Enabling Un-trusted Mashups

c70n3r@gmail.com
Talk Outline

- Definition
- Web Mashups
- Mashup Vulnerabilities
- Secure Mashup Principles & Technologies
- Solution: Today and Tomorrow
What is a Mashup?

- An experience delivered by combining properties of more than one source.
Web Mashup: Types

- An app that combines services from multiple origins to create new experiences
- Mostly based on DHTML. Also, the focus of this talk.

Mashup Approaches:
- Client-side: Browser retrieves and aggregates as per the provided template
- Server-side: Backend aggregation then serving pre-composed mashup to the browser
- Hybrid: Leverages benefits of approaches above

Note: From here on we refer web mashups as mashups
Ads were and will remain to be the backbone of Internet business model
– User behavioral targeting, social and interactive ads are set to revolutionize further

Social plugins and 3rd party widgets help drive engagement and rich user experience

Need to grow. Getting viral. Enter app platforms. The ultimate manifestation of user generated content in mashups – FB, YAP, iGoogle
Traditional tech stack provides little or no control over the embedded service

- Intrinsically insecure model
- Ads do go bad. Widgets do get compromised.

- But we have a legal agreement?
  - Yes, that’s just a partial reactive solution.
  - What about 3rd part developer? Does the agreement going to stop him sitting in *^%$(!, %$@#),......
Web Mashup: Client-side with Ads and Social Plugins

Dil Toh Baccha Hai Ji

Dil Toh Baccha Hai Ji is a 2011 Hindi romantic comedy film, directed by Madhur Bhandarkar. It stars Shazahn Padamsee, Shruti Haasan and Shraddha Das in the lead roles.
Web Mashup: Client-side with 3rd Party developers

http://www.facebook.com/Levis

Levi's Store - SoHo
Shopping & Retail
New York, NY

Like · Get Directions · Edit

With your help, we can raise awareness of a global crisis and bring clean water to up to 8,000 people—for life. All we need is 100,000 pledges to make it happen.

MORE ABOUT WATER.ORG

Did you know that nearly 1 billion people don't have access to safe, clean water? Water.org does, and they're doing something about it. So can you. Join us as we partner with Water.org to assist people around the globe in obtaining sustainable water and sanitation systems. Pledge your support today and help create a world where every person can have access to this basic human right.
Web Mashups: Hybrid with 3rd party developers & providers
Server-side: No standard solutions. Mostly custom implementations, proxies and hacks

Client-side: The more popular mashup. Leverages browser SOP (Same Origin Policy).
- Two solutions: Scripts and iframes
  - Script based
    - Offers NO separation but provides FULL interaction
    - Interaction not authenticated, nor can confidentiality or integrity be ensured
  - Iframe based
    - FULL separation between cross origins
    - NO separation within the same origin
    - NO provision for interaction between components
Mashup Vulnerabilities: iframe based

- **Malicious Redirection**
  - `top.location = http://s0m3phishing.com`
- **Fake / Malicious UI**
  - `<form method=...>, window.open()`
- **Drive-by Downloads/Malware**
  - `Content-Disposition: attachment`
- **Denial of Service (DoS) and Noise**
  - `Infinite alert() and while loops`
- **History Sniffing/Mining**
  - `getComputedStyle()`
- **Referrer Leak**
  - `Referrer: http://<ip>/r.html?a=secret&b=private`
- **LAN Scanning**
  - `<img src=http://10.0.0.1 onerror=...>`
Mashup Vulnerabilities: Script based

- Steal Username, Password and other secret data by calling, intercepting or spoofing DOM events like `onsubmit`
- Steal cookies via `document.cookie`
- Malicious GET and POST via `xhr.open`
- Abuse features like `autocomplete`
- All iframe vulnerabilities
- And, many more……
Redirection Attack: FB iframe tab
Redirection Attack: igoogle gadgets

A script which is inlined on a page can redirect the page.

Result:

Do Evil Redirect
Fake Login Pop-up
Drive-by Downloads
Warning: Something's Not Right Here!

69.50.201.152 contains malware. Your computer might catch a virus if you visit this site.

Google has found malicious software may be installed onto your computer if you proceed. If you’ve visited this site in the past or you trust this site, it’s possible that it has just recently been compromised by a hacker. You should not proceed, and perhaps try again tomorrow or go somewhere else.

We have already notified 69.50.201.152 that we found malware on the site. For more about the problems found on 69.50.201.152, visit the Google Safe Browsing diagnostic page.

Go back

If you understand that visiting this site may harm your computer, proceed anyway.

Help improve detection of malware by sending additional data to Google about sites on which you see this warning. This data will be handled in accordance with the Safe Browsing privacy policies.
DoS / Noise
DoS / Noise
History Sniffing/Mining

Promote your Facebook Page with the Contact Importer for Pages
Already have a list of customers, members or email subscribers? Use the Contact Importer tool to upload your list, find them on Facebook, and let them know about your Facebook Page.

History_iiframe ➤ History

Community  ➤ Edit Info

A script can examine a link's computed style to deduce whether the user has visited that site.

The following link is red if you've visited www.yahoo.com recently and blue if you haven't: link

Does getComputedStyle say you've visited http://yahoo.com?

Yes!
Referrer Leak

iframe referrer mis-use

Date: 2011-08-30 03:55:25
IP:
Referrer: http://79.125.60.136/ref.html?a=secret&b=confidential&c=private
Browser: Mozilla/5.0 (Windows NT 5.1) AppleWebKit/535.1 (KHTML, like Gecko) Chrome/13.0.782.112 Safari/535.1
Secure Mashup Principles

- Integrator responsible to enforce intended separation and interaction
- **Separation** – Deter interference and malicious behavior of rogue components
- **Interaction** – Enable secure cross-document messaging
Secure Mashup Principles: Client-side Developments

- **Separation: Iframe sandbox**
  - When set, enables new restrictions on any content hosted by the iframe
  - By default, the content is treated as being from a unique origin, forms and scripts are disabled, links are prevented from targeting other browsing contexts, and plugins are disabled

- **Interaction: postMessage**
  - Secure & improved replacement of Fragment Identifier Messaging (FIM)
  - Provides controlled and safe cross-document messaging between iframes
  - Enables authentication, other than confidentiality and integrity that FIM provided too
  - Authentication achieved by browser validating the destination when sending a message and the recipients ability to validate sender on message receive.

- **Content Security Policy (CSP)**
  - Primarily designed to defend against XSS, as a side-effect, enables better mashups
  - Provides better granularity over authority of components by restricting their capabilities that make certain attacks possible
    - `eval()`, `setTimeout()`, `javascript:`, `new Function()`, `onclick()` and the likes are restricted
Secure Mashup Principles: iframe Sandbox
Secure Mashup Principles: iframe Sandbox

**Pros**
- Simplicity and low learning curve
- Incremental features lead to better adoption within devs
- Better adoption by browser vendors being part of HTML5 std
- Completely backward compatible
- Minimal hooks needed on existing code base
- Excellent support on mobile browsers

**Cons**
- **Not production ready!**
  - No support in older browsers (IE6 must die!)
  - Not a replacement but complement to existing defenses
  - Supported only in Chrome. FF proposes through CSP but yet to be implemented. No support in IE
Secure Mashup Principles: postMessage
Secure Mashup Principles: postMessage

Pros

- Nearly production ready
  - Supported by all major browsers (FF, Chrome, and IE8+ covers 85% browser market share)
- Simplicity and low learning curve
- Better adoption by browser vendors being part of HTML5 std
- Excellent support on mobile browsers

Cons

- No support in older browsers
Secure Mashup Principles: CSP

Pros
- Good granular control over authority of components
- Simplicity and low learning curve

Cons
- Not production ready!
  - No support in older browsers
  - A Mozilla standard supported only by FF4+. Chrome is next. No word on IE
  - Moving spec. Needs more scrutiny
Secure Mashup Principles: Server-side Developments

- **Interaction** delegated to browser SOP
- **Separation** introduced by implementing object capability model i.e. an object cannot be created if there is no reference to it
  - Achieved by restricting JavaScript (JS) to a subset and providing run-time control over specific operations, such as DOM access.
- Popular implementations include – Caja, FBJS, AdSafe and Web Sandbox
- Caja better adopted and supported (YAP, iGoogle, Orkut) compared to others
Secure Mashup Principles: Caja
Secure Mashup Principles: Caja

Pros
- Production ready
- Excellent granular control over authority of components
- Excellent protection against most of the DHTML mashup attacks

Cons
- High learning curve for developers
  - Compiling and debugging challenges
  - the line numbers don’t correspond to line numbers in your source code.
- Not a standard
- Limited adoption over the years
- Limited support for JS libraries
- Limited protection against JS DoS conditions
- Quirky support on old browsers (Again, IE6 must die!)
- Performance hit at compilation and run-time due to virtualization
Secure Mashup Principles: Caja Virtualization

- Replaces references to real global variables with references to per-sandbox globals
- Rewrites references to this to prevent access to the real global scope
- Replaces most JavaScript code with semantically similar code that has runtime checks for security
- Rejects some JavaScript code early, such as `with(obj){...}`

Here's an example transformation. This JavaScript source code:

```javascript
size = 3;
function arf(geo, out) {
    var s4 = geo.compute(4 * size);
    var s5 = geo.compute(5 * size);
    out.value = (s4+s5)/2;
    return this;
}
```

is cajoled into something like this:

```javascript
$v.so('arf', (function () {
    function arf$_caller($dis, geo, out) {
        var s4 = $v.cm(geo, 'compute', [ 4 * $v.ro('size') ]);$
        var s5 = $v.cm(geo, 'compute', [ 5 * $v.ro('size') ]);$
        $v.s(out, 'value', (s4+s5)/2);
        return $dis;
    }
    $v.so('arf', 3);
});
```
Secure Mashup Principles: Caja – JS DoS Attack
Solution: Today

3rd Party Partners
- Keep doing what you are doing to sanitize at the server-side
- **Iframe** Ads, Widgets and other content. Avoid scripting
- Keep signing/updating legal and security agreements

3rd Party Developers
- Your choice!
  - Minimal policing, low learning, high portability (iframe) leads to high growth and viral networks
  - Policing (Caja, FBJS), high learning curve, low portability - low growth
3rd Party Partners

- One day. Some day. Once those are dead and buried. Yes, you can leave it to them - Sandbox, postMessage, CSP.
- Some would still need exceptions
  - Iframe Ads, Widgets and other content. Avoid scripting
- Keep signing/updating legal and security agreements

3rd Party Developers

- Iframe sandbox + postMessage + CSP + <?>
References

Web Links
- FB Iframe Tabs http://developers.facebook.com/blog/post/462
- HTML5 Demos http://html5demos.com/
- Mozilla CSP http://people.mozilla.com/~bsterne/content-security-policy
- Google Caja http://code.google.com/p/google-caja/
- Caja Playground http://caja.appspot.com/
- Caja DoS http://code.google.com/p/google-caja/issues/detail?id=1406&sort=-id
- Microsoft Web Sandbox http://www.websandbox.org/
- Browser Security Features http://www.browserscope.org/?category=security&v=1

Creative Commons Image Attribution
- Slide 1 http://www.flickr.com/photos/donkeyhotey/5679642883/sizes/o/in/photostream/
- Slide 3
  - http://www.flickr.com/photos/jesse_sneed/2383953694/
  - http://www.flickr.com/photos/31856336@N03/4848321266/
  - http://www.flickr.com/photos/umpcportal/4147481868/