CSE 409, Fall 2011, Rob Johnson, http://www.cs.stonybrook.edu/~rob/teaching/cse409-fa11/
Alin Tomescu, October 21st, 2011

Mashups

Mashups

The challenge comes when the browser has credentials for service A and service B and the mashup service needs to access those credentials to talk to service A and service B.

Same origin policy: If you're loading a page from the mashup server, then that page cannot communicate with service A or service B.

You can put a <script src="serviceA.com/script.js"> in your mashup page. That will load the JavaScript code, but the same origin policy still applies and the script is run in the security domain of the mashup.

<ifame> tags

`<iframe>` tags provide a solution. An `<iframe>` is like a separate webpage within a webpage.

The mashup can have **two iframes**, one for service A and one for service B. Each iframe has a different origin and the browser will not let JavaScript in one iframe to manipulate anything going on in another iframe or the main page, which can also be thought of as an iframe.

This is the **downside**: There is no interaction or communication between the iframes and the main page. This is only useful for portal websites that want to display information from a bunch of websites or stuff like iGoogle.

HTML5 post message

HTML5 introduced a message posting API that allows two or more iframes to send each other messages on the client side (on the browser).

You can have a page with an outer iframe (mashup.com) and an inner iframe (facebook.com) and by communicating via PostMessage the inner iframe can receive messages from the outer iframe and respond with a result.

Problem: The facebook.com iframe needs to authenticate the outer mashup.com iframe.

- The browser has to tell the receiving iframe where the message is coming from (similar to IPC setups like Binder).
- This way, since the browser is trusted, evil.com won't be able to lie and say it's mashup.com

What if you also added a bing.com iframe that trusted facebook.com. These two would like to talk, but they can't just trust mashup.com to forward messages from one to the other.

The mashup.com page can request the browser to open a "socket" for the bing.com iframe and give it to the facebook.com iframe. Somehow the browser has to correctly "let go" of the socket so that he does not listen in on the conversation. If that browser is Google Chrome, some concerns might arise :P

Defenses against bugs

They happen at many stages in the software engineering process:

CSE 409, Fall 2011, Rob Johnson, http://www.cs.stonybrook.edu/~rob/teaching/cse409-fa11/
Alin Tomescu, October 21st, 2011

- **Design:** separation of privileges in our UNIX printing scheme
- **Coding:** banning the use of `sql.query`, automatic code auditing
- **Testing:** fuzzing
- **Prevent bug exploit:** Deploy stuff like ASLR or ASCII armoring to prevent missed bug exploits
- **Damage mitigation:** Intrusion Detection System, chroot jails, VMs, they all help mitigate the damage once the attacker broke your software and he is in