql_query("select * from Users
        where(name='frank' OR 1=1); --
         and password='whocares');");
```

SQL injection


```
$result = mysql_query("select * from Users
        where(name='$user' and password='$pass');");
```

Can chain together statements with semicolon:
STATEMENT 1 ; STATEMENT 2

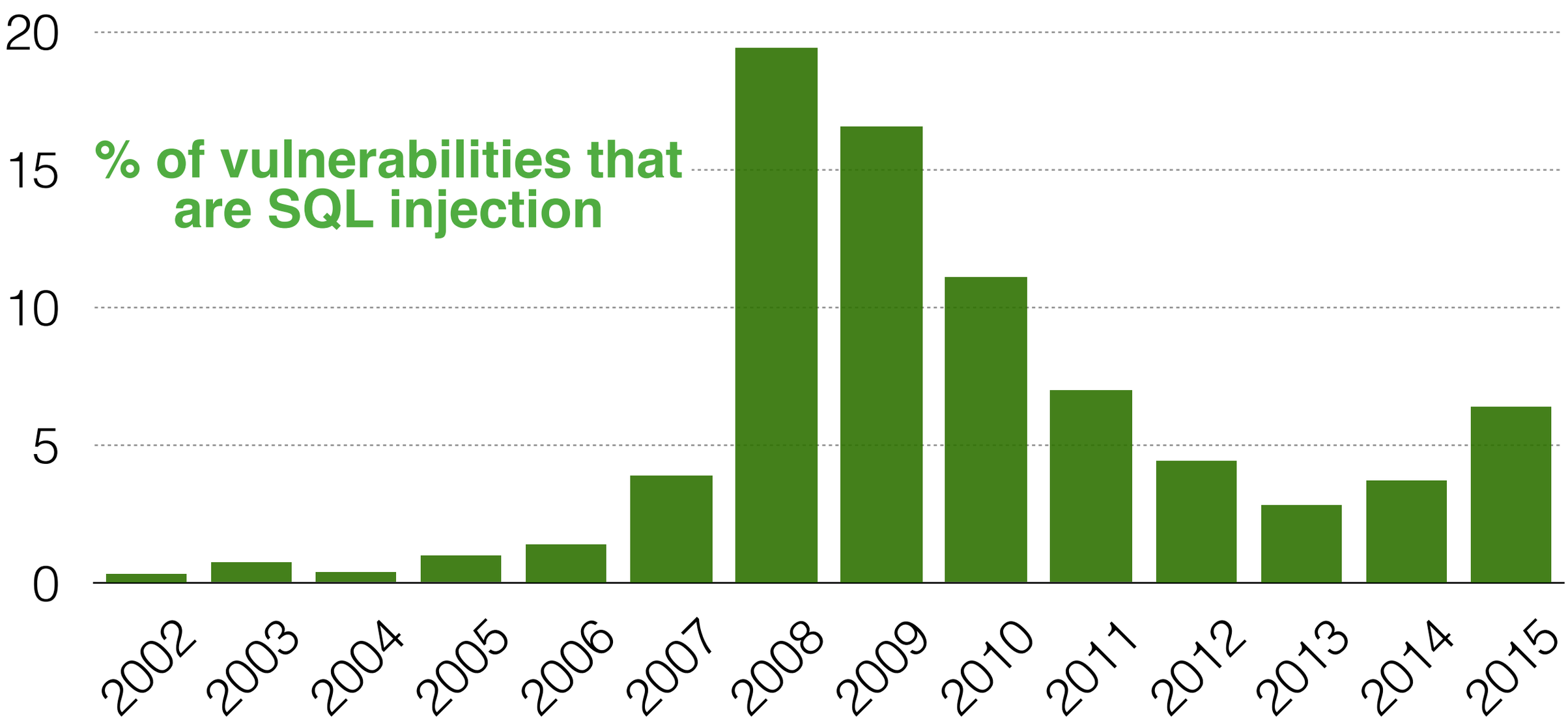
SQL injection


```
$result = mysql_query("select * from Users
        where(name='$user' and password='$pass');");

$result = mysql_query("select * from Users
        where(name='frank' OR 1=1);
        DROP TABLE Users; --
        ' and password='whocares');");
```

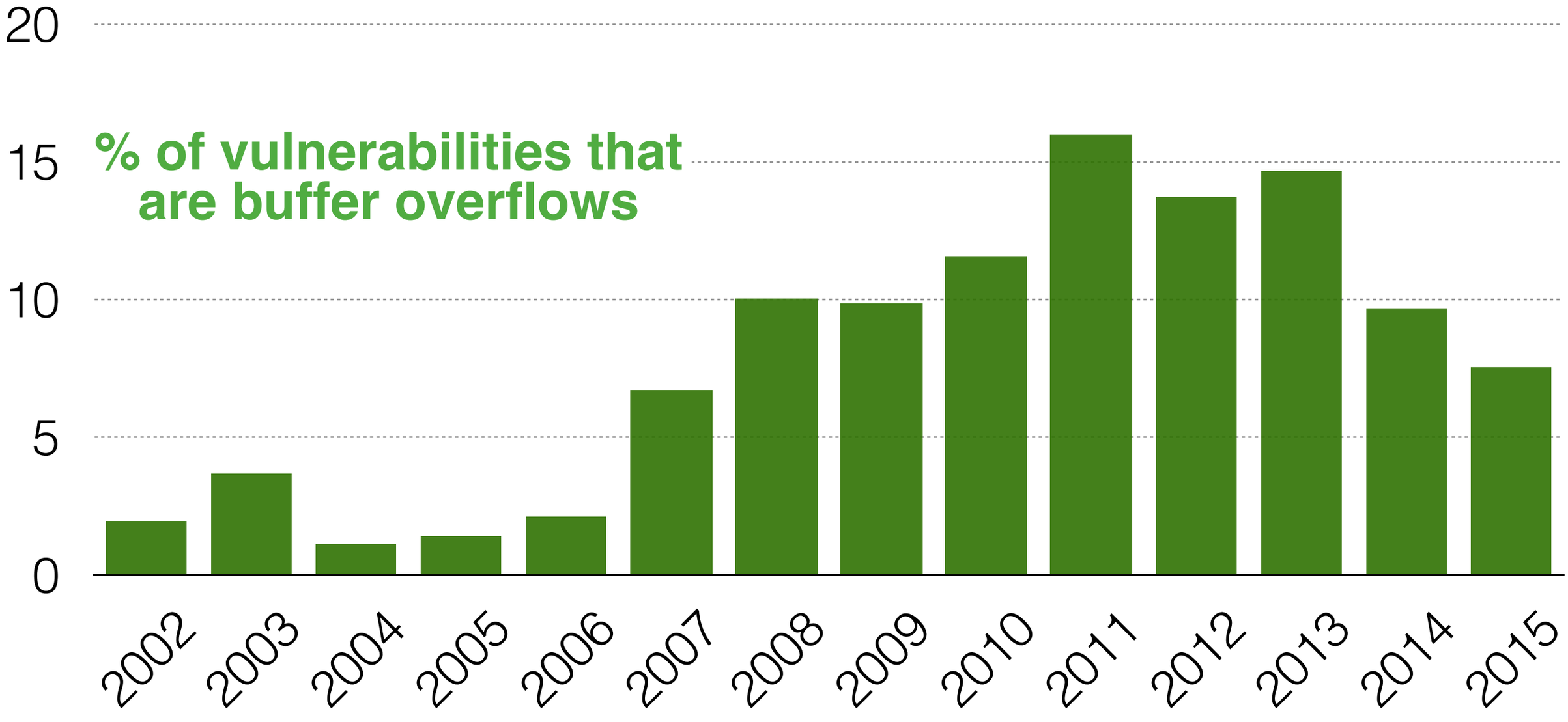
Can chain together statements with semicolon:
STATEMENT 1 ; STATEMENT 2

SQL injection attacks are prevalent

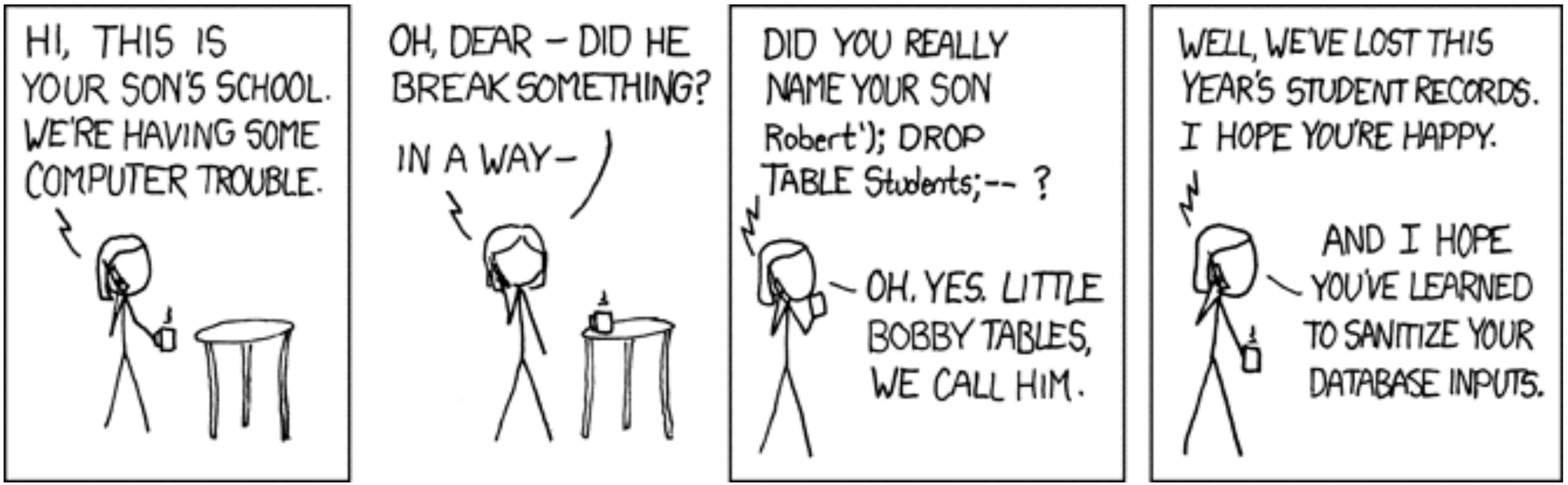


http://web.nvd.nist.gov/view/vuln/statistics

Buffer overflow attacks are prevalent



http://web.nvd.nist.gov/view/vuln/statistics


HI, THIS IS YOUR SON'S SCHOOL. WE'RE HAVING SOME COMPUTER TROUBLE.
OH, DEAR - DID HE BREAK SOMETHING?
IN A WAY-
DID YOU REALLY NAME YOUR SON Robert'); DROP TABLE Students;-- ?
OH. YES. LITTLE BOBBY TABLES, WE CALL HIM.
WELL, WE'VE LOST THIS YEAR'S STUDENT RECORDS. I HOPE YOU'RE HAPPY.
AND I HOPE YOU'VE LEARNED TO SANITIZE YOUR DATABASE INPUTS.


ZU 0666', 0, 0); DROP DATABASE TABLICE;

SQL injection countermeasures

- Blacklisting: Delete the characters you don't want
  - '
  - --
  - ;
- Downside: "Peter O'Connor"
  - You want these characters sometimes!
  - How do you know if/when the characters are bad?

SQL injection countermeasures

1. Whitelisting

- Check that the user-provided input is in some set of values known to be safe
  - Integer within the right range

- Given an invalid input, better to reject than to fix
  - "Fixes" may introduce vulnerabilities
  - *Principle of fail-safe defaults*

- Downside:
  - Um.. Names come from a well-known dictionary?

SQL injection countermeasures

2. Escape characters

- Escape characters that could alter control
  - ’ ⇒ \’
  - ; ⇒ \;
  - - ⇒ \-
  - \ ⇒ \\
- Hard by hand, but there are many libs & methods
  - magic_quotes_gpc = On
  - mysql_real_escape_string()
- Downside: Sometimes you want these in your SQL!

The underlying issue

```
$result = mysql_query(“select * from Users
        where(name=‘$user’ and password=‘$pass’);”);
```

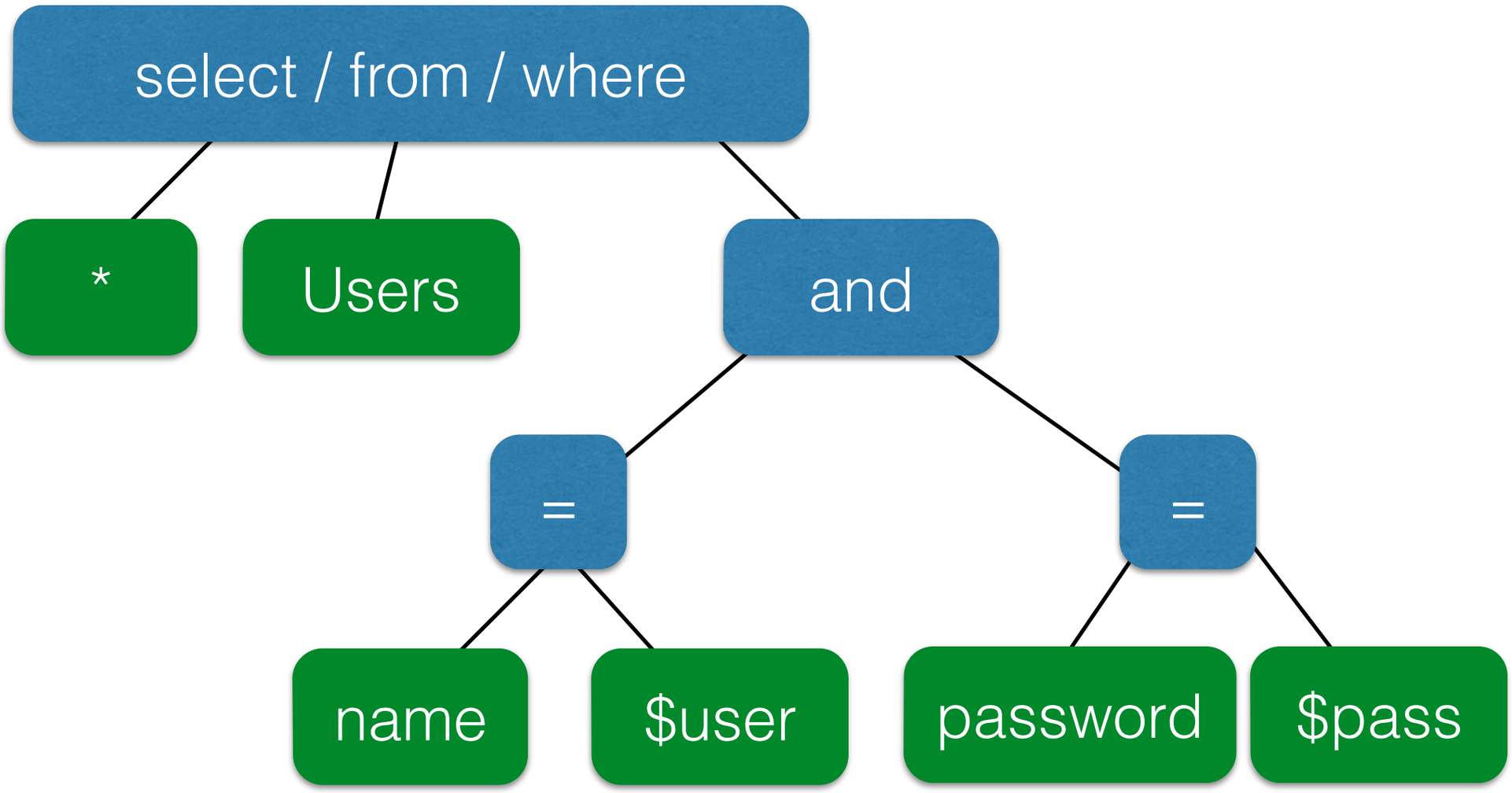
- This one string combines the code and the data

- Similar to buffer overflows:

When the boundary between code and data blurs, we open ourselves up to vulnerabilities

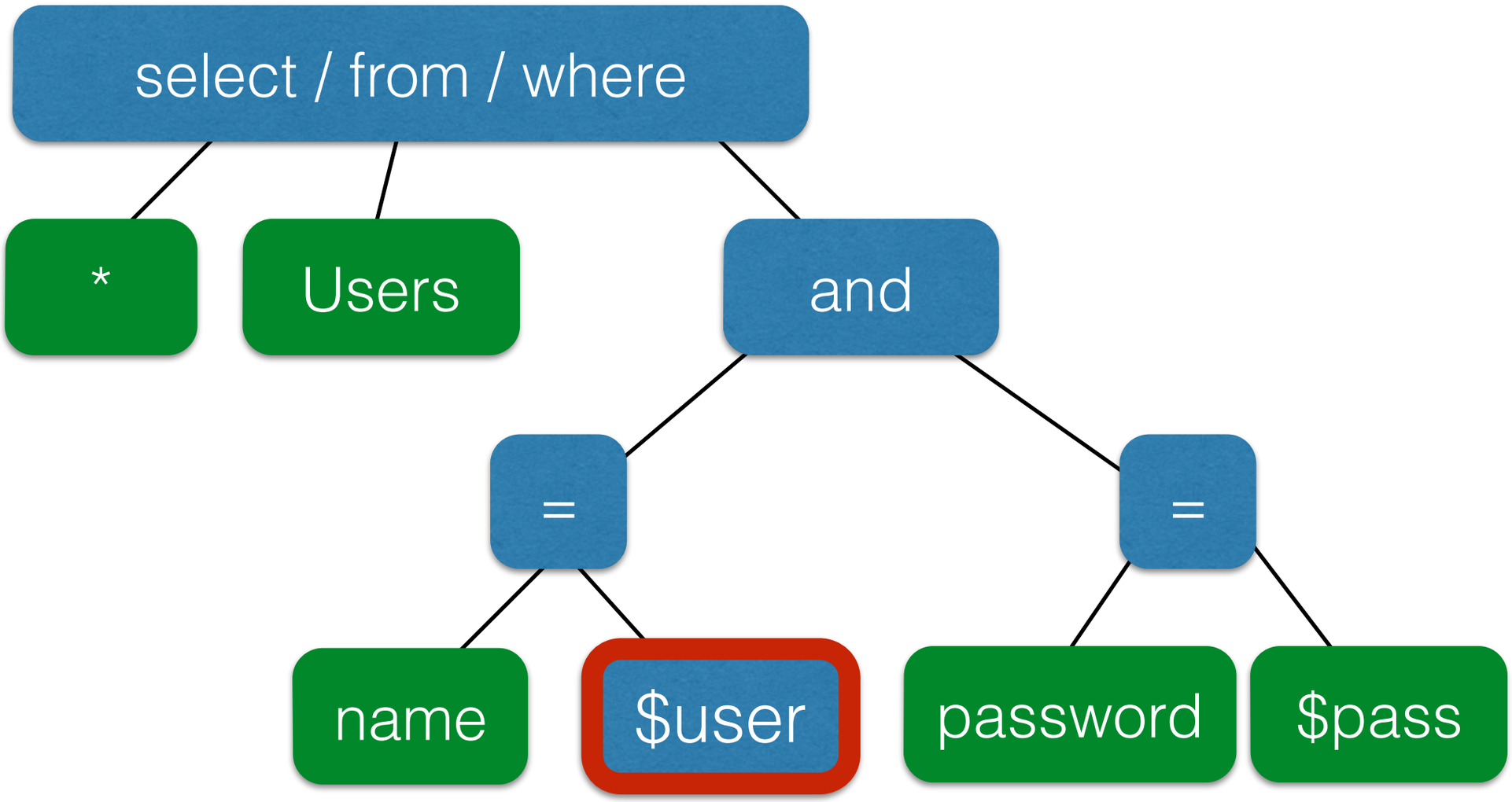
The underlying issue

```
$result = mysql_query(“select * from Users
        where(name=‘$user’ and password=‘$pass’);”);
```



The underlying issue

```
$result = mysql_query(“select * from Users
        where(name=‘$user’ and password=‘$pass’);”);
```



SQL injection countermeasures

3. Prepared statements & bind variables

Key idea: *Decouple* the code and the data

```
$result = mysql_query("select * from Users
        where(name='$user' and password='$pass');");
```

SQL injection countermeasures

3. Prepared statements & bind variables

Key idea: *Decouple* the code and the data

```
$result = mysql_query("select * from Users
        where(name='$user' and password='$pass');");
```

```
$db = new mysql("localhost", "user", "pass", "DB");

$statement = $db->prepare("select * from Users
     where(name=? and password=?);");


$statement->bind_param("ss", $user, $pass);
$statement->execute();
```

SQL injection countermeasures

3. Prepared statements & bind variables

Key idea: *Decouple* the code and the data

```
$result = mysql_query("select * from Users
        where(name='$user' and password='$pass');");
```

```
$db = new mysql("localhost", "user", "pass", "DB");

$statement = $db->prepare("select * from Users
     where(name=? and password=?);");
```

Bind variables

```
$statement->bind_param("ss", $user, $pass);
$statement->execute();
```

SQL injection countermeasures

3. Prepared statements & bind variables

Key idea: *Decouple* the code and the data

```
$result = mysql_query("select * from Users
        where(name='$user' and password='$pass');");
```

```
$db = new mysql("localhost", "user", "pass", "DB");

$statement = $db->prepare("select * from Users
     where(name=? and password=?);");
```

Bind variables

```
$statement->bind_param("ss", $user, $pass);
$statement->execute();
```

Bind variables are typed

SQL injection countermeasures

3. Prepared statements & bind variables

Key idea: *Decouple* the code and the data

```
$result = mysql_query("select * from Users
        where(name='$user' and password='$pass');");
```

```
$db = new mysql("localhost", "user", "pass", "DB");

$statement = $db->prepare("select * from Users
     where(name=? and password=?);");
```

Bind variables

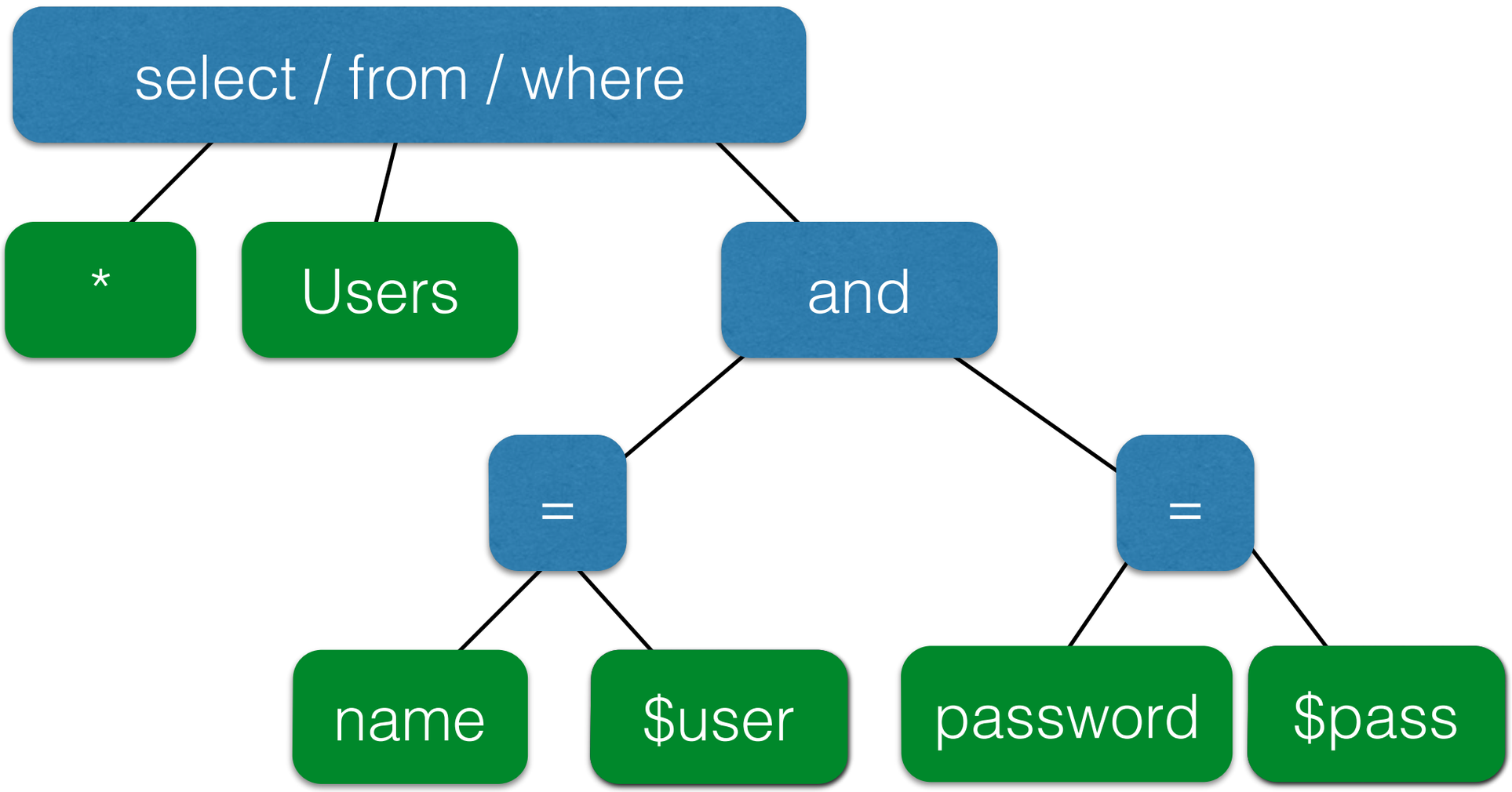
Decoupling lets us compile now, before binding the data

```
$statement->bind_param("ss", $user, $pass);
$statement->execute();
```

Bind variables are typed

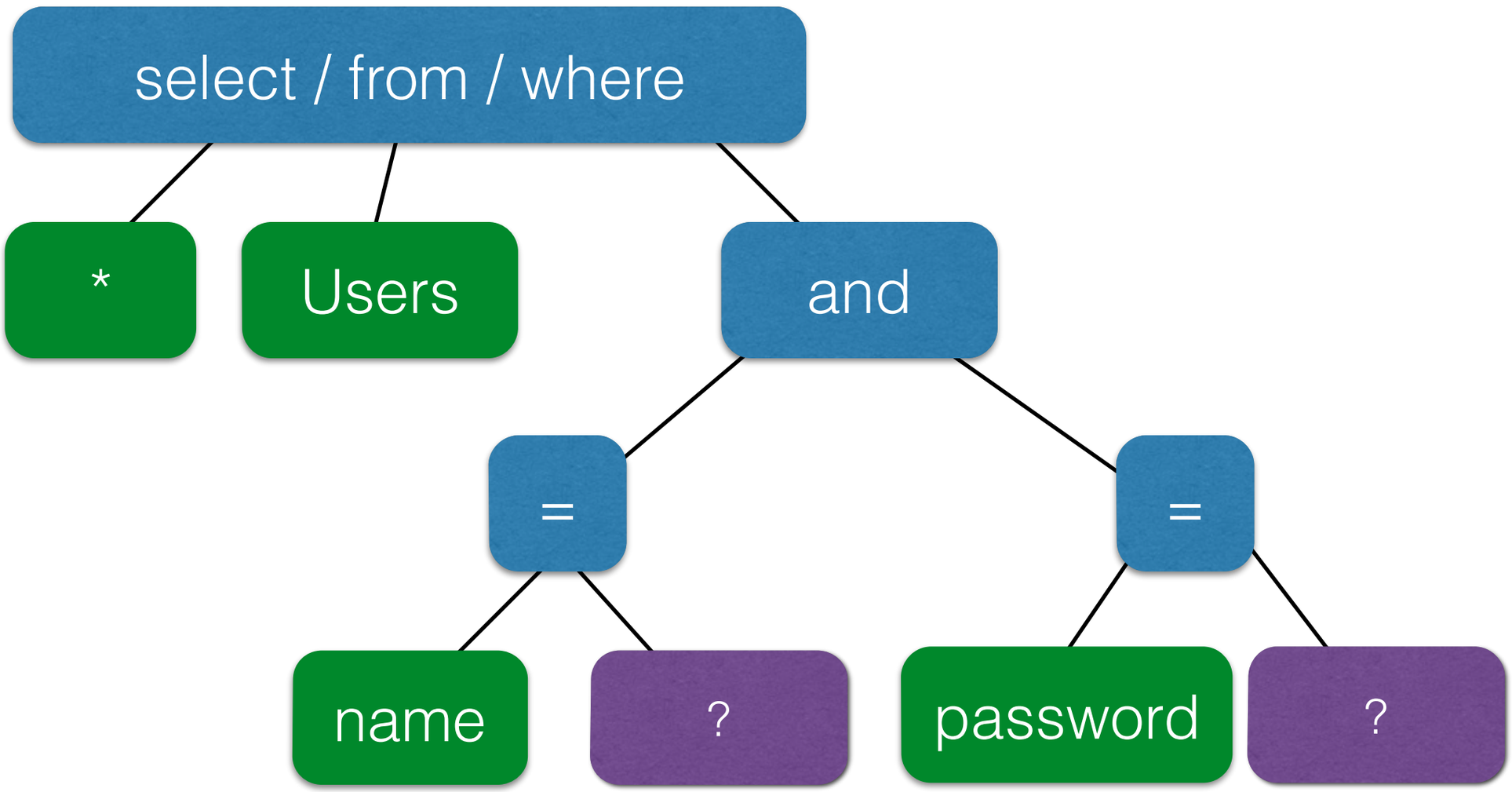
The underlying issue

```
$statement = $db->prepare(“select * from Users
     where(name=? and password=?);”);
```



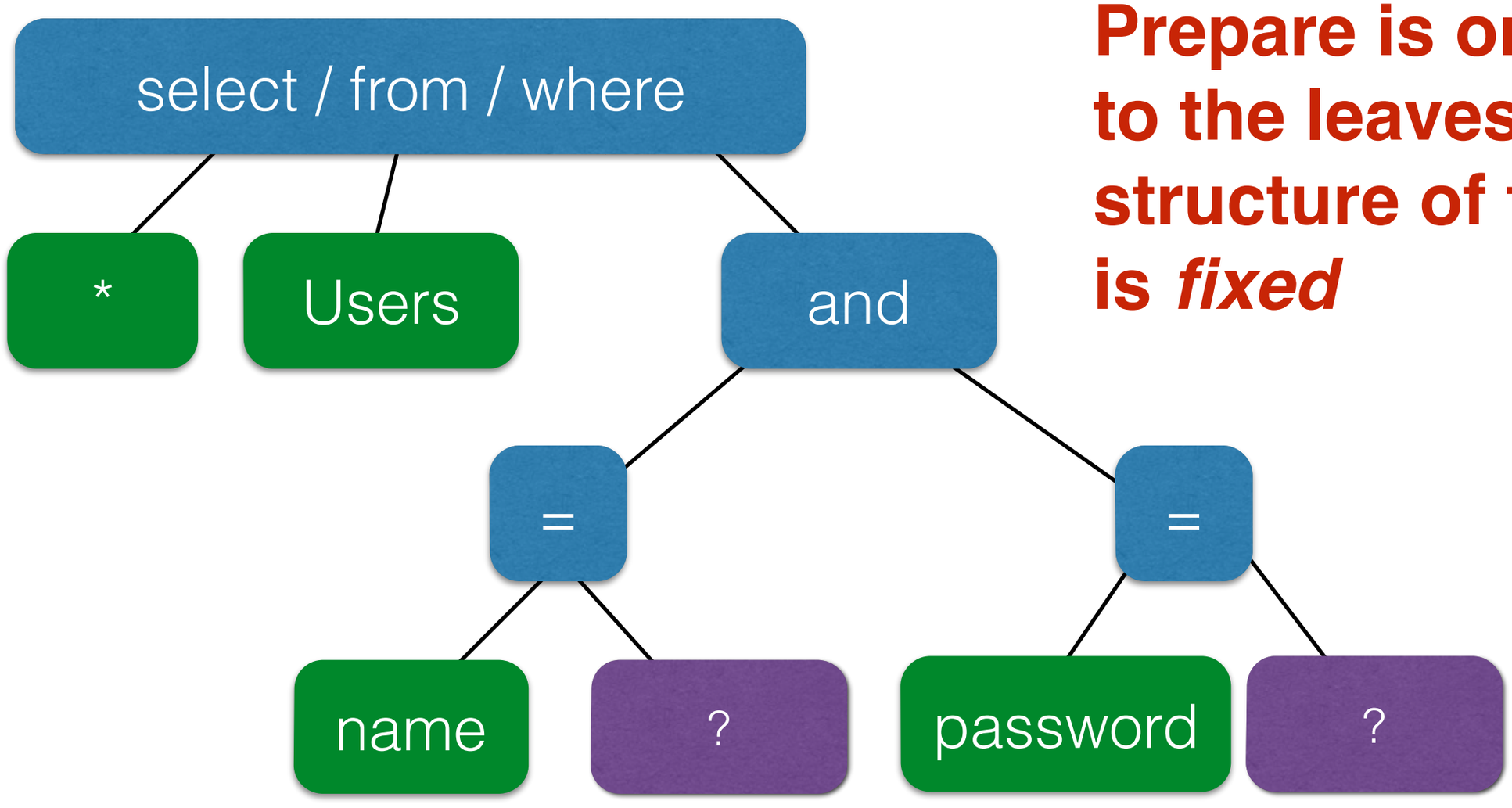
The underlying issue

```
$statement = $db->prepare(“select * from Users
     where(name=? and password=?);”);
```



The underlying issue

```
$statement = $db->prepare(“select * from Users
     where(name=? and password=?);”);
```



Prepare is only applied to the leaves, so the structure of the tree is *fixed*

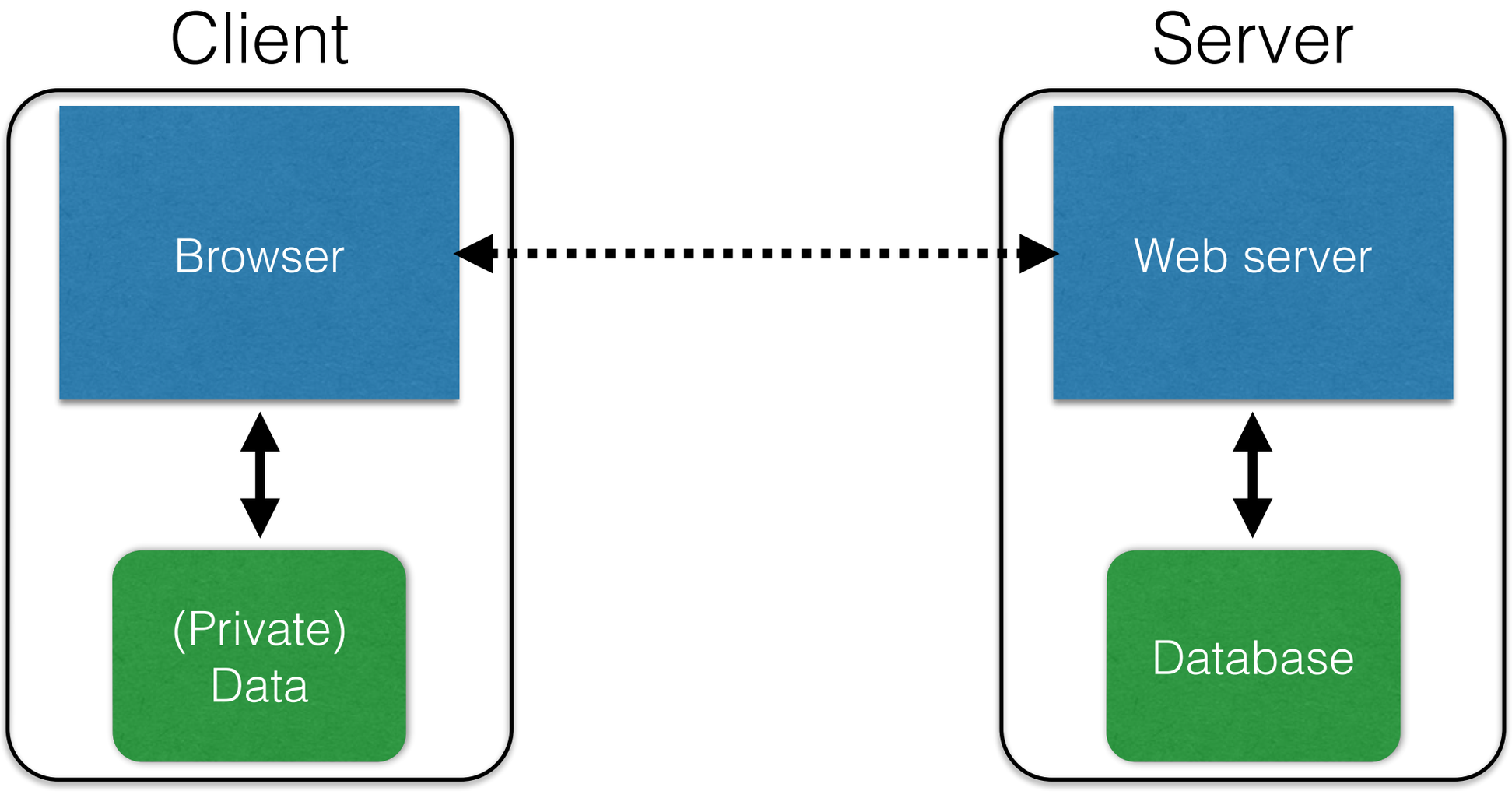
Mitigating the impact

- Limit privileges
  - Can limit commands and/or tables a user can access
    - Allow SELECT queries on Orders_Table but not on Creditcards_Table
  - Follow the principle of least privilege
    - Incomplete fix, but helpful

- Encrypt sensitive data stored in the database
  - May not need to encrypt Orders_Table
  - But certainly encrypt Creditcards_Table.cc_numbers

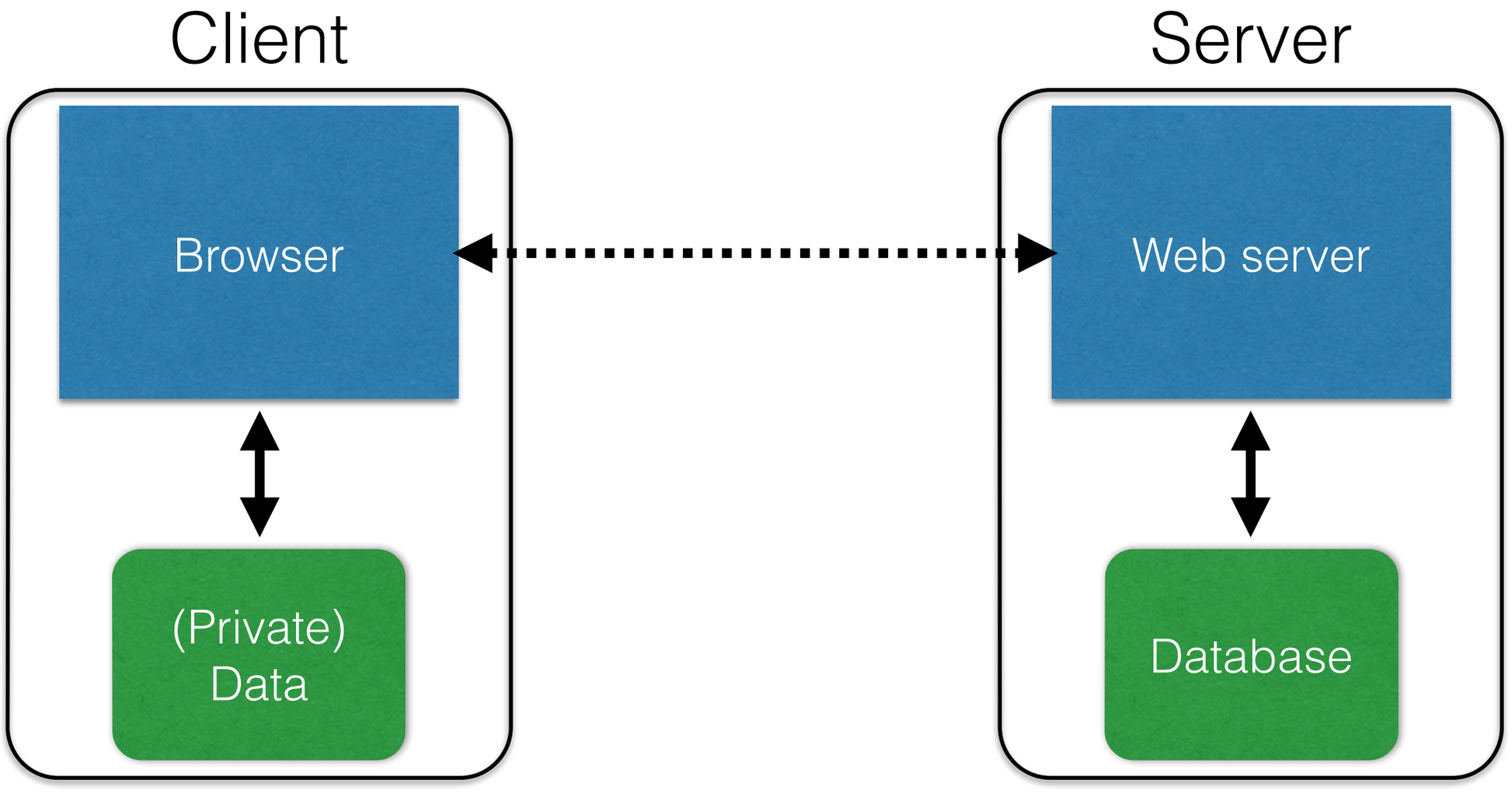
Web security

A very basic web architecture



DB is a separate entity, logically (and often physically)

A very basic web architecture



(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

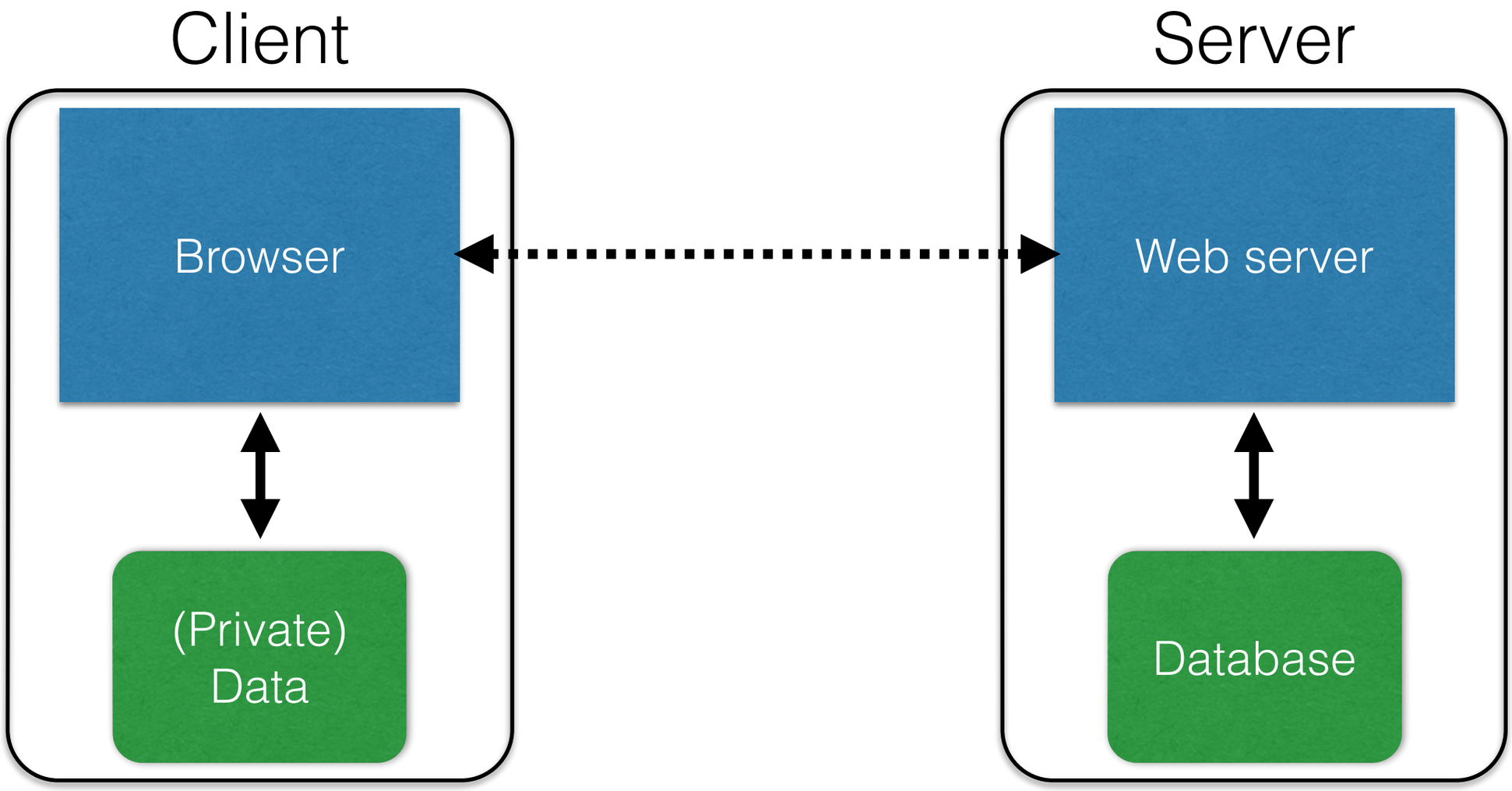
Here, the file home.html is static content
i.e., a fixed file returned by the server

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

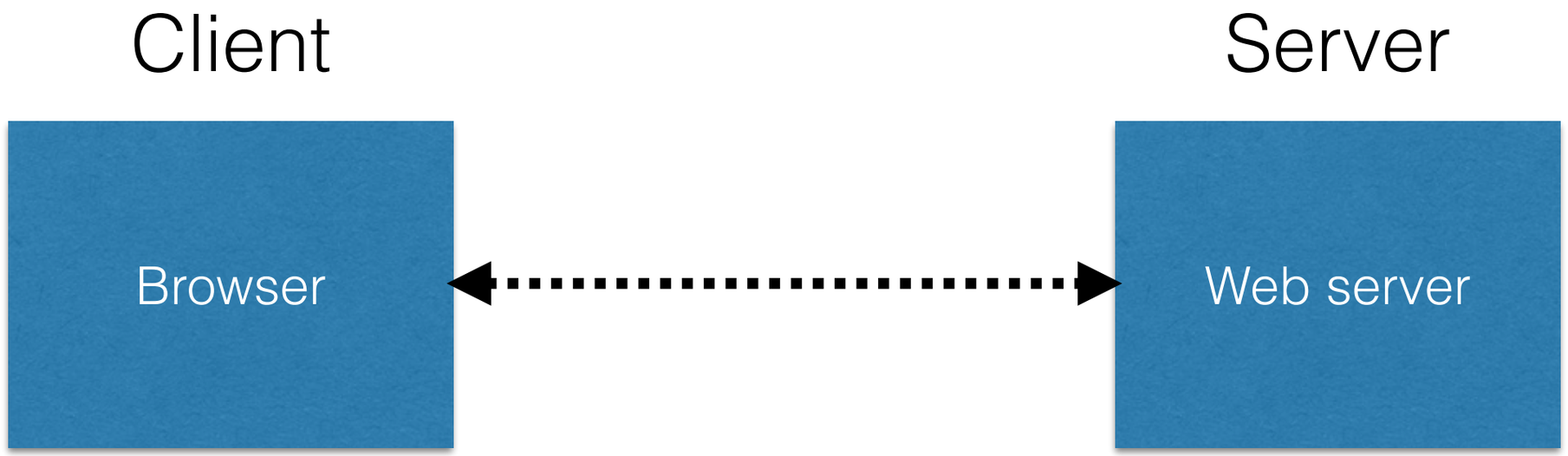
Arguments

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

Basic structure of web traffic

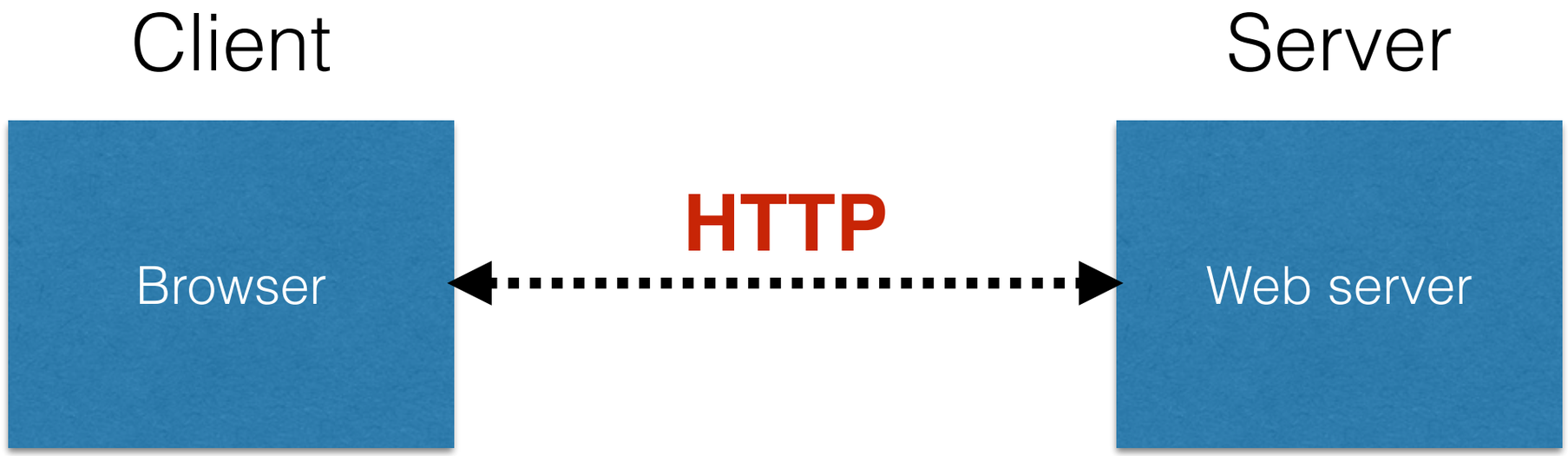


Basic structure of web traffic



Basic structure of web traffic

Basic structure of web traffic



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

Basic structure of web traffic

Client

Server

Browser

Web server

Basic structure of web traffic

Client

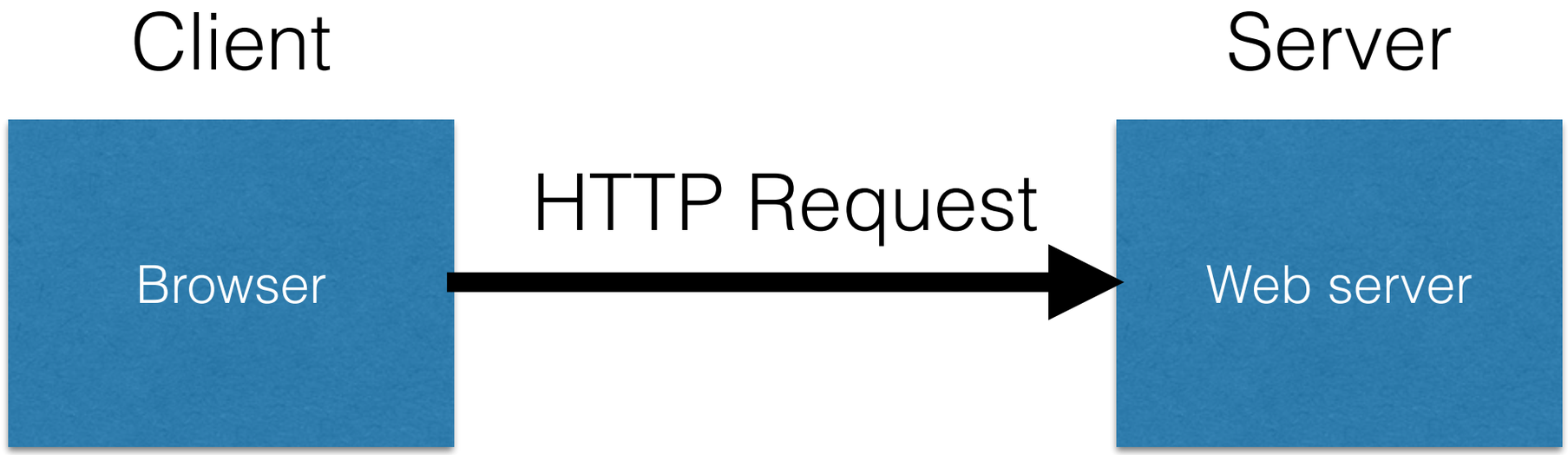
Server

Browser

Web server

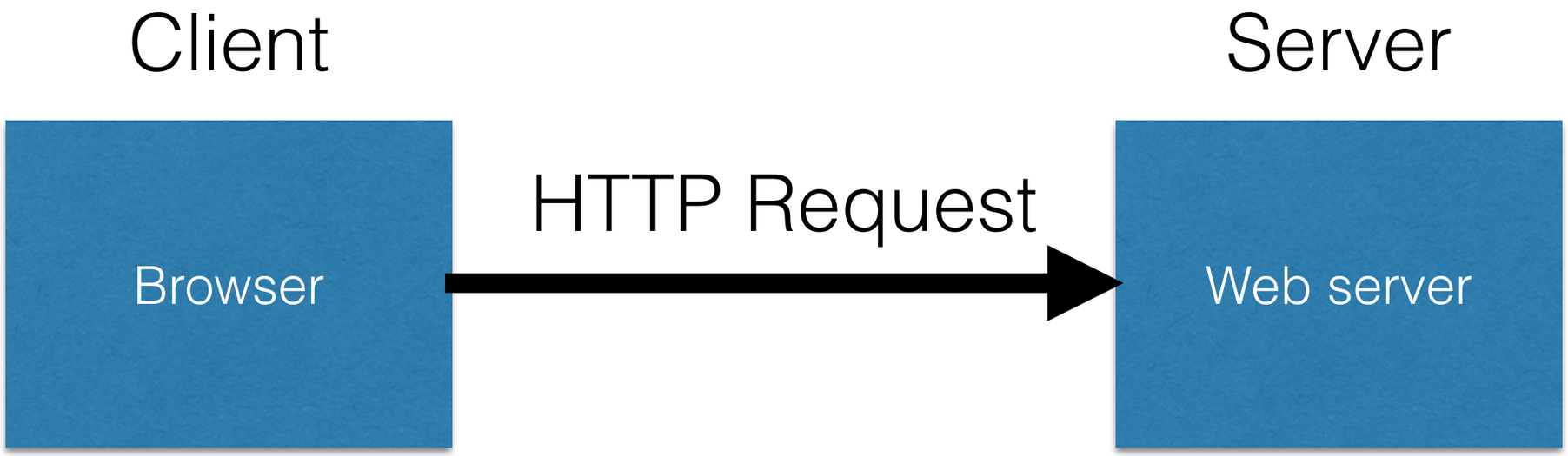
User clicks

Basic structure of web traffic



User clicks

Basic structure of web traffic



User clicks

- 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

HTTP Headers

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

GET /r/security HTTP/1.1
Host: www.reddit.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

HTTP GET requests

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

HTTP Headers

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

GET /r/security HTTP/1.1
Host: www.reddit.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

HTTP GET requests

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

HTTP Headers

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

GET /r/security HTTP/1.1
Host: www.reddit.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

User-Agent is typically a browser
but it can be wget, JDK, etc.

reddit **SECURITY** hot | new | rising | controversial | top | gilded | promoted

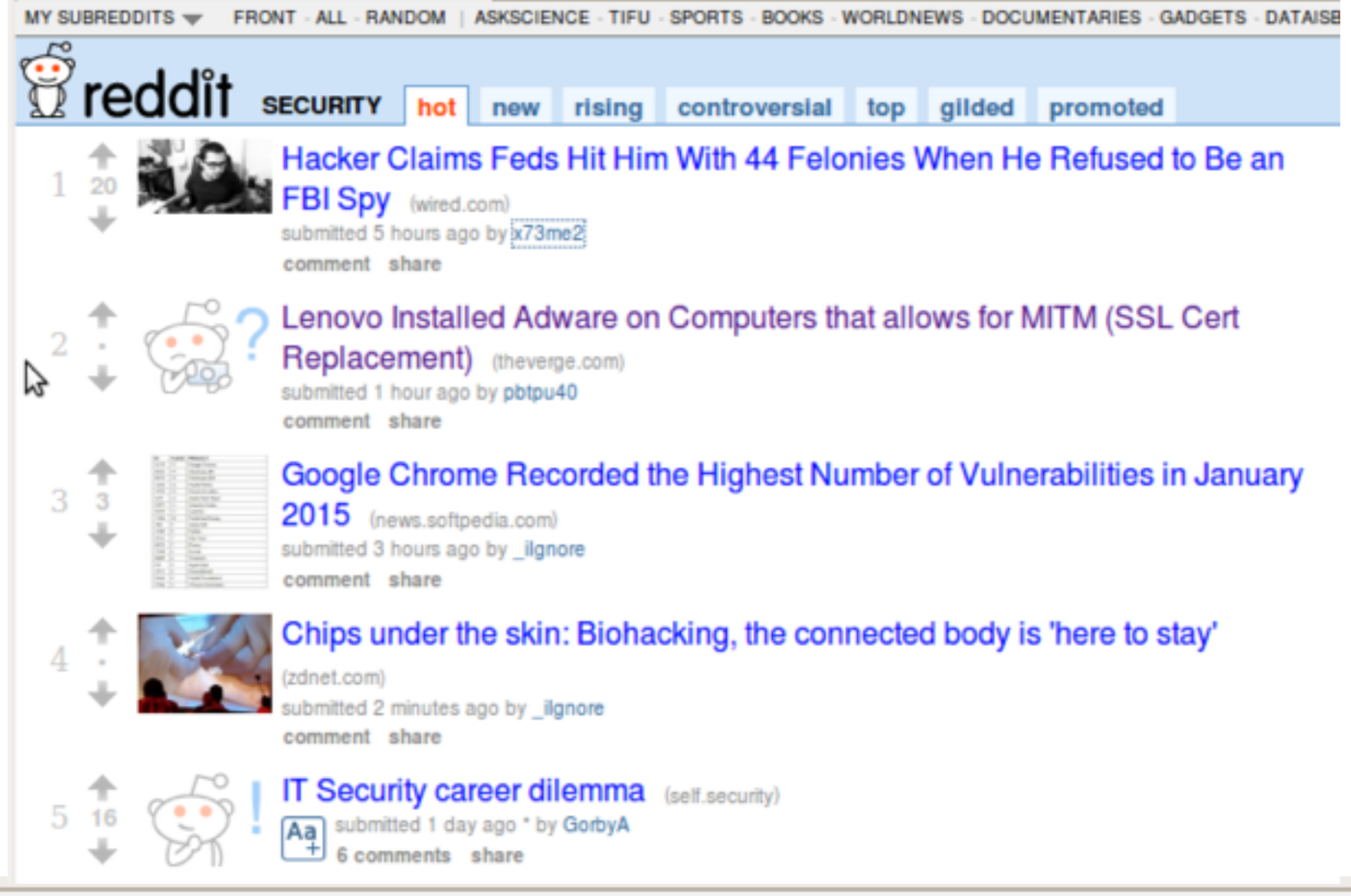
1 20 
Hacker Claims Feds Hit Him With 44 Felonies When He Refused to Be an FBI Spy (wired.com)
submitted 5 hours ago by x73me2
comment share

2 · 
Lenovo Installed Adware on Computers that allows for MITM (SSL Cert Replacement) (theverge.com)
submitted 1 hour ago by pbtpu40
comment share

3 3 
Google Chrome Recorded the Highest Number of Vulnerabilities in January 2015 (news.softpedia.com)
submitted 3 hours ago by _ilgnore
comment share

4 · 
Chips under the skin: Biohacking, the connected body is 'here to stay'
(zdnet.com)
submitted 2 minutes ago by _ilgnore
comment share

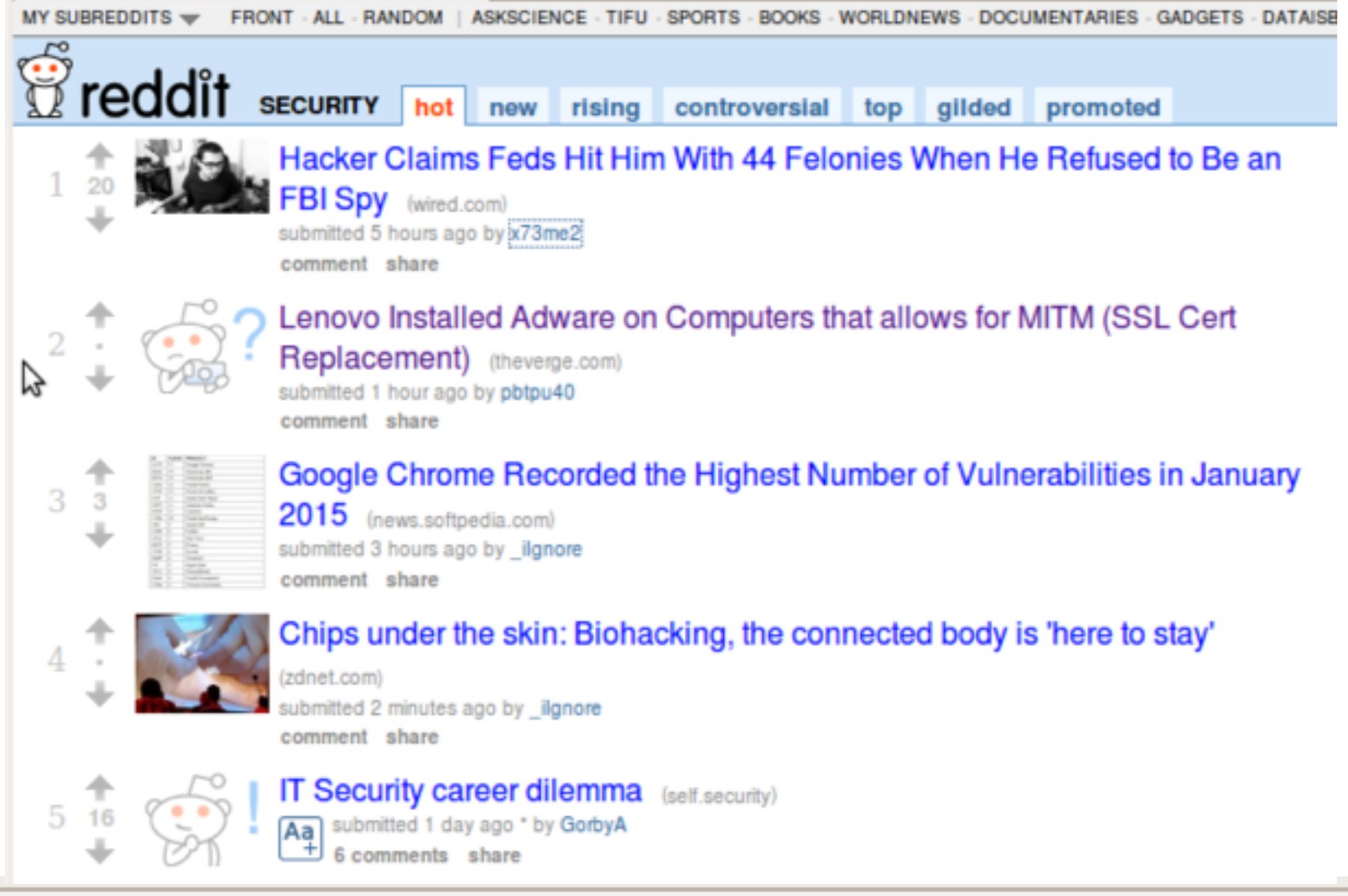
5 16 
IT Security career dilemma (self.security)
submitted 1 day ago * by GorbyA
6 comments share



HTTP Headers

http://www.theverge.com/2015/2/19/8067505/lenovo-installs-adware-private-data-hackers

GET /2015/2/19/8067505/lenovo-installs-adware-private-data-hackers HTTP/1.1
Host: www.theverge.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
Referer: http://www.reddit.com/r/security



HTTP Headers

http://www.theverge.com/2015/2/19/8067505/lenovo-installs-adware-private-data-hackers

GET /2015/2/19/8067505/lenovo-installs-adware-private-data-hackers HTTP/1.1
Host: www.theverge.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
Referer: http://www.reddit.com/r/security

Referrer URL: the site from which this request was issued.

HTTP POST requests

Posting on Piazza

HTTP Headers

https://piazza.com/logic/api?method=content.create&aid=i6ceq3skno48

POST /logic/api?method=content.create&aid=i6ceq3skno48 HTTP/1.1
Host: piazza.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: application/json, text/javascript, */*; q=0.01
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
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
Referer: https://piazza.com/class?nid=i55texo54nv3eh
Content-Length: 640
Cookie: piazza_session="
Pragma: no-cache
Cache-Control: no-cache

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

{"method":"content.create","params":{"nid":"i55texo54nv3eh","type":"note","subject":"Live HTTP headers","content":"<p>Starting today ...

HTTP POST requests

Posting on Piazza

HTTP Headers

https://piazza.com/logic/api?method=content.create&aid=i6ceq3skno48

POST /logic/api?method=content.create&aid=i6ceq3skno48 HTTP/1.1
Host: piazza.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: application/json, text/javascript, */*; q=0.01
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
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
Referer: https://piazza.com/class?nid=i55texo54nv3eh
Content-Length: 640
Cookie: piazza_session="
Pragma: no-cache
Cache-Control: no-cache

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

{"method":"content.create","params":{"nid":"i55texo54nv3eh","type":"note","subject":"Live HTTP headers","content":"<p>Starting today ...

HTTP POST requests

Posting on Piazza

HTTP Headers

https://piazza.com/logic/api?method=content.create&aid=i6ceq3skno48

POST /logic/api?method=content.create&aid=i6ceq3skno48 HTTP/1.1
Host: piazza.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: application/json, text/javascript, */*; q=0.01
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
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
Referer: https://piazza.com/class?nid=i55texo54nv3eh
Content-Length: 640
Cookie: piazza_session="
Pragma: no-cache
Cache-Control: no-cache

{"method":"content.create","params":{"nid":"i55texo54nv3eh","type":"note","subject":"Live HTTP headers","content":"<p>Starting today ...

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

HTTP Headers

https://piazza.com/logic/api?method=content.create&aid=i6ceq3skno48

POST /logic/api?method=content.create&aid=i6ceq3skno48 HTTP/1.1
Host: piazza.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: application/json, text/javascript, */*; q=0.01
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
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
Referer: https://piazza.com/class?nid=i55texo54nv3eh
Content-Length: 640
Cookie: piazza_session="
Pragma: no-cache
Cache-Control: no-cache

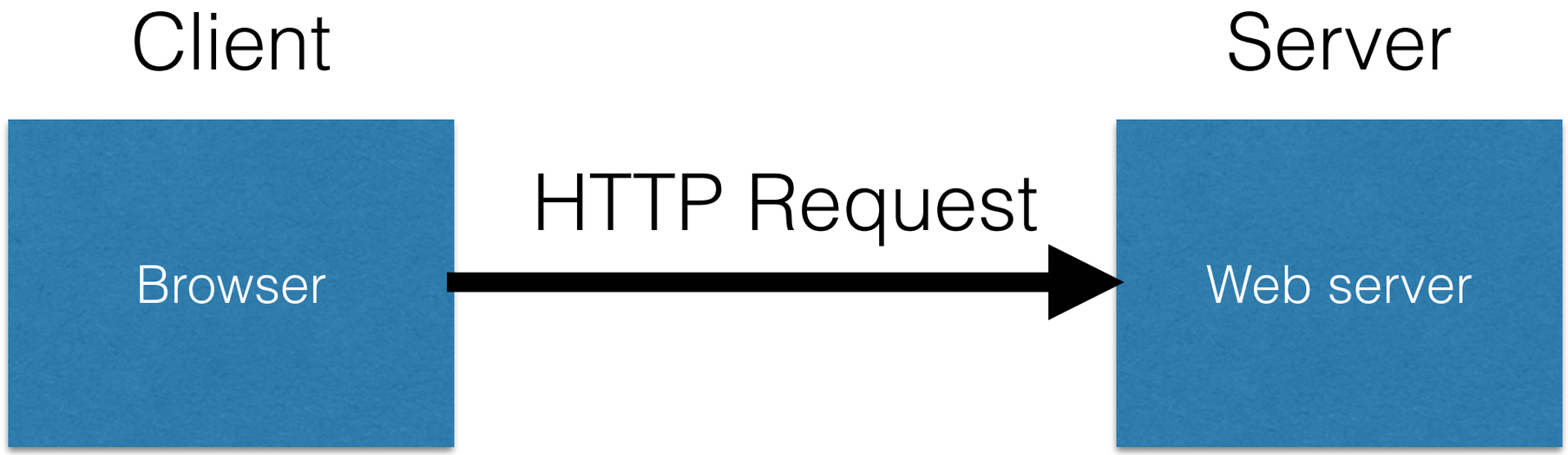
{"method":"content.create","params":{"nid":"i55texo54nv3eh","type":"note","subject":"Live HTTP headers","content":"<p>Starting today ...

Implicitly includes data as a part of the URL

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

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

Basic structure of web traffic



User clicks

Basic structure of web traffic

Client

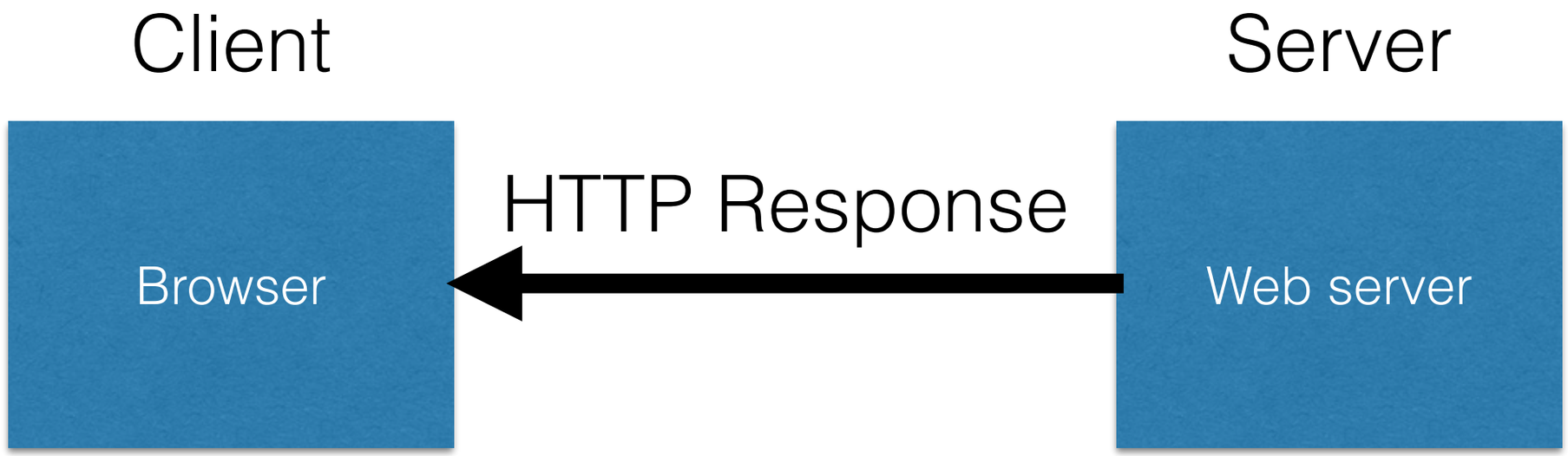
Server

Browser

Web server

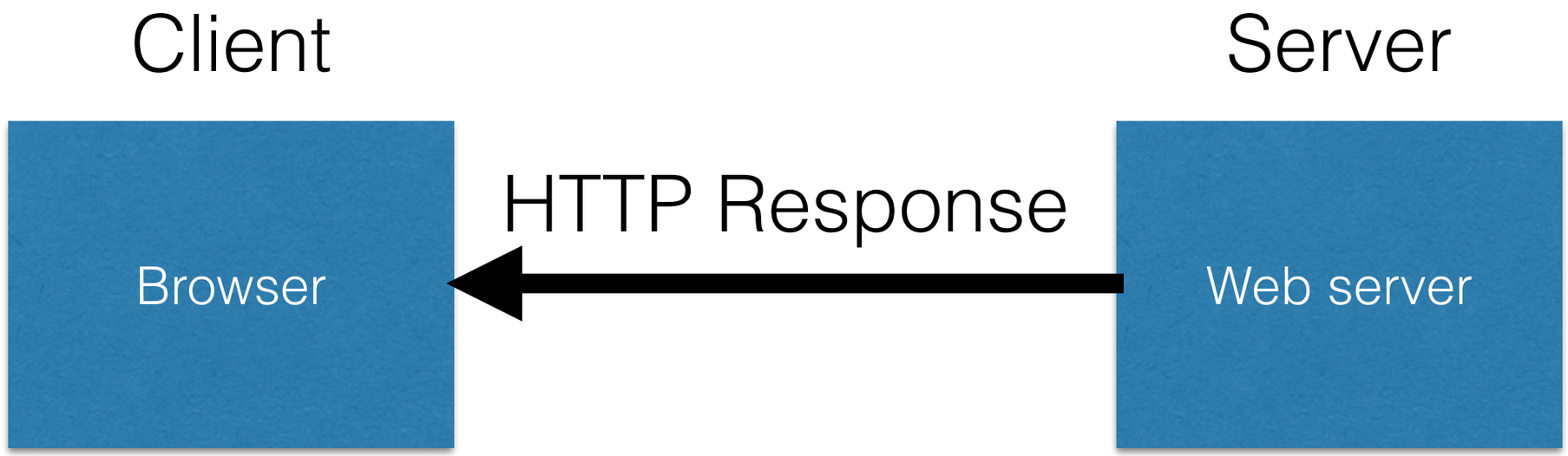
User clicks

Basic structure of web traffic



User clicks

Basic structure of web traffic



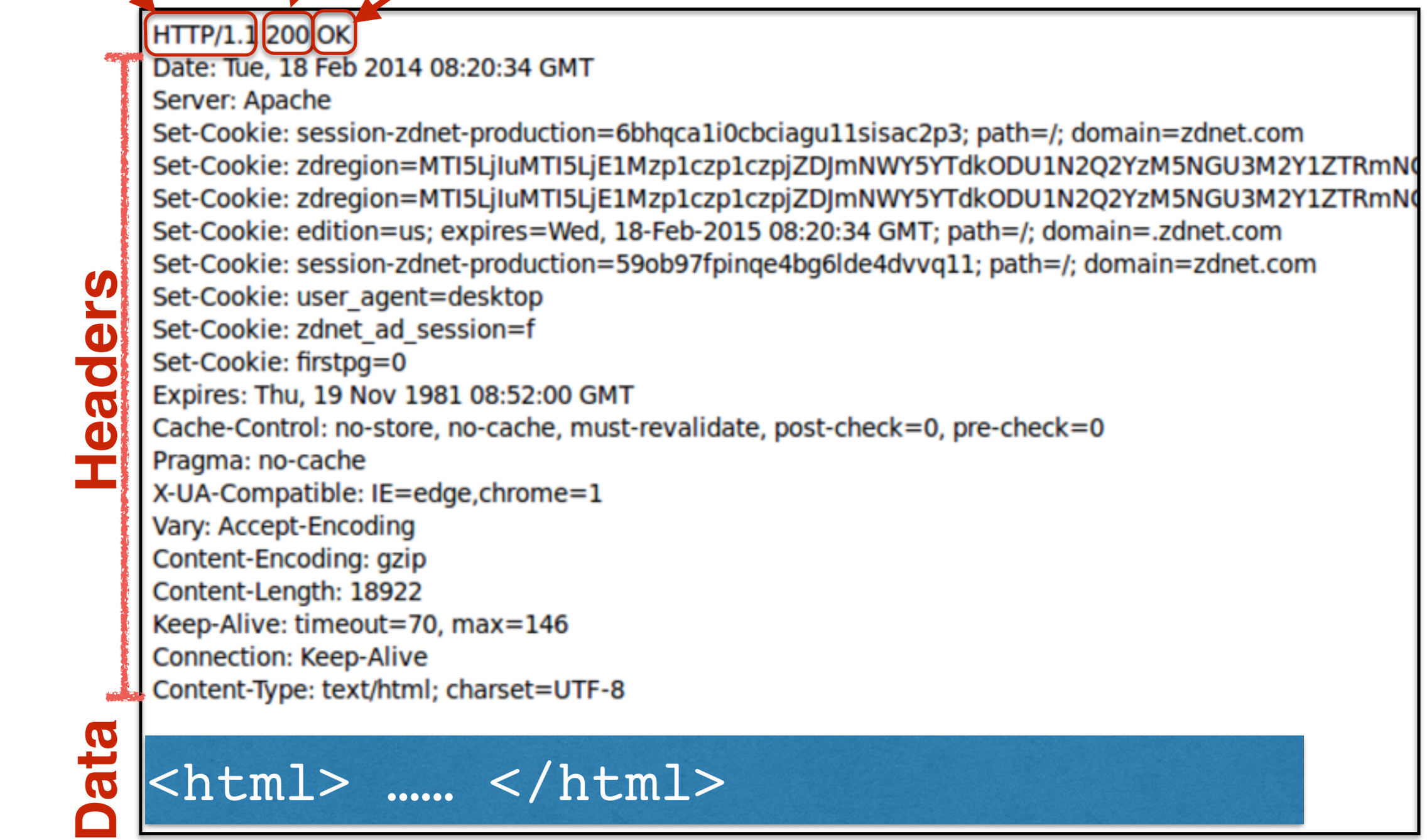
- Responses contain:
  - Status code
  - Headers describing what the server provides
  - Data
  - Cookies
    - State it would like the browser to store on the site's behalf

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=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
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 Headers

http://blog.lifars.com/2015/02/18/weird-security-term-of-the-week-clickjacking/

GET /2015/02/18/weird-security-term-of-the-week-clickjacking/ HTTP/1.1
Host: blog.lifars.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
Referer: http://www.reddit.com/r/security

HTTP/1.1 200 OK
Server: nginx
Date: Thu, 19 Feb 2015 17:25:28 GMT
Content-Type: text/html; charset=UTF-8
Transfer-Encoding: chunked
Connection: keep-alive
Vary: Accept-Encoding, Cookie
X-hacker: If you're reading this, you should visit automattic.com/jobs and apply to join the fun, mention this header.
X-Pingback: http://blog.lifars.com/xmlrpc.php
Link: <http://wp.me/p4BZPV-iV>; rel=shortlink
Last-Modified: Thu, 19 Feb 2015 17:25:28 GMT
Cache-Control: max-age=300, must-revalidate
X-nananana: Batcache
Content-Encoding: gzip

HTTP Headers

http://blog.lifars.com/2015/02/18/weird-security-term-of-the-week-clickjacking/

GET /2015/02/18/weird-security-term-of-the-week-clickjacking/ HTTP/1.1
Host: blog.lifars.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
Referer: http://www.reddit.com/r/security

HTTP/1.1 200 OK
Server: nginx
Date: Thu, 19 Feb 2015 17:25:28 GMT
Content-Type: text/html; charset=UTF-8
Transfer-Encoding: chunked
Connection: keep-alive
Vary: Accept-Encoding, Cookie
X-hacker: If you're reading this, you should visit automattic.com/jobs and apply to join the fun, mention this header.
X-Pingback: http://blog.lifars.com/xmlrpc.php
Link: <http://wp.me/p4BZPV-iV>; rel=shortlink
Last-Modified: Thu, 19 Feb 2015 17:25:28 GMT
Cache-Control: max-age=300, must-revalidate
X-nananana: Batcache
Content-Encoding: gzip

HTTP Headers

http://blog.lifars.com/2015/02/18/weird-security-term-of-the-week-clickjacking/

GET /2015/02/18/weird-security-term-of-the-week-clickjacking/ HTTP/1.1
Host: blog.lifars.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
Referer: http://www.reddit.com/r/security

HTTP/1.1 200 OK
Server: nginx
Date: Thu, 19 Feb 2015 17:25:28 GMT
Content-Type: text/html; charset=UTF-8
Transfer-Encoding: chunked
Connection: keep-alive
Vary: Accept-Encoding, Cookie
X-hacker: If you're reading this, you should visit automattic.com/jobs and apply to join the fun, mention this header.
X-Pingback: http://blog.lifars.com/xmlrpc.php
Link: <http://wp.me/p4BZPV-iV>; rel=shortlink
Last-Modified: Thu, 19 Feb 2015 17:25:28 GMT
Cache-Control: max-age=300, must-revalidate
X-nananana: Batcache
Content-Encoding: gzip

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*

Next time

Continuing with

Web Security

Cookies

XSS & CSRF

Required reading for next lecture:

"Web Security: Are You Part Of The Problem?"

"Cross Site Request Forgery: An Introduction…"

Optional reading for this lecture:

"SQL Injection Attacks by Example"