Feasibility and Real-world Implications of Web Browser History Detection

Artur Janc, Łukasz Olejnik

What the Internet Knows About You

W2SP 2010

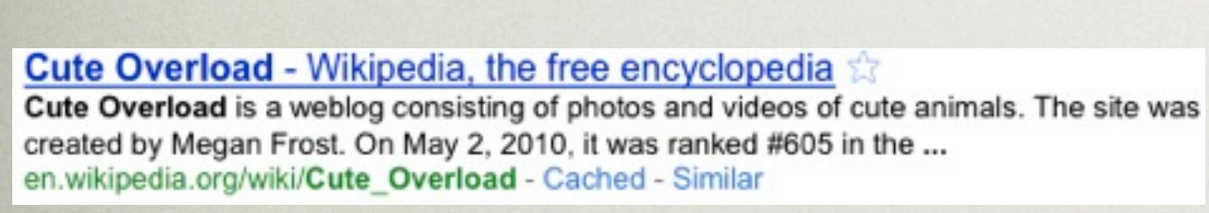
Outline

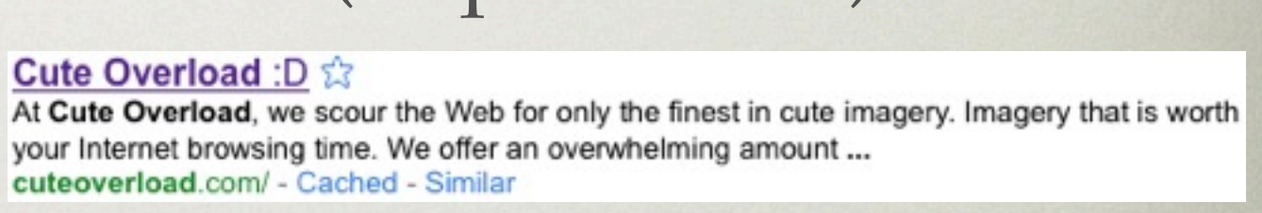
Attacks on privacy using CSS :visited to inspect users' Web browsing histories

1. Basics (quick) and history
2. Analysis
    - What can be detected, performance
    - Building a history detection system
3. Results
4. Current work / Countermeasures

How it Works

- CSS :visited, :link styling
  - Browsers apply additional styles to links which the user had visited (requirement)





- Attack:
  - Insert a link with a URL to check for
  - Check if visited style was applied (JS) or if a visited “marker” resource was downloaded

Examples

CSS

```
<style>
#foo:visited {background: url(/?yes-foo);}
#bar:link {background: url(/?no-bar);}
</style>
<a id="foo" href="http://foo.org"></a>
<a id="bar" href="http://bar.biz"></a>
```

JavaScript

```
<script>
var r1 = 'a {color:green;}';
var r2 = 'a:visited {color:red;}';

document.styleSheets[0].insertRule(r1, 0);
document.styleSheets[0].insertRule(r2, 1);

var a_el = document.createElement('a');
a_el.href = "http://foo.org";

var a_style = document.defaultView.\
  getComputedStyle(a_el, "");

if (a_style.getPropertyValue("color")
    == 'red') { // link was visited }
</script>
```

A known Mozilla “bug” since at least 2000

History (of) Detection

- Mozilla bugs #57351 (2000), #147777 (2002)
- Issue described by:
  - (Felten & Schneider), Ruderman, Jakobsson & Stamm., Jackson et al., others
  - Several analyses of Web security issues (including Google's BSH)
- Rediscovered on multiple occasions (PoCs)
- Life always goes on

What Changed Since Then

- Browsers still support :visited selectors
- The Web has changed
- More apps are Web-based
- More personal interactions with the Web (social networks/news, forums)
- Browsers are much faster

What Can Be Detected?

- Protocols
- Framed content
- HTTP status codes

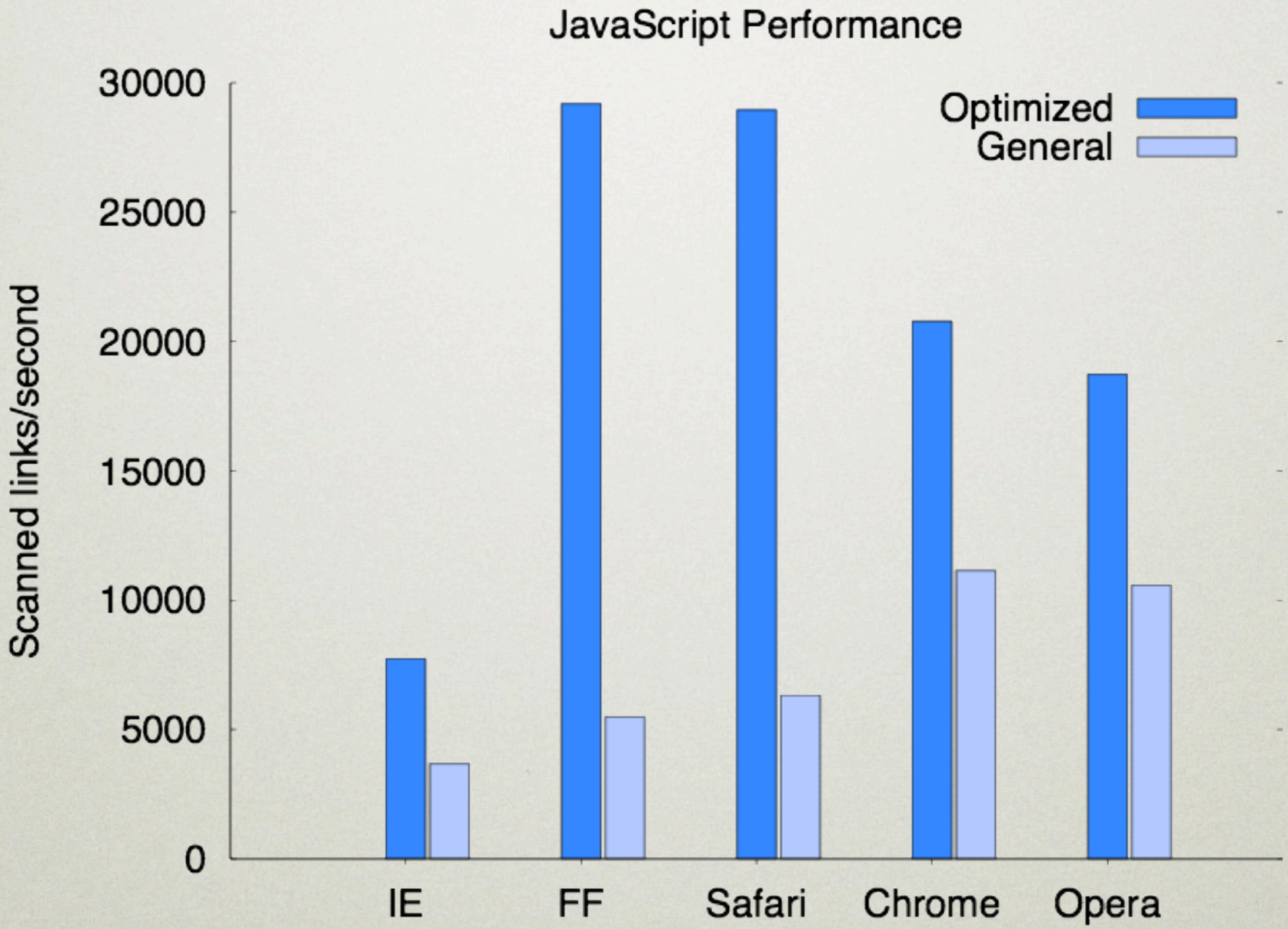
	IE	Firefox	Safari	Chrome	Opera
http	✓	✓	✓	✓	✓
https	✓	✓	✓	✓	✓
ftp	✓	✓	✓	✓	✓
file	✓	✓	✓		✓
frames		✓		✓	
iframes		✓		✓	
200	✓	✓	✓	✓	✓
30x	n/a	both	original	both	both
meta redir	n/a	✓	✓	✓	✓
4xx		✓	✓	✓	✓
5xx		✓	✓	✓	✓

- Usually: if in address bar ⇔ detectable
- Can detect parameters from forms submitted with HTTP GET (not POST)
- Affected by history expiration policies

How Long Does it Take?

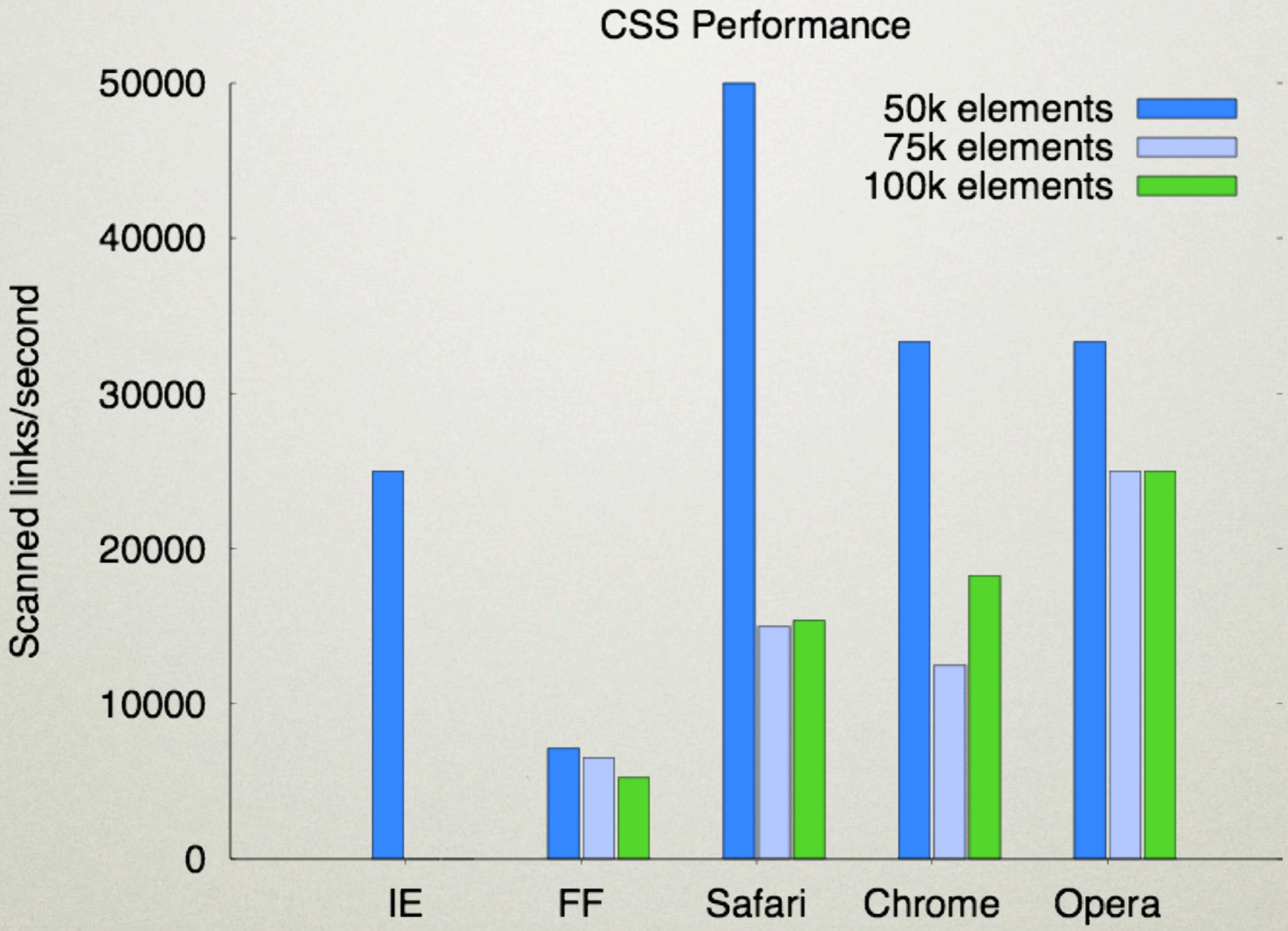
- Modern browsers are **fast**
- Can do a few smart things to improve performance & avoid resource limits
- Can optimize JS detection code for each browser (can be significantly faster)
- Fallback CSS-only technique still good

How Long Does it Take?



- JavaScript: ~ 20,000 links / second

How Long Does it Take?



- CSS: up to 25,000 links/sec (small sets)

Detection System

- Demonstrate browser history detection
  - Thousands of websites, categorized
    - Detect *secondary* resources (subpages) and other information (usernames, etc)
- Educate users, describe issue
- Gather real world data (analyze impact)

This page checks your browser history and determines which of the 5000 most popular Internet websites you've recently visited.

100%

Done

Popular websites you've recently visited (8)

Google
(4 visited pages detected)

Quantcast top #1 site

MSN
(5 visited pages detected)

Quantcast top #3 site

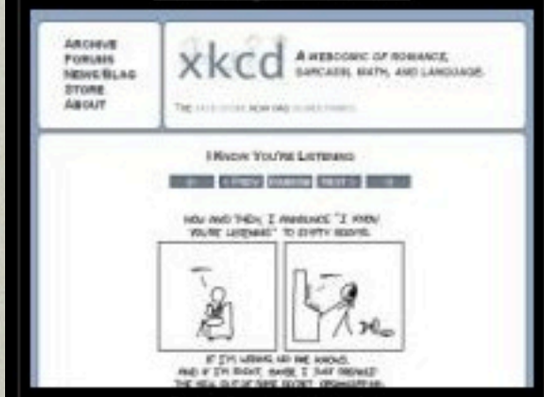
ieee.org
(1 visited page detected)

Quantcast top #2425 site

cuteoverload.com
(1 visited page detected)

Quantcast top #4789 site

xkcd
(3 visited pages detected)

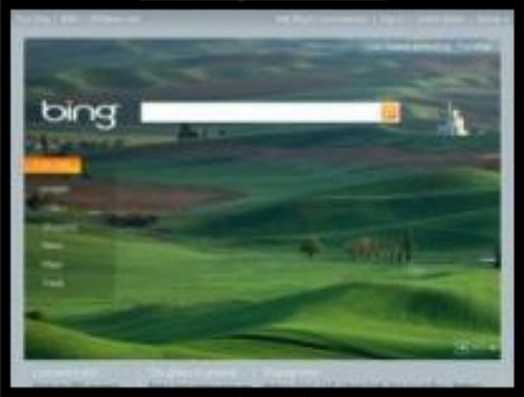


Popular webcomic

Reddit
(1 visited page detected)

Reddit social news

Bing
(1 visited page detected)



Search engine

Yahoo!
(1 visited page detected)

Tech site in the category: Web

List of detected Web pages

Google	*Quantcast top #1 site*
http://www.google.com/intl/en/ads/	
http://www.google.com/services/	
http://www.google.com/intl/en/about.html	
http://www.google.com/	
MSN	*Quantcast top #3 site*
http://moneycentral.msn.com/home.asp	
http://entertainment.msn.com/	
http://g.msn.com/0TO_/enus	
http://msn.com/	
http://lifestyle.msn.com/	
ieee.org	*Quantcast top #2425 site*
http://ieee.org	
cuteoverload.com	*Quantcast top #4789 site*
http://cuteoverload.com/	
xkcd	*Popular webcomic*
http://xkcd.com/archive/	
http://blag.xkcd.com/	
http://xkcd.com/	
Reddit	*Reddit social news*
http://www.reddit.com/	
Bing	*Search engine*
http://www.bing.com	
Yahoo!	*Tech site in the category: Web*
http://www.yahoo.com/	

How it Works

- For each test send *primary* links to user
  - http://msn.com, http://msn.com/home.asp
- For each found link check ~100 popular *secondary* links (subpages & resources)
  - Crawling, search engine API, manual
- For certain sites, enumerate resources
  - Usernames, search terms, zipcodes

Test Categories

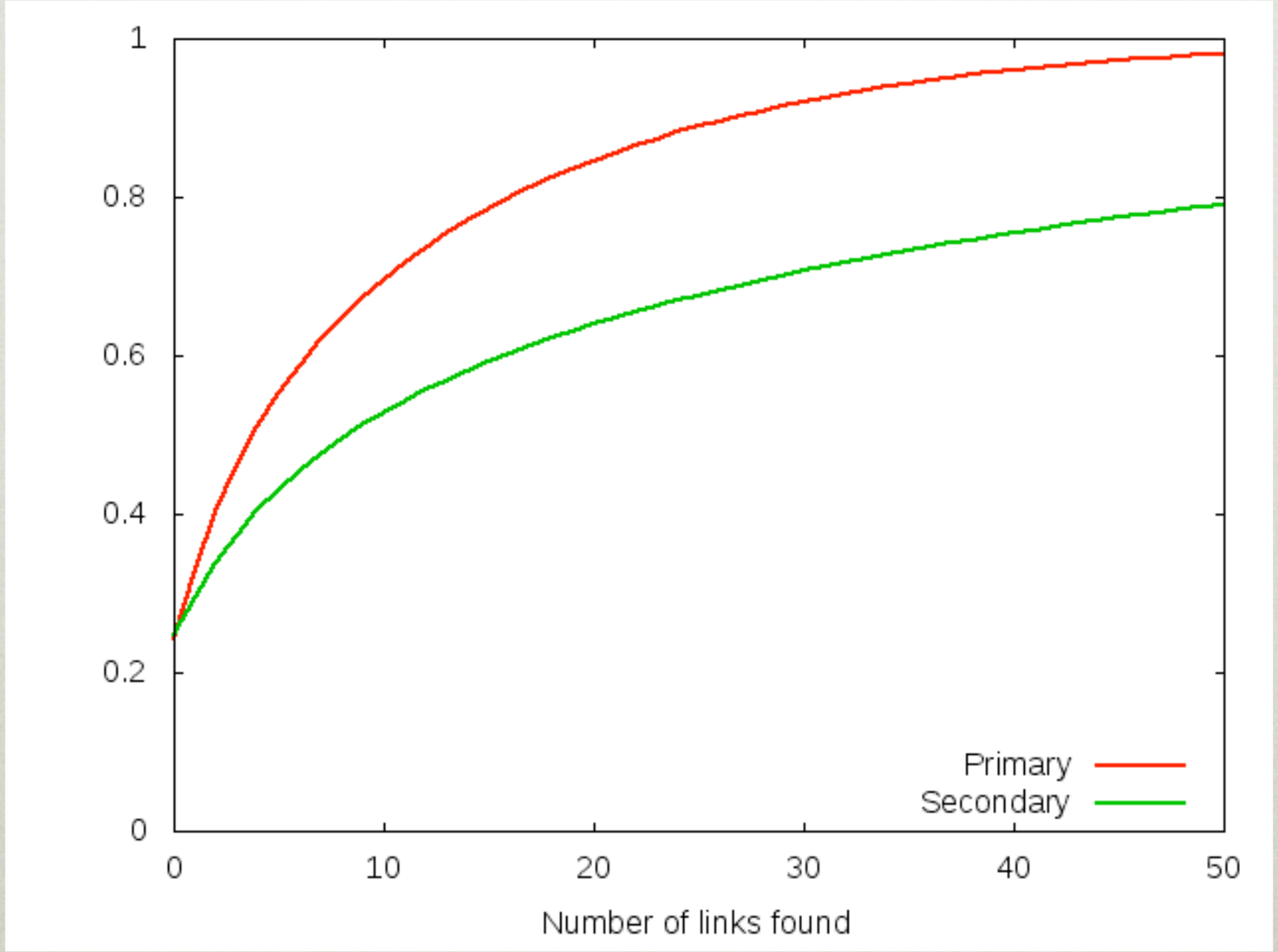
- Popular websites (Alexa, Quantcast, ...)
- Categorized sites
  - Online stores, .gov / .mil sites, banks, dating sites, universities, adult
- Social news sites: Slashdot, Digg, Reddit
- Sensitive sites (also zipcodes, search terms)
- 21 tests, 72k primary URLs, 8.6M secondary

General Results

- Gathered between 09/2009 and 02/2010
- 271,576 users, 703,895 tests executed

	Users		Found pri		#pri (med)		#sec (med)	
	JS	CSS	JS	CSS	JS	CSS	JS	CSS
top5k	206,437	8,165	76.1%	76.9%	12.7 (8)	9.8 (5)	49.9 (17)	34.6 (9)
top20k	31,151	1,263	75.4%	87.3%	13.6 (7)	15.1 (8)	48.1 (15)	51.0 (13)
all	32,158	1,325	69.7%	80.6%	15.3 (7)	20.0	49.1 (14)	61.2

Top5k Distribution



90th percentile: ~30 primary, ~120 secondary

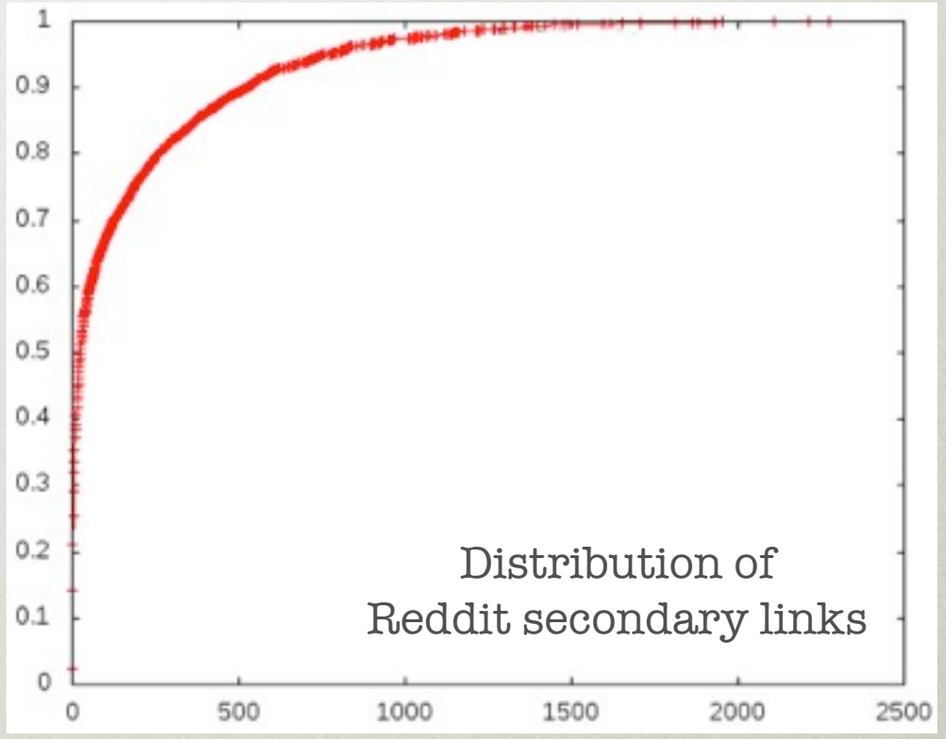
Browser Differences

	IE		Firefox		Safari		Chrome		Opera	
	JS	CSS	JS	CSS	JS	CSS	JS	CSS	JS	CSS
top5k	73	92	75	77	83	79	93	100	70	82
top20k	81	95	69	86	89	97	90	100	88	95
all	78	97	62	79	85	89	87	98	85	83

Social News

- Links from RSS feeds of popular social news sites and 32 regular news services

	Median secondary	Average secondary
All news	7	45.0
Slashdot	3	15.2
Digg	7	51.8
Reddit	26	163.3



- Monitored for visited profile pages to detect usernames (Reddit: 2.4%)

Some Random Results

- Found some zipcodes (9.8%) and search engine queries (~0.2%)
- Can identify Wikileaks power users

Fixing It

- All browsers susceptible
- A server-side fix won't help (impractical)
- Hard to get adoption for a plug-in (has been tried with SafeHistory)
- Hard to change browser behavior to close the hole (standards; developers get angry)
- But...

Coming Soon

- David Baron's/Mozilla Corp.'s proposal
  - Apply only **-color* rules to visited styles
    - Make JS functions lie about actual style
- Should be in Firefox 4.0 (~November)
- Similar changes rumored for WebKit
- Not ideal, but a big step forward; now we **must** get other browsers to do the same

Thank you