ECS 189
WEB PROGRAMMING

5/24

File-system security

- There has been a good discussion of security on Piazza
- One simple thing: make your directory on the server unreadable to your classmates (Alex and Loc suggested this on Piazza as soon as we got these accounts)

chmod 700 .

….from your home directory.

Our servers have no security

- Any browser can access any server
- If your server is running, it is willing to hand out any file from /public
- This includes your photos.js, which contains some of the code you wrote for this assignment
- In general, browser code is public; even for a (hopefully) very secure site like Bank of America, you can see the source (HTML, CSS, Javascript) of any Web page you download

Upload

- Allowing users to upload files to a site introduces all kinds of possible security problems
- For instance, they could upload something that overwrites an important file, either bringing down the server or making it try to attack any browser that uses it

Backups!

- You're doing a big project on a system with no security and no backups!
- Even if you're not attacked, it is so easy to accidentally delete an important file the night before the project is due…
- At least every day, copy your files to somewhere safer (your laptop, a directory at school, a (private) git repository)

Improving security for uploads

```
form.on('fileBegin', function (name, file){
  if ((file.type == "image/jpeg") | (file.type ==
  "image/png")) {
    file.path = __dirname + '/public/' + file.name;
    localFilename = file.name;
  } else {
    console.log("cannot upload type "+file.type);
  }
```

Some other protections

- Only accept alpha-numeric filenames with one dot
- Put uploaded files in a different directory from any code
- Run uploaded files through virus protection before storing permanently
- Set maximum file upload size to prevent denial-of-service attacks

Ethics

- Is it ethical to tamper with a classmate's project, possibly ruining their grade, in order to teach them a lesson about being careless with security?
- What is a professional course of action when you discover a security violation?

Private data

- On a real photo sharing site, different users would have different collections of photos, some of which might be private
- How come I see my photos when I go to Flickr, and you see yours?

Private data

- On a real photo sharing site, different users would have different collections of photos, some of which might be private
- How come I see my photos when I go to Flickr, and you see yours?
- We need to log in!
- Making users log in lets you:
  - give them private data
  - differentiate what they see (my photos, not all photos)
  - maybe charge money!

Authentication on our site

- Alex suggests this node module:

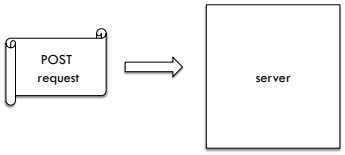
npm install basic-auth

- Adds a (very simple kind of) login and password to your app on the splash page
- This will makes it harder for people to get to your Upload button
- Does a login prevent them from getting to the Upload button?

Maybe!

- If your main page is a separate html page (easiest!) then they might know or guess its name and go there directly, skipping the splash/login page
- You could have your app send this second page, or perhaps just modify the first page, in the reply to an AJAX request
- AJAX is more secure than public static files
- Does preventing them from getting to your Upload pages prevent them from uploading?

Not entirely



- Server forgets about users, accepts any POST request it gets
- Hacker can still put together an HTTP request, type POST, addressed to server, with a file in the body

More secure login

- Login to a secure site, eg. BofA, will
  - use a more complicated login,
  - encrypt traffic going back and forth, and
  - issue a session cookie to your browser.
- Browser remembers the session cookie, and includes it in future AJAX requests to BofA, until tab is closed.

But even simple login helps!

- When they can get to your Upload button, it makes it super-easy to create an evil POST request.
- Can they just target their Upload button code to send requests to your server?

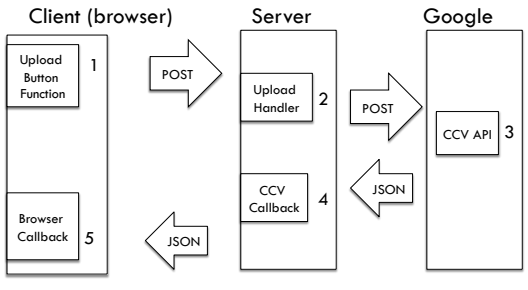
But it gets harder!

- When they can get to your Upload button, it makes it super-easy to create such a POST request.
- Can they just target their browser Upload button code to send requests to your server?
- Not really! Browsers only send AJAX requests to **their own servers**.

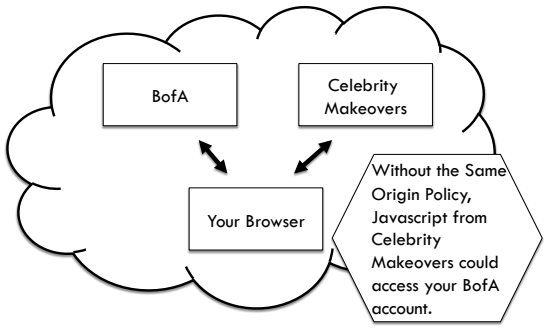
Same Origin Policy

- Mainstream browsers implement the Same Origin Policy – AJAX requests can only go to the same server the Web page came from.
- So in general, if we want to use a 3rd party API, we can't access it directly from the browser. We need to go through the server

Path to get data from API



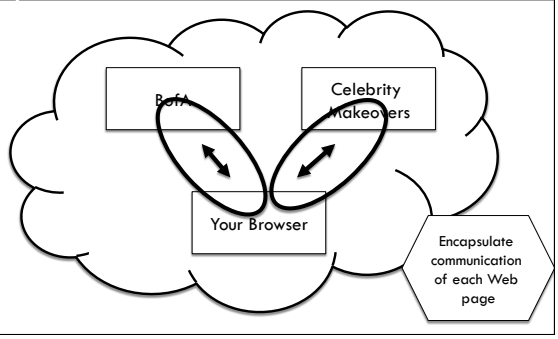
Cross Site Scripting Attack



How would that work?

- You log into BofA, or maybe some site that does not have such good security (eg. no session cookies)
- Then you open a new tab at Celebrity Makeovers
- If there were no same-origin policy, CM's Javascript could try accessing BofA, all the time, just in case it discovers that you are logged in.
- When it gets lucky, it sends the hackers a big check.

Same Origin Policy



But….???

- How did we do that Weather App?
- Weren't we using an API to get the weather from the browser?
- We didn't use a server in the middle to satisfy the Same Origin Policy…?

JSONp

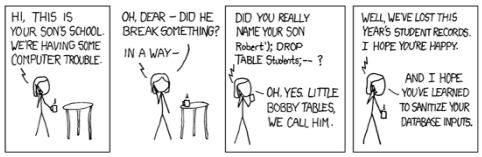
- JSONp lets you get JSON from servers other than your own.
- Uses an exception to the Same Origin Policy – you can use Javascript libraries on any Web page.
- Yahoo Weather sets up a JSONP service to make its weather data public.

Another classic

- Allow users to access a database over the Web
- For instance, let users get arbitrary data out of our database
  - Let them enter SQL commands, we run them on server, send results back
  - Terrible idea; why?

Protecting a database

- You want only your users to access the database; any whacko can get there by sending queries to your server, and destroy data.
- Attempt 2: Users enter search term in a text form, sends it to server, and we put it into server database for them
  - In server code, take user data and put it into a SELECT command
  - Terrible idea; why?



How to use arbitrary user input?

- Rather than building a command string using arbitrary user input, use the SQL PREPARE command.
- Takes input, checks it carefully and then pastes it into the command.
- This is an example of sanitization.
- Almost always better to use a sanitization function associated with your database or framework than to write one yourself.