Crossing the Chasm
Pitching Security Research to Mainstream Browser Vendors

Collin Jackson
Carnegie Mellon University
Why a security feature is like a startup

Geoffrey Moore’s ‘Crossing the Chasm’ diagram
(circa 1991)

<10 users  ~1 million users  >1 billion users
Ideas trying to cross the chasm

For every idea here there are 100 that never got any adoption
Good ideas get adopted very quickly

<table>
<thead>
<tr>
<th>Name</th>
<th>Score</th>
<th>postMessage</th>
<th>JSON.parse</th>
<th>toStaticHTML</th>
<th>X-Frame-Options</th>
<th>X-Content-Options</th>
<th>Block reflected XSS</th>
<th>Block location spoofing</th>
<th>Block JSON hijacking</th>
<th>Block XSS in CSS</th>
<th>Sandbox attribute</th>
<th>Origin header</th>
<th>Strict Transport Security</th>
<th>Block cross-origin CSS attacks</th>
<th>Cross Origin Resource Sharing</th>
<th>Block visited link sniffing</th>
<th>Content Security Policy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chrome 13</td>
<td>16/17</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>Chrome 14</td>
<td>15/17</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>RockMelt 0.9</td>
<td>15/17</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>IE 9</td>
<td>13/17</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>iPhone 4.2</td>
<td>13/17</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>Safari 5.0</td>
<td>13/17</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>Android 3.1</td>
<td>12/17</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>Firefox 5</td>
<td>12/17</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>Firefox 6</td>
<td>12/17</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>Firefox Beta 7</td>
<td>12/17</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>IE 9</td>
<td>11/17</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>Android 2.3</td>
<td>10/17</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>iPhone 3.1</td>
<td>8/17</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>Opera 11</td>
<td>6/17</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>no</td>
</tr>
</tbody>
</table>

Two years after

One year after

Preventing attacks on a user's history through CSS :visited selectors

L. David Baron, Mozilla Corporation
Not all ideas are so lucky...

- **Browser-based identity management**
  - Password generators
  - Client certs
  - PAKE

- **Fine-grained sandbox architectures**
  - Plugin isolation
  - Origin isolation

- **Automatic clickjacking protection**
  - Wait, what?
NoScript has 90m downloads!

● Less than <0.1% of active internet users

● Dumping ground for chasm-challenged features

● Fundamentally different outlook than mainstream browsers
  ○ Extensive user interaction
  ○ Highly complex behavior
  ○ Breaks sites... by design!

flattr.com/profile/ma1
Are browser vendors too conservative?

- Features are not free!
  - Simplicity as a selling point
  - Rely on addons for niche functionality
- Breakage is **very** expensive
  - Web sites slow to adapt
  - Switching costs are low

*Browser Not Supported*

The browser you are using is not supported by this application. If you wish to use this application, please use one of the links below to download and install the current version of a supported browser:

- Microsoft Internet Explorer
- Netscape
- Firefox
What program committees care about

- Novel
  - Not substantially similar to previous work
  - Opens new avenues of research
  - Unconstrained by conventional thinking

- Non-trivial
  - Makes clever use of advanced tools and techniques
  - Substantial work involved in system implementation

These will get you a conference paper...

... but they **actively harm** a proposal's mainstream appeal
What browser vendors care about

● Must-have
  ○ Replaces broken, band-aid approaches that are nevertheless *already being widely used*
  ○ No browser wants to be the only one without it

● Easy
  ○ Deployable *unilaterally*, with little effort
  ○ Everyone can implement in the same way
  ○ Can determine if implementation is correct

● Low-risk
  ○ Doesn't break anything important, even in the long tail
  ○ Any change that's not opt-in is risky
Make your proposal a **must-have**

- Can always find **someone** who likes your idea...
  - Early adoption not a sure-fire sign of mainstream need

- Addons are a final resting place for many niche features
  - A vendor needs to be embarrassed **not** to have it
  - Browser vendors are like dominos

- Marketing
  - Compelling demos
  - Mainstream press
  - Large web sites who will champion it
Must-have #1: Same-origin policy

- Origin = protocol://host:port

- Full access to same origin
  - Full network access
  - Read/write DOM
  - Storage

- Limited interaction with other origins
  - Import of library resources (e.g. scripts)
  - Forms, hyperlinks

- Introduced by Netscape in 1996 in response to media reports of cross-origin scripting attacks
How postMessage became must-have

- Allows client-side messaging between origins
- Increasingly popular web sites like Facebook build mechanisms around hacks (fragment identifier messaging)
- Microsoft decided it was safe, implemented in IE8
- Firefox wanted HTML5 feature parity with IE
- Safari wanted HTML5 feature parity with Firefox/IE
- By the time we dropped this bomb, it was too late to stop it
How history privacy became must-have

WHAT THE INTERNET KNOWS ABOUT YOU

● Compelling demos
● Real-world attacks
● Lawmakers and media interested

Perfect ingredients for competition among browser vendors

● Only partial solution but easy and low-risk
Make your proposal **easy**

Strongly preferable:

- Deployable unilaterally: doesn't require cooperation among multiple vendors
- Web sites don't have to adopt right away
- Everyone can implement it exactly the same

Non-examples

- Taint tracking
- toStaticHTML
- DNSSEC
X-Frame-Options versus ClearClick
Strict Transport Security

Original ForceHTTPS involved

- Cookies
- User-configurable options
- Mixed content protection

Stripped down proposal to make it easier to implement
Make your proposal **low-risk**

- Does it break functionality?
- Does it slow things down?
- Does it interfere with getting stuff done?
- Are you making more people sad than happy?
De-risking a security proposal

Choose one:

1. Make the security opt-in
   ○ Huge evangelism cost
   ○ Yet another thing to forget to do

1. Create brand new functionality
   ○ Sidesteps legacy considerations
   ○ Adoption barrier?

2. **Very** thorough performance & compatibility evaluation
   ○ Often ~5x harder than the actual implementation
   ○ Some features just weren't meant to be!
Opt-in security

X-Secure-Me-Harder: yes!
- Extremely popular approach! X-Frame-Options, X-Content-Type-Options, Strict-Transport-Security, etc.
- Header bloat problem

How many opt-in features had an impact on the world?
- Trickling down from the PayPals and Twitters
- Long tail takes many years

Alternative policy delivery mechanisms
- Host-meta
- New HTML tags/attributes
- Content Security Policy
New platforms

chrome web store

[Diagram showing components of an browser with annotations for Attacker, Page DOM, Isolated World, Content Script, Extension Core, Native Binary, Tabs, Bookmarks, Process Creation, and File System.]
Web Sockets and the risks of unfinished standards

By: Stephen Shankland
DECEMBER 10, 2010 3:54 AM PST

Enthusiasm for a promising new standard called **Web Sockets** has quickly cooled in some quarters as a potential security problem led some browser makers to hastily postpone support.

The Web Sockets technology, which opens up a live communication link between a browser and a server, remains an important part of plans to make the Web a home for more dynamic, interactive sites. It could, for example, speed up Google Instant searching and multiplayer games. But **Mozilla** and **Opera** put their Web Socket plans on hold this week until the wrinkles are ironed out.

The reversal is only the latest difficulty, though. Web Sockets development already had become somewhat contentious as eager browser makers—Google in particular—began including support for a specification they knew wasn’t done. Overall, the Web Sockets history illustrates some pitfalls of the style and pace of Web standards development.
On-by-default security? Yikes.

- Things fail mysteriously, and more often than you'd think
- Failures are (usually) not attacks
- For every bug filed, how many users just give up or switch browsers?
How securing Gmail ruined my Korean class

- Get a website to host your SWF
  http://victim.com/attack.swf

- User logs in to victim.com

- Get user to visit
  http://attack.com/

- Embed the SWF and hijack the session
  <embed src="http://victim.com/attack.swf"/>
Another on-by-default fail: OCSP

- Validating certificate takes >1sec for 10% of HTTPS requests
- Adds to initial page load time dramatically when dependent scripts, images, etc. are on other hostnames
- Must-have, yet high-risk. Browsers don't enforce
- Defeating OCSP with the number 3
Compatibility Evaluation Failures

"I checked the Alexa top 100"

"I changed the plugin security policy and I played a YouTube video"

"I went to 10 websites and only 2 of them broke"
Better ideas

● **Deep crawl**
  ○ Get beyond login pages
  ○ *Execute JavaScript (Kudzu)*

● **Client-side measurement**
  ○ *Google Chrome User Metrics*
  ○ *Firefox Test Pilot*

● **Ad networks**
  ○ *Flash Player ads*
  ○ *Iframe ads*
What a real evaluation looks like

Content Sniffing Algorithm

- Searched the entire Google crawl index for common mime type mismatches; eliminated unused sniffing rules

- QA team visited the top 500 sites and tested extensively while logged in

- Google Chrome user metrics study found less than 0.004% compatibility impact
If you don't have the Google index...

Alexa top 100,000

<table>
<thead>
<tr>
<th>Requesting Server</th>
<th>Rendering Mode</th>
<th>Total</th>
<th>HTTP Error</th>
<th>Correct Type</th>
<th>Incorrect Type</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Well-formed</td>
<td>Mal-formed</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Well-formed</td>
<td>Mal-formed</td>
</tr>
<tr>
<td>Same-origin</td>
<td>Standards</td>
<td>180,445</td>
<td>1,497</td>
<td>178,017</td>
<td>506</td>
</tr>
<tr>
<td></td>
<td>Quirks</td>
<td>25,606</td>
<td>455</td>
<td>24,445</td>
<td>332</td>
</tr>
<tr>
<td>Cross-origin</td>
<td>Standards</td>
<td>47,943</td>
<td>347</td>
<td>47,345</td>
<td>104</td>
</tr>
<tr>
<td></td>
<td>Quirks</td>
<td>6,075</td>
<td>53</td>
<td>5,891</td>
<td>57</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>260,069</td>
<td>2,363</td>
<td>255,698</td>
<td>999</td>
</tr>
</tbody>
</table>

CSS references blocked by strict enforcement or minimal enforcement
If you don't have your own browser…

![Graph showing percent of incoming links covered versus number of entry points for different websites. The graph includes lines for Flixster, Wells Fargo, Gmail, Capital One, Grooveshark, Bank of America, Facebook, New York Times, and Last FM.](image)
Sometimes the answer is "no"

Stanford SafeHistory

- Only showed links as visited if you visited from the current site
- Perfect protection from attack, but at what cost?
Compatibility numbers aren't good?

An unlikely savior...
Frame navigation

```javascript
window.open("https://attacker.com/", "awglogin");
```
Mixed content

Security Information
This page contains both secure and nonsecure items.
Do you want to display the nonsecure items?
Yes  No  More Info
Origin contamination

Gmail is a new kind of webmail, built on the idea that email can be more intuitive, efficient, and useful. And maybe even fun. After all, Gmail has:

**Less spam**
Keep unwanted messages out of your inbox with Google's innovative technology

**Fast search**
Use Google search to find the exact message you want, no matter when it was sent or received.

iGoogle - Mozilla Firefox

iGoogle

Google Search  I'm Feeling Lucky
Taking one for the team
Dangers of chasm thinking

1. You are here
2. And you should be trying to enter here...
3. But you are overly concerned about being here
4. So, instead, you get nowhere.
Should researchers bother with nice-to-have, difficult, risky ideas?
Yes!

but...
Let your idea be hacked apart

- Don't expect the final solution to resemble the original form
  - Rebranded
  - User interface changed/removed
  - Unnecessary complexity dropped

- The best ideas are easily tweaked and repurposed

- Sometimes just a problem statement is a contribution

- Celebrate indirect impact!
Perspectives

- Firefox addon topped out at ~10,000 users
- Crossed the chasm in another form:
  ~100,000,000 Chrome users benefiting from HTTPS monitoring
MashupOS

- Never went anywhere in original form
- Key ideas survived
  - `postMessage(message, targetOrigin)`
  - `text/html-sandboxed` MIME type
- Gazelle may find a similar fate
What you can do right now to help

- Analyze existing new proposals in standardization
- Catch problems before legacy concerns creep in
- **WebRTC** - direct network communication between web clients
- **Component Model** - lets you construct your DOM out of mutually distrusting components with security boundaries between them
- **ECMAScript 6** - have untrusted JavaScript run in your page
- **Content Security Policy** - protect the developer from themselves
- **Web Intents** - allow one web application to invoke another

The time to get involved is now!
Show up!

- Meet the decision-makers
  - Many are in this room!
  - Many Mozilla meetings are open
- Join mailing lists
  - WHATWG
  - W3C public-web-security
  - IETF WebRTC
  - IETF Web Security Working Group
- Write code!
  - Firefox, Google Chrome, and most of Safari are open source
  - Nothing says "implement me now" like a patch ready for approval
Controversial things I just said

- NoScript is a niche browser... not the browser of the future
- Program committees actively harm good ideas
- OCSP is risky
- Taint tracking is hard
- SafeHistory is undeployable
- Breaking web sockets for 6 months was not a mistake
- You should crash Mozilla team meetings