Same origin policy

Elie Bursztein
Outline

• Browser evolution
• Same origin policy
• SOP limitation
• Origin Contamination
• Frame Navigation
RSS slams Pope on conversions - Rediff - 3 hours ago
We cannot comment, says RSS - Indian Express - 4 hours ago
We want every inch of PoK back: RSS - Times of India - 4 hours ago
Try Google News: Search news for rss or browse the latest headlines

RDF Site Summary (RSS) 1.0
RDF Site Summary (RSS) 1.0. Official Specification. The RSS 1.0 specification was released on 2000-12-06. It was published ...
www.purl.org/rss/1.0/ - 12k - Cached - Similar pages

XML.com: What is RSS? [Dec. 18, 2002]
In Mark Pilgrim’s inaugural Dive Into XML column, he reviews the history and technical details of the varieties of RSS on the Web. ... What is RSS? ...
www.xml.com/pub/a/2002/12/18/dive-into-xml.html - 37k - Cached - Similar pages

Latest RSS News (RSS Info)
The RSS format allows quick and easy syndication of news, headlines, and more. RSS specification, RSS Info, News and information on the RSS format.
News
NOVEMBER 20, 2005
Xbox 360 in the House
Sarel Tunies In and Drop Out
Continue reading "Xbox 360 in the House"

NOVEMBER 18, 2005
Legend's The Three Musketeers playable demo - available now!
The Three Musketeers have been kidnapped!
Continue reading "Legend's The Three Musketeers playable demo - available now!"

Reviews
NOVEMBER 14, 2005
Shattered Union
Can you fix it?
Continue reading "Shattered Union"

Previews
NOVEMBER 8, 2005
2K Sports Xbox 360 Preview
Next generation sports titles:
Cloth physics and sweat to highlight
Continue reading "2K Sports Xbox 360 Preview"

OCTOBER 31, 2005
The Matrix: Path of Neo
Which do you want? The red pill or the blue one?
Continue reading "The Matrix: Path of Neo"

Events
SEPTEMBER 16, 2005
World Poker Tour Party
A trip to AC, free food and drink, a good poker game, what more could you ask for?
Continue reading "World Poker Tour Party"
Google Chrome

From Wikipedia, the free encyclopedia

This article is about the web browser. For the operating system, see Google Chrome OS.

Google Chrome is a web browser developed by Google that uses the WebKit layout engine and application framework. It was first released as a beta version for Microsoft Windows on 2 September 2008, and the public stable release was on 11 December 2008. The name is derived from the graphical user interface frame, or "chrome", of web browsers. As of February 2010, Chrome was the third most widely used browser, with 5.61% of worldwide usage share of web browsers, according to Net Applications.[1]

In September 2008, Google released a large portion of Chrome's source code, including its V8 JavaScript engine, as an open source project entitled Chromium.[2][3] This move enabled third-party developers to study the underlying source code and help port the browser to Mac OS X and Linux. A Google spokesperson also expressed hope that other browsers would adopt V8 to help web applications.[4] The Google-authored portion of Chromium is released under the permissive BSD license,[5] which allows portions to be incorporated into both open source and proprietary software programs.[6] Other portions of the source code are subject to a variety of open-source licenses.[7] Chromium implements the same feature set as Chrome, but lacks automatic updates and Google Branding, most notably it has a blue-colored logo in place of the multicolored Google logo.[8]
The browser is the new OS?

<table>
<thead>
<tr>
<th>Feature</th>
<th>Gears</th>
<th>BrowserPlus</th>
<th>Firefox</th>
<th>IE8</th>
<th>Webkit</th>
</tr>
</thead>
<tbody>
<tr>
<td>ApplicationCache</td>
<td>soon</td>
<td>?</td>
<td>✅</td>
<td>✅</td>
<td>✅</td>
</tr>
<tr>
<td>detect onLine</td>
<td>✅</td>
<td>?</td>
<td>✅</td>
<td>✅</td>
<td>✅</td>
</tr>
<tr>
<td>LocalServer</td>
<td>✅</td>
<td>?</td>
<td>✅</td>
<td>✏️</td>
<td>✅</td>
</tr>
<tr>
<td>Storage</td>
<td>✅</td>
<td>?</td>
<td>✅</td>
<td>✅</td>
<td>✅</td>
</tr>
<tr>
<td>Database</td>
<td>✅</td>
<td>?</td>
<td>✅</td>
<td>✏️</td>
<td>✅</td>
</tr>
<tr>
<td>Threading</td>
<td>✅</td>
<td>?</td>
<td>✅</td>
<td>✏️</td>
<td>✅</td>
</tr>
<tr>
<td>SQL</td>
<td>✅</td>
<td>?</td>
<td>✅</td>
<td>✏️</td>
<td>✅</td>
</tr>
</tbody>
</table>
Running Remote Code is Risky

• Integrity
  – Compromise your machine
  – Install malware rootkit
  – Transact on your accounts

• Confidentiality
  – Read your information
  – Steal passwords
  – Read your email
Browser vulnerability trend

- Safari
- IE
- Firefox
Browser Sandbox

• Goal
  – Run remote web applications safely
  – Limited access to OS, network, and browser data

• Approach
  – Isolate sites in different security contexts
  – Browser manages resources, like an OS
Chrome architecture
Chrome architecture

**Rendering Engine**
- HTML Parsing
- CSS parsing
- Image decoding
- Javascript Interpreter
- Regular Expression
- DOM
- Rendering
- SVG
- XML parsing
- XSLT

**Browser kernel**
- Cookie database
- History database
- Password database
- Window management
- Safe Browsing blacklist
- Network stack
- SSL/TLS
- Disk cache
- Download manager
- Clipboard

A. Barth, C. Jackson, C. Reis, Google Chrome team
Dealing with multiple contents

• Windows and tabs
• Frame and Iframe
• Embedding content directly
Popup windows

• With hyperlinks
  
  <a href="http://www.b.com" target="foo">click here</a>

• With JavaScript
  
  mywin = window.open("http://www.b.com", "foo", "width=10,height=10")
  
  – Navigating named window re-uses existing one
  – Can access properties of remote window:
    
    mywin.document.body
    mywin.location = "http://www.c.com";
Frames

• Modularity
  – Brings together content from multiple sources
  – Client-side aggregation

• Delegation
  – Frame can draw only on its own rectangle
Windows interaction
A good interaction?
Same-Origin Policy

How does the browser isolate different sites?
Policy Goals

• Safe to visit an evil web site

• Safe to visit two pages at the same time
  – Address bar distinguishes them

• Allow safe delegation
Announcements

1. New Lab virtual machine
   - Slimer (3.5 go)
   - Faster
   - Apt-get update, more editor
   - More exercises 😊

• Confirmed invited lectures
  - Sid from Mozilla
  - Jeremiah from Whitehat
Project planning

• Project 1 update
  – SQL structure is now left to your choice
• Project 2 : Building an exploitation framework
• Project 3 : Build a bookmarklet system
• Project 4 : research project
How to separate web site content?
Origin

https://login.yahoo.com/config/login

Scheme
Host
(Port)
Same Origin Policy

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

<table>
<thead>
<tr>
<th>Origin</th>
<th>Read</th>
<th>Write</th>
</tr>
</thead>
<tbody>
<tr>
<td>Same</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Different</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

It is more tricky than it look like!
Password auto filling?
Password auto filling?

<form action=https://www.facebook.com/xxx>
<form action=http://www.facebook.com/xxx>
Password filling?

- `a.com/y/login.html` vs `b.com/x/login.html`
- `action="a.com/login"` vs `action="b.com/logme"`
Cross origin

Interactions between origins
Library import

```
<script
src=https://seal.verisign.com/getseal?host_name=a.com>
</script>
```

- Script has privileges of imported page, NOT source server.
- Can script other pages in this origin, load more scripts
- Other forms of importing
HTML Form

• Many ways to send information to other origins

  <form action="http://www.bank.com/">
    <input name="data" type="hidden" value="hello">
  </form>

  <img src="http://www.b.com/?data=hello"/>

• No user involvement required
• Cannot read back response
window.postMessage

• New API for inter-frame communication
  – Supported in latest betas of many browsers
  – A network-like channel between frames

![Add a contact](facebook.png)

![Share contacts](contacts.png)
postMessage syntax

frames[0].postMessage("Attack at dawn!",
                        "http://b.com/");

window.addEventListener("message", function (e) {
    if (e.origin == "http://a.com") {
        ... e.data ... }
}, false);
XHR request

• Introduced in IE 5

```javascript
var xhr = new XMLHttpRequest;
xhr.open("GET", "http://dev.jquery.com/~john/xdomain/test.php", true);
xhr.onreadystatechange = function(){
    if (xhr.readyState == 4) {
        if (xhr.status == 200) {
            document.body.innerHTML = "My Name is: " + xhr.responseText;
        } else {
            document.body.innerHTML = "ERROR";
        }
    }
}
xhr.send(null);
```
XHR and same origin policy

- Post data
- Read data
- Read status

How to load dynamic data from another web site?
Tricks

- `<script src="xxx?req=bla">`
- Message passing via API
- XHTML request response code
Cross domain post

- Lift-off same origin policy for XHR requests
- Require both side cooperation
- Available since FF 3.5
- IE8 have almost the same with a different API
Access-Control-Allow-Origin: http://A.com
Cross domain headers

- Access-Control-Allow-Origin
- Access-Control-Allow-Methods
- Access-Control-Allow-Headers
- Access-Control-Allow-Credentials

https://developer.mozilla.org/En/HTTP_Access_Control
Example

Access-Control-Allow-Origin: http://foo.example
Access-Control-Allow-Methods: POST, GET, OPTIONS
Access-Control-Allow-Headers: X-PINGOTHER
Access-Control-Max-Age: 1728000
More way to bypass the same origin

• Timing attack
• Extension
• Flash (comeback Thursday 😊)

![Bar chart showing examples of valid and invalid data]

- **Example 1**: Valid: 50, Invalid: 30
- **Example 2**: Valid: 150, Invalid: 120
Origin contamination

There is no turning back
<iframe src="b.html"></iframe>
Secure Content Policy

• Planned for FF 3.7
• Control the contamination by enforcing content restriction
• User HTTP headers so it can’t be manipulated

• Come to Sid lecture to know more
Navigation

Who decides what content goes in a frame?
A Guninski Attack

```javascript
window.open("https://attacker.com/", "awglogin");
```
What should the policy be?
# Legacy Browser Behavior

<table>
<thead>
<tr>
<th>Browser</th>
<th>Policy</th>
</tr>
</thead>
<tbody>
<tr>
<td>IE 6 (default)</td>
<td>Permissive</td>
</tr>
<tr>
<td>IE 6 (option)</td>
<td>Child</td>
</tr>
<tr>
<td>IE7 (no Flash)</td>
<td>Descendant</td>
</tr>
<tr>
<td>IE7 (with Flash)</td>
<td>Permissive</td>
</tr>
<tr>
<td>Firefox 2</td>
<td>Window</td>
</tr>
<tr>
<td>Safari 3</td>
<td>Permissive</td>
</tr>
<tr>
<td>Opera 9</td>
<td>Window</td>
</tr>
<tr>
<td>HTML 5</td>
<td>Child</td>
</tr>
</tbody>
</table>
Window Policy Anomaly
## Adoption of Descendant Policy

<table>
<thead>
<tr>
<th>Browser</th>
<th>Policy</th>
</tr>
</thead>
<tbody>
<tr>
<td>IE7 (no Flash)</td>
<td>Descendant</td>
</tr>
<tr>
<td>IE7 (with Flash)</td>
<td>Descendant</td>
</tr>
<tr>
<td>Firefox 3</td>
<td>Descendant</td>
</tr>
<tr>
<td>Safari 3</td>
<td>Descendant</td>
</tr>
<tr>
<td>Opera 9</td>
<td>(many policies)</td>
</tr>
<tr>
<td>HTML 5</td>
<td>Descendant</td>
</tr>
</tbody>
</table>
Why include “targetOrigin”?

• What goes wrong?
  
  ```javascript
  frames[0].postMessage("Attack at dawn!");
  ```

• Messages sent to *frames*, not principals
  – When would this happen?
Conclusion

• Same origin policy is flexible
  – Address bar reflects the principal that's in control
  – Content may be affected by other principals

• Delegation
  – Library import
  – Domain relaxation
  – Pixel delegation via frames

• Communication
  – Data export
  – Opt-in messaging