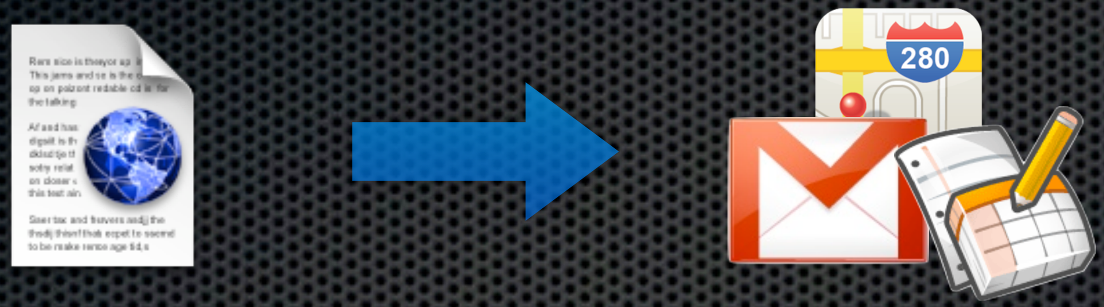
Web Browsers as Operating Systems:

Supporting Robust and Secure Web Programs

Charles Reis

Final Exam - May 27, 2009

Web is Evolving



Pages

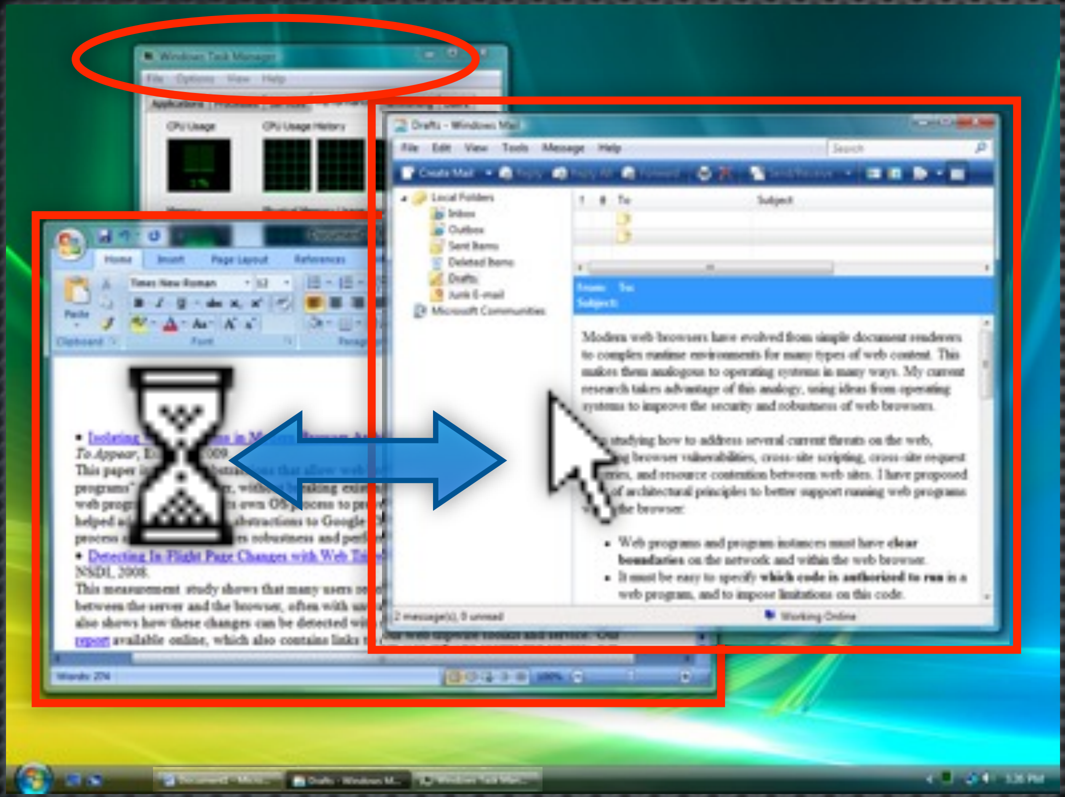
Programs

- **More complex, active content**
- **Browser now in role of OS, but faces challenges**
    - Browsers aren’t built for programs
    - Web content not designed to express programs

Concrete Problems

Problems	*Contributions*
Program Interference	Multi-Process Browsers [EuroSys '09]
In-Flight Page Changes	Web Tripwires [NSDI '08]
XSS	Script Whitelists
Browser Exploits	BrowserShield [OSDI '06]

Consider OS Landscape



Modern web browsers have evolved from simple document renderers to complex runtime environments for many types of web content. This makes them analogous to operating systems in many ways. My current research takes advantage of this analogy, using ideas from operating systems to improve the security and robustness of web browsers.

studying how to address several current threats on the web, ng browser vulnerabilities, cross-site scripting, cross-site request ries, and resource contention between web sites. I have proposed of architectural principles to better support running web programs the browser.

- Web programs and program instances must have **clear boundaries** on the network and within the web browser.
- It must be easy to specify **which code is authorized to run** in a web program, and to impose limitations on this code.

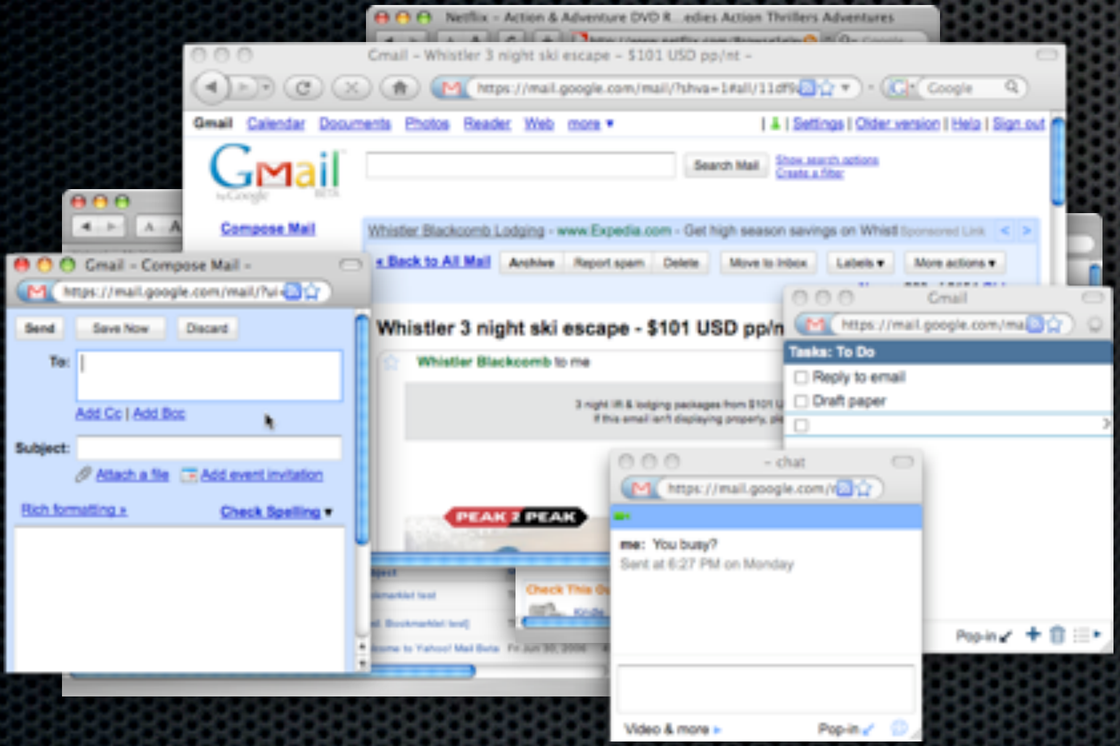
- Detecting In-Flight Page Changes with Web Tri

NSDI, 2008.

This measurement study shows that many users re between the server and the browser, often with also shows how these changes can be detected wi report available online, which also contains links t

- Performance isolation
- Resource accounting
- Failure isolation
- **Clear program abstraction**

Browsers Fall Short



Whistler Blackcomb Lodging - www.Expedia.com - Get high season savings on Whistl Sponsored Link

Whistler 3 night ski escape - $101 USD pp/n

Whistler Blackcomb to me

3 night lift & lodging packages from $101 U
If this email isn't displaying properly, pl

□ Reply to email

□ Draft paper

me: You busy?

Sent at 6:27 PM on Monday

- Unresponsiveness
- Jumbled accounting
- Browser crashes
- **Unclear what a program is!**

Preserve Web’s Strengths

- **Improve program support, but keep it:**
  - Easy to publish content
  - Easy to compose content
  - Generally safe to explore

Thesis:

Adapt lessons from the OS
to improve robustness and security
of web browsers and web content

- **Support four architectural principles:**

1. Identify program boundaries
2. Isolate programs from each other
3. Authorize program code
4. Enforce policies on program behavior

[HotNets '07]

Outline

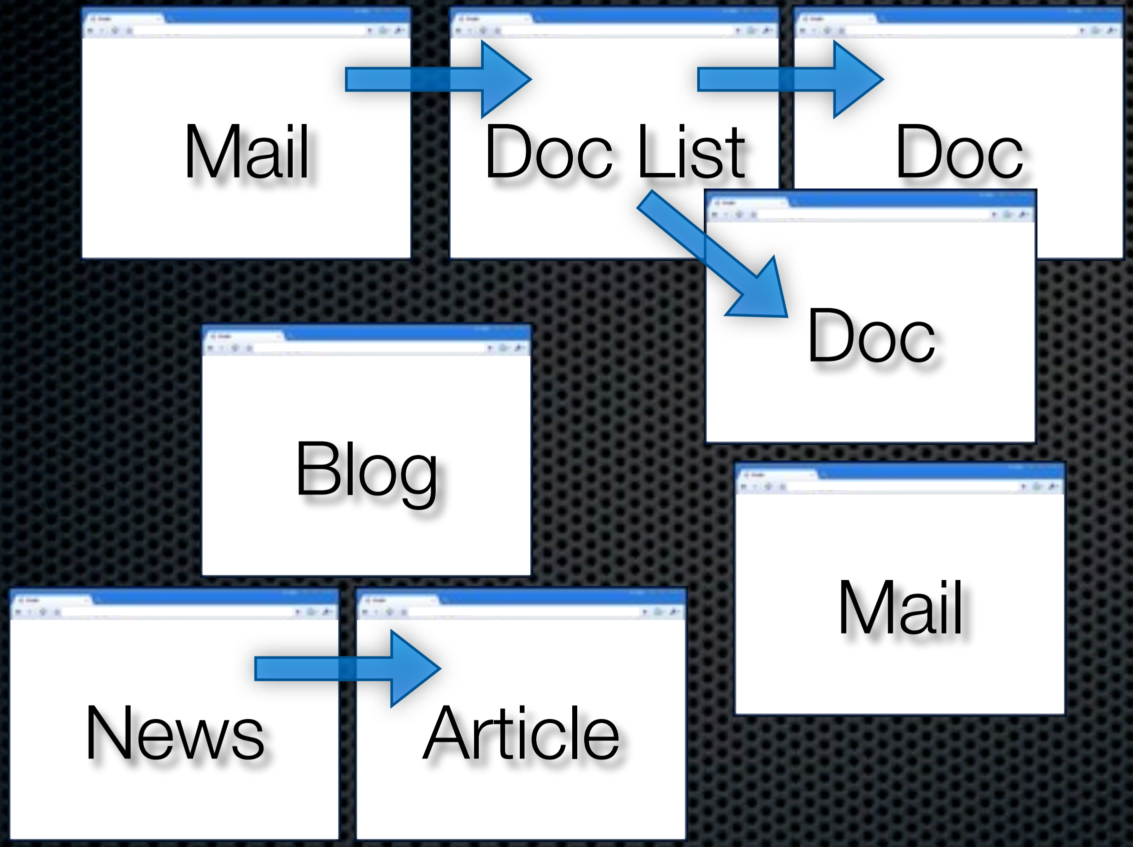
- **Browser Architecture: Chromium**
  - Identify program boundaries
  - Isolate programs from each other

Web Tripwires

Additional Contributions

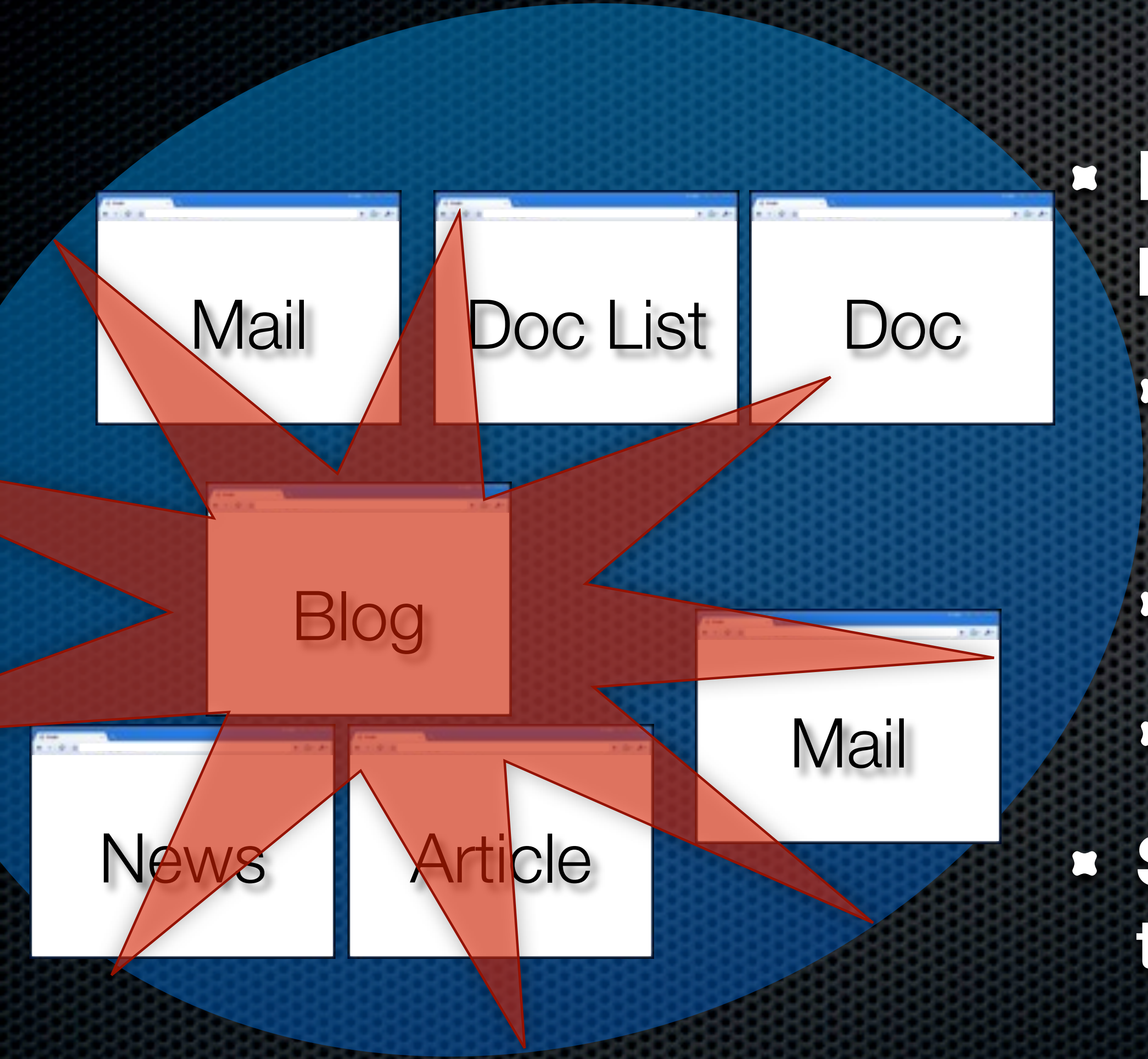
Future Directions

Programs in the Browser



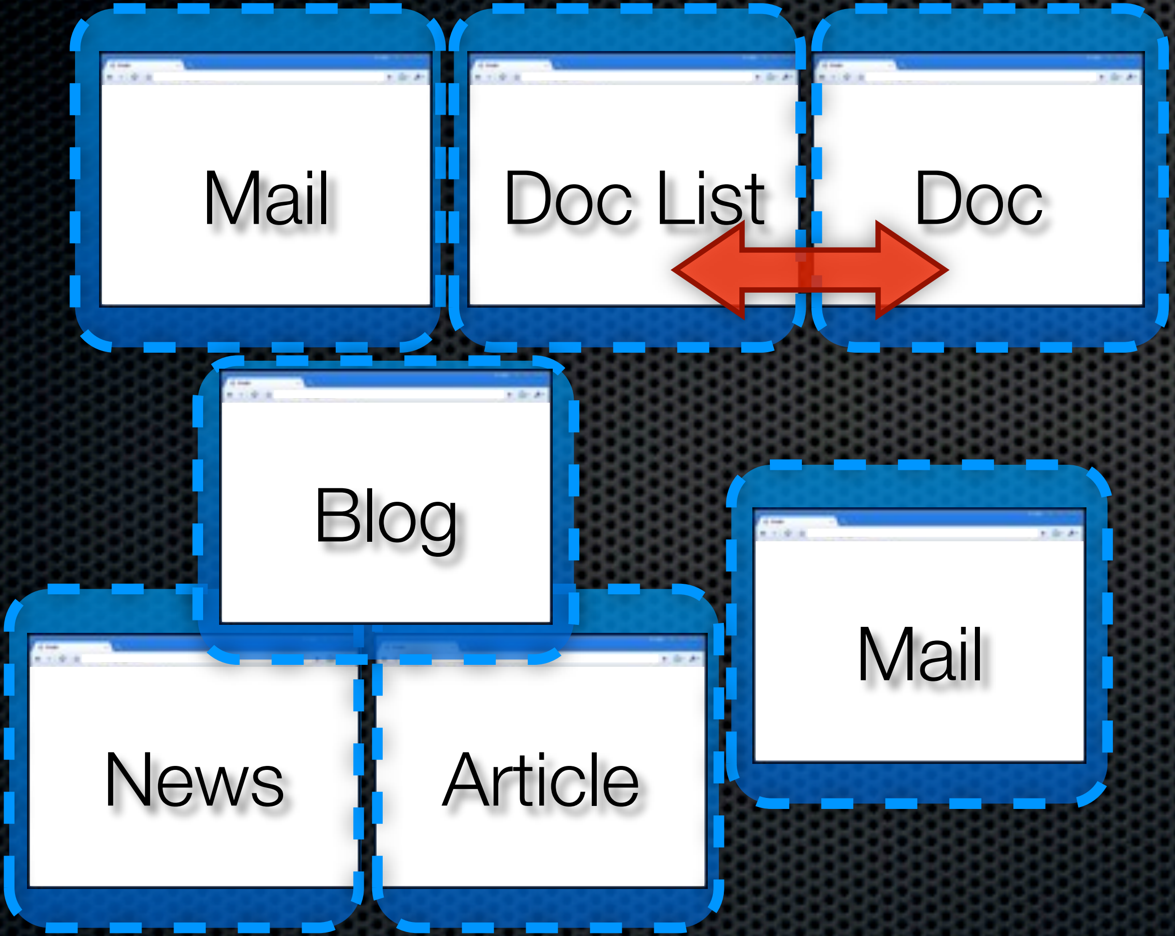
- Consider an example browsing session
  - Several independent programs

Monolithic Browsers



- **Most browsers put all pages in one process**
  - Poor performance isolation
  - Poor failure isolation
  - Poor security
- **Should re-architect the browser**

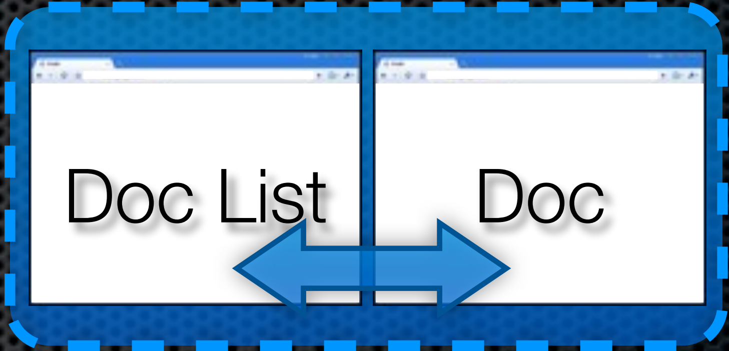
Process per Window?



- **Breaks pages** that directly communicate
  - Shared access to data structures, etc.
- **Fails as a program abstraction**

Need a Program Abstraction

- Aim for **new groupings** that:
  - **Match our intuitions**
  - **Preserve compatibility**
- Take cues from browser's existing rules
- Isolate each grouping in an OS process
- Will get **performance and failure isolation**, but not security between sites



Outline

- Browser Architecture
- **Program Abstractions**
- Program Isolation
- Evaluation

Ideal Abstractions

- **Web Program**
  - Set of pages and sub-resources providing a service
- **Web Program Instance**
  - Live copy of a web program in the browser
  - Will be isolated in the browser's architecture

Intuitive, but how to define concretely?

Compatible Abstractions

- Three ways to group pages into processes:

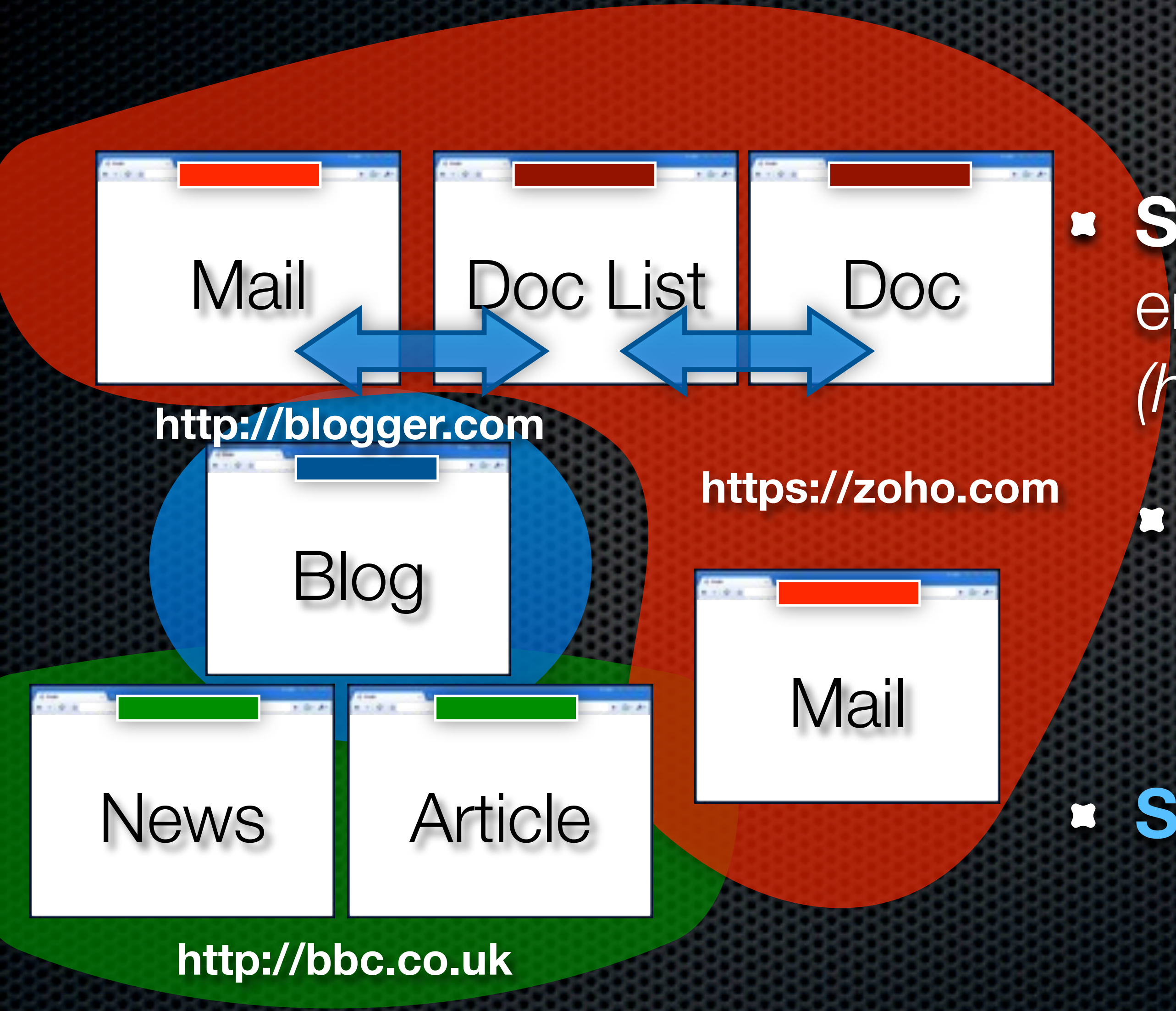
1. **Site:** based on *access control policies*

2. **Browsing Instance:** *communication channels* between pages

3. **Site Instance:** intersection of first two

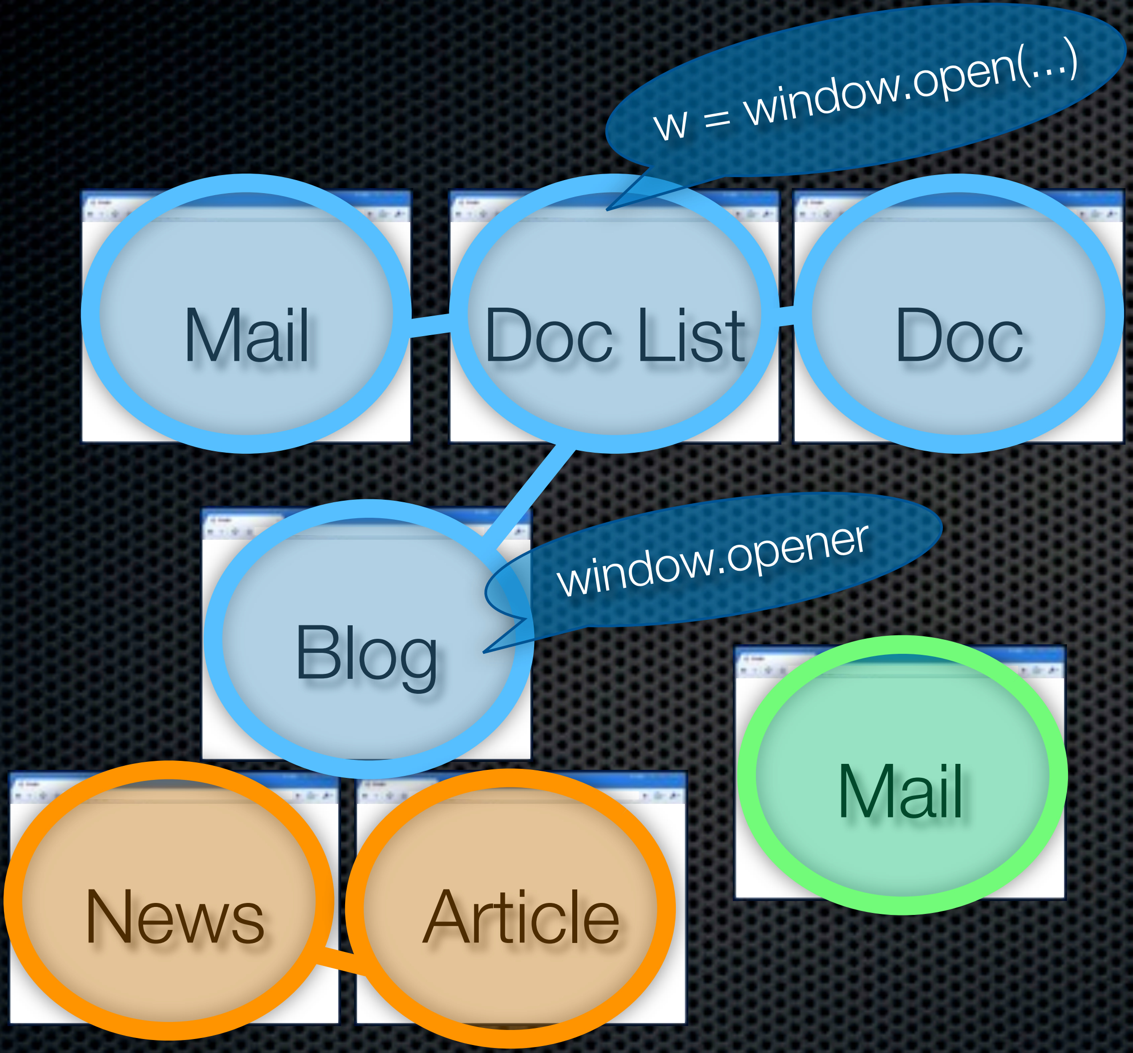
[EuroSys '09]

1. Sites



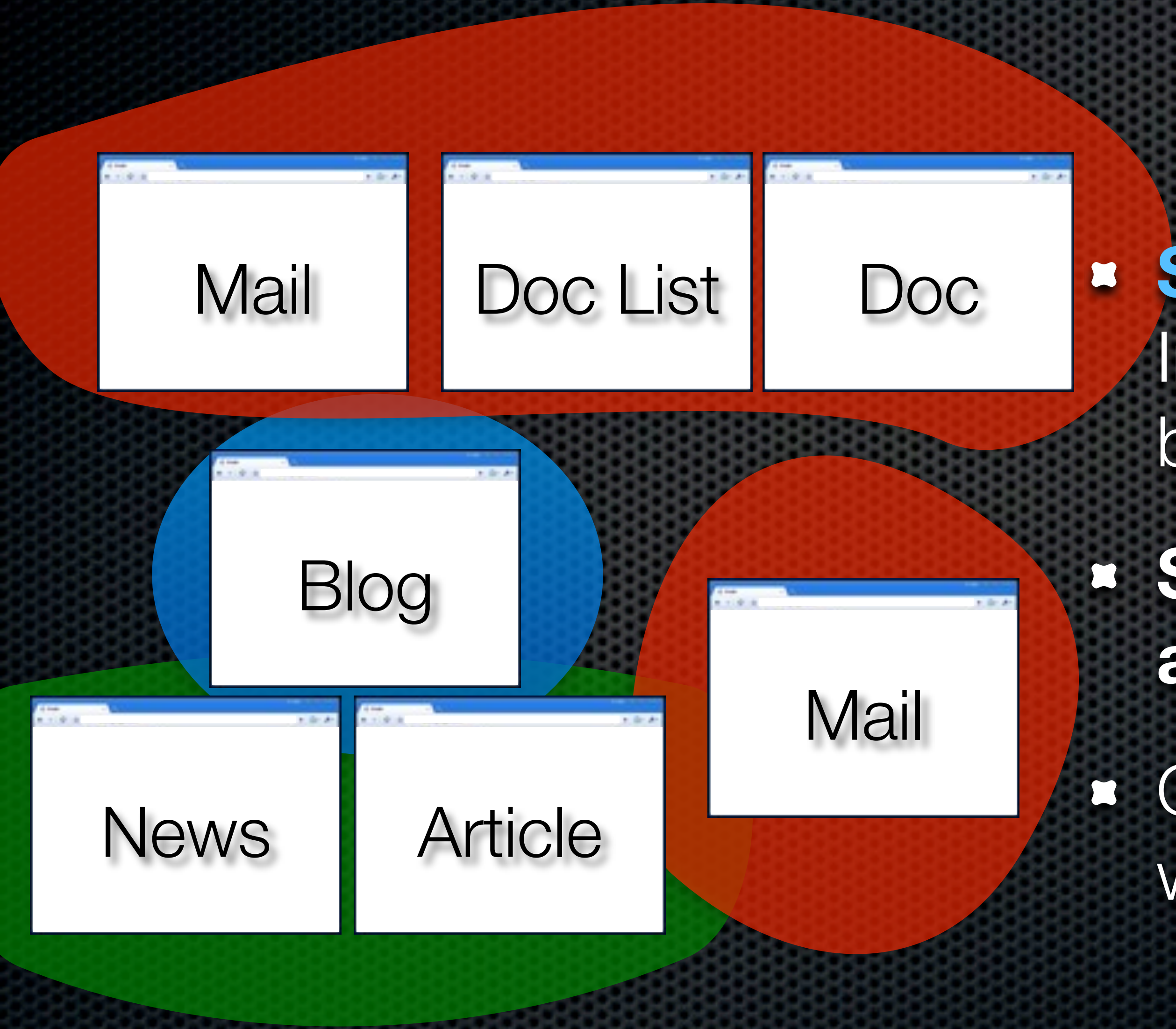
- **Same Origin Policy** enforces isolation *(host+protocol+port)*
  - Actual limit is *Registry-controlled domain name*
- **Site:** RCDN + protocol

2. Browsing Instances



- Which pages can talk?
- References between "related" windows
  - Parents and children
  - Lifetime of window
- **Browsing Instance:** connected windows, regardless of site

3. Site Instances



- **Site Instance:** Intersection of site & browsing instance
- **Safe to isolate from any other pages**
- Compatible notion of a web program instance

Outline

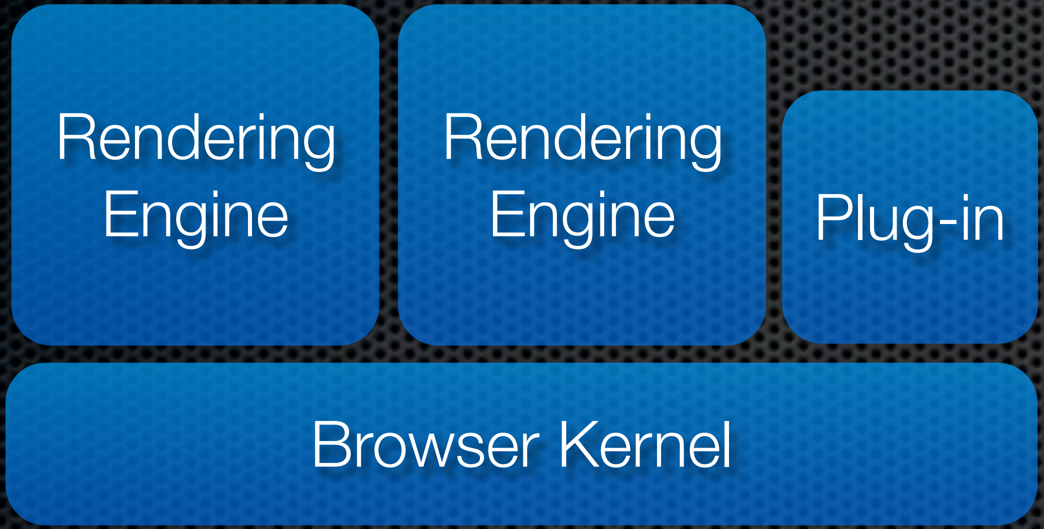
Browser Architecture

Program Abstractions

Program Isolation

Evaluation

Multi-Process Browser



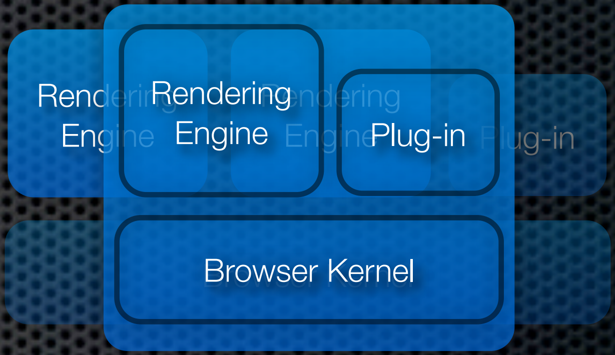
- **Browser Kernel**
  - Storage, network, UI
- **Rendering Engines**
  - Web program and runtime environment
- **Plug-ins**

Modules in Separate OS Processes

Implementations

- **Konqueror Prototype** (2006)
  - Proof of concept on Linux
- **Chromium** (Google Chrome, 2008)
  - Added support for Site Instance isolation

Chromium Process Models



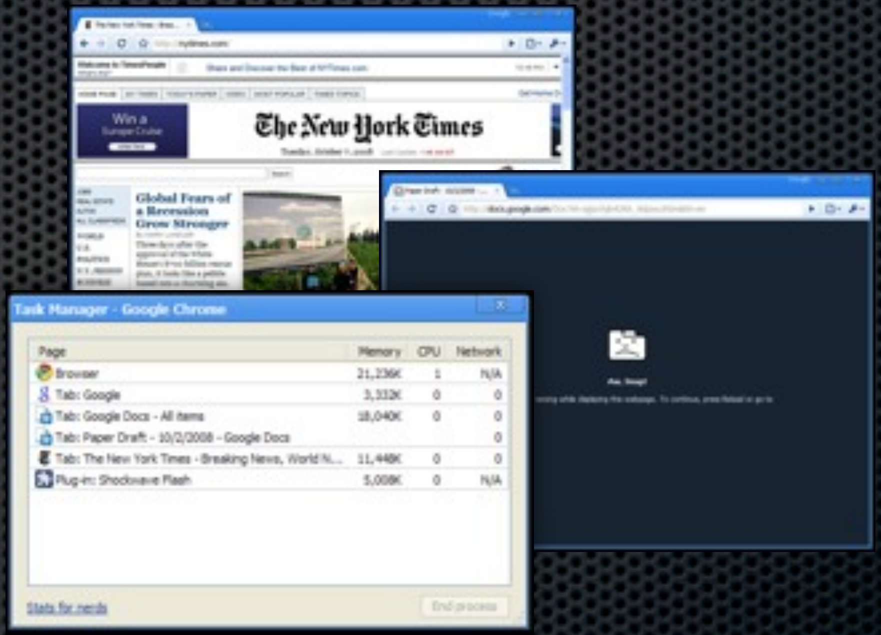
1. **Monolithic**
2. **Process-per-Browsing-Instance**
   - New window = new renderer process
3. **Process-per-Site-Instance** *(default)*
   - Create renderer process when navigating cross-site
4. **Process-per-Site**
   - Combine instances: fewer processes, less isolation

Outline

- Browser Architecture
- Program Abstractions
- Program Isolation
- **Evaluation**

Robustness Benefits

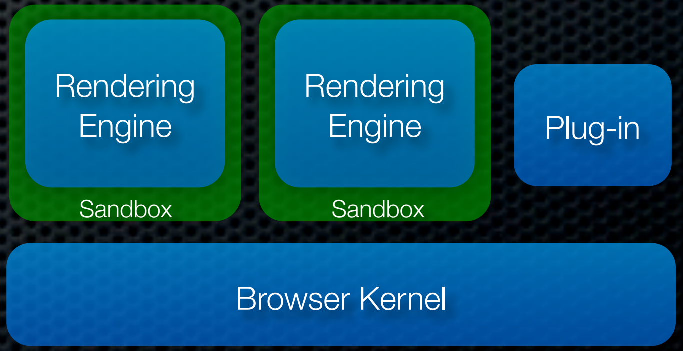
- Failure Isolation
- Accountability
- Memory Management



Task Manager - Google Chrome

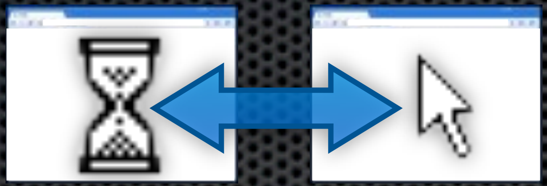
Page	Memory	CPU	Network
Browser	21,236K	1	N/A
Tab: Google	3,332K	0	0
Tab: Google Docs - All items	18,040K	0	0
Tab: Paper Draft - 10/2/2008 - Google Docs			0
Tab: The New York Times - Breaking News, World N...	11,448K	0	0
Plug-in: Shockwave Flash	5,008K	0	N/A

- Some additional security (e.g., Chromium's sandbox)



Performance Impact

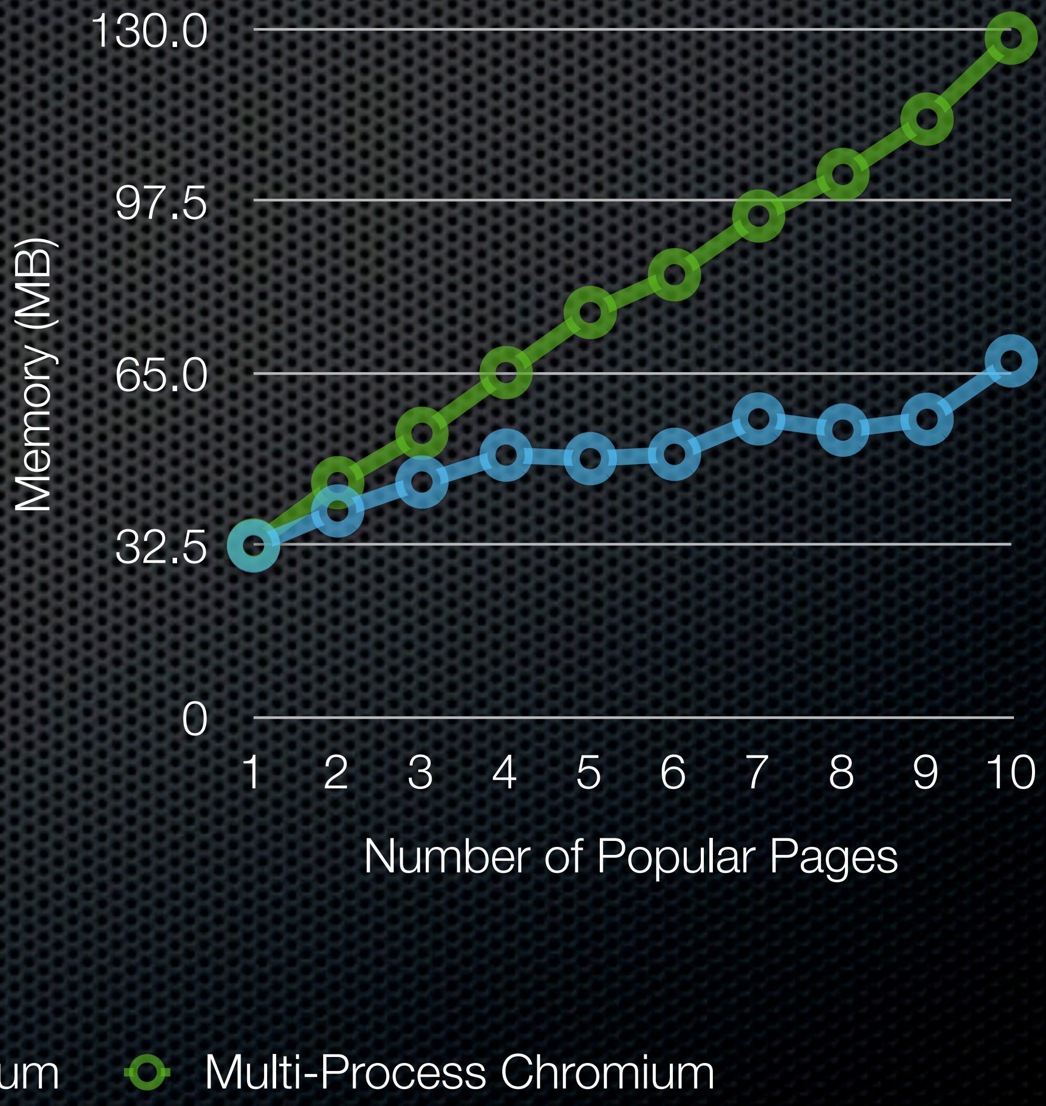
- **Responsiveness**



  - No delays while other pages are working
- **Speedups**
  - More work done concurrently, leveraging cores
- **Process Latency**
  - 100 ms, but masked by other speedups in practice

Memory Overhead

- Robustness benefits do have a cost
  - Reasonable for many real users



Summary

- Browsers must recognize programs to support them
  - Identify boundaries with **Site Instances**
  - **Compatible** with existing web content
  - Prevent interference with **process isolation**

More major browsers becoming multi-process:
IE8, possibly Firefox

Outline

Browser Architecture

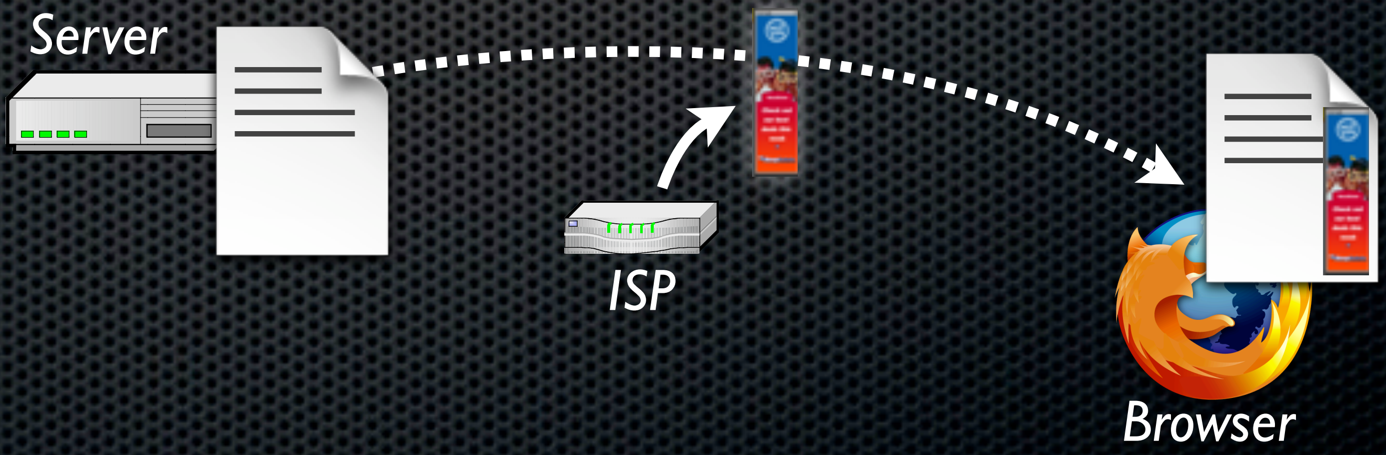
- **Web Tripwires**
  - Help publishers detect unauthorized code

Additional Contributions

Future Directions

Web Program Integrity

- Can users or publishers trust web program contents?
  - HTTP can be **modified in-flight**
  - Changes become part of the site instance

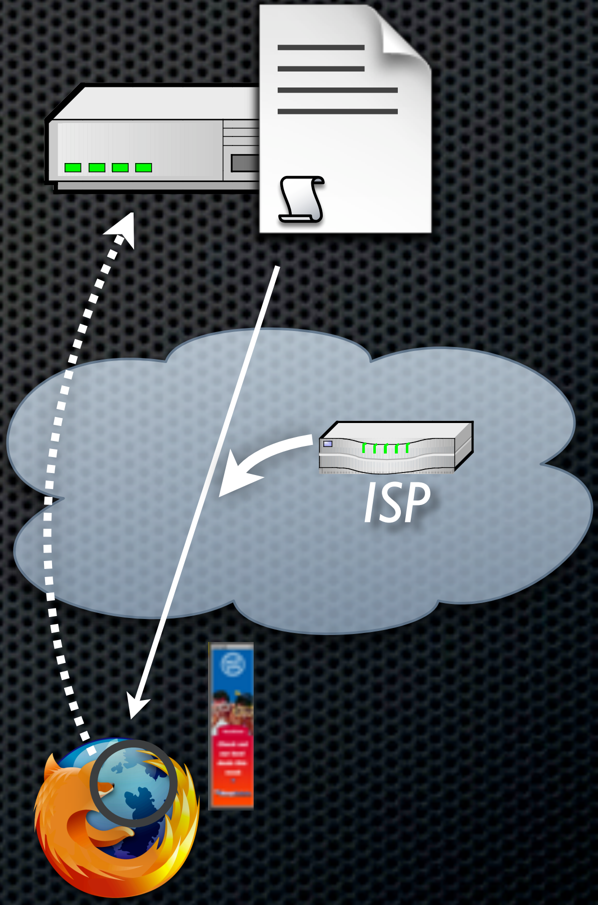


Is this a concern?

- **Measurements say it is!**
  - Of 50,000 clients, 1% saw in-flight changes
  - Results in **unauthorized program code**
  - Ads, exploits, broken pages, new vulnerabilities

Detecting Page Changes

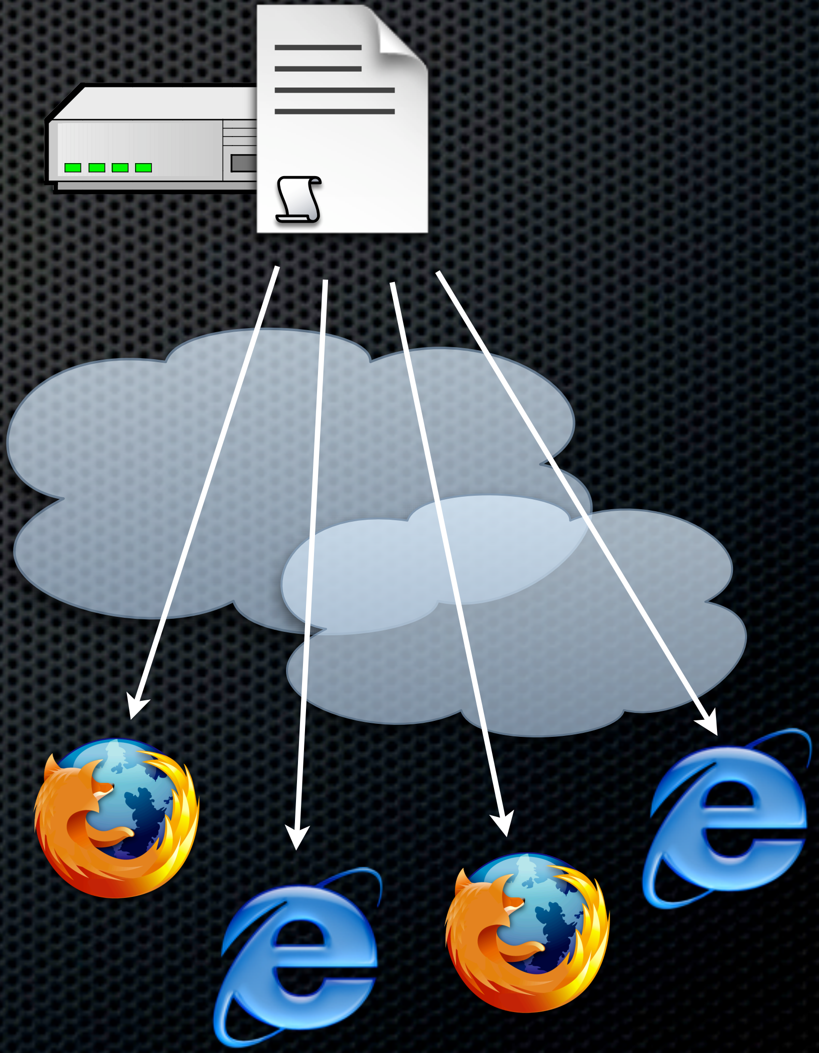
- Can detect with JavaScript
- Built a **Web Tripwire:**
  - Runs in client's browser
  - Finds most changes to HTML
  - Reports to user & server



http://vancouver.cs.washington.edu

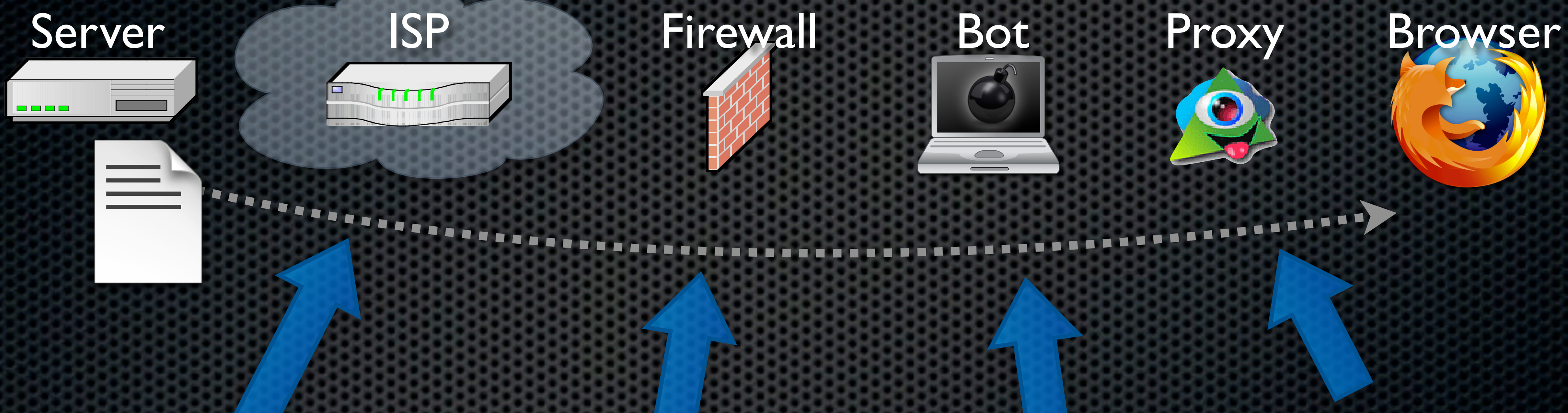
Measurement Study

- Wanted view of many clients on many networks
- Posted to **Slashdot**, **Digg**, etc.
  - Visits from over 50,000 unique IP addresses
  - 653 reported changes



http://vancouver.cs.washington.edu

Diverse Changes Observed



Ad Injection
(Free wireless,
NebuAd, etc)

Security Checks
(Enterprises)

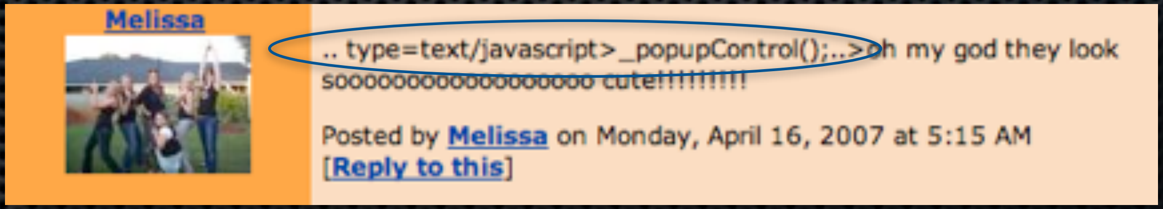
Exploits
(ARP poisoning)

**Ad / Popup
Blockers**
(on client)

http://vancouver.cs.washington.edu

The best intentions...

- **Bugs introduced**



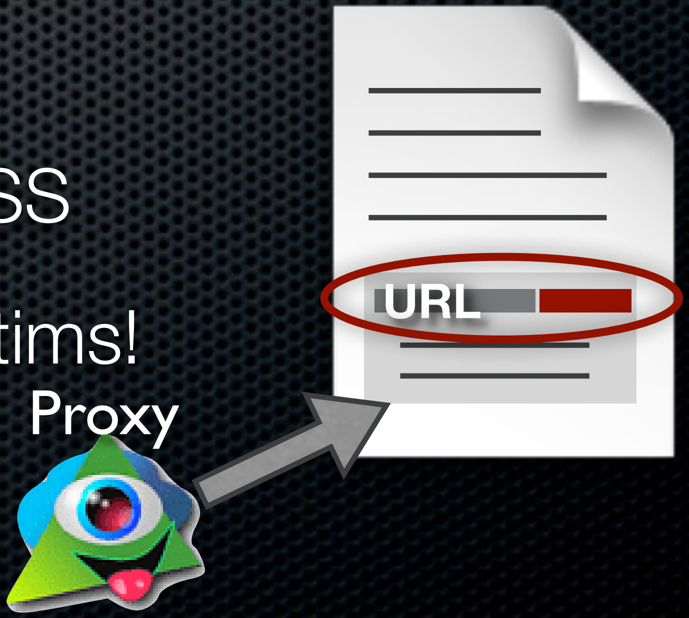
Melissa

.. type=text/javascript>_popupControl();..>oh my god they look sooooooooooooooooooo cute!!!!!!!!!!

Posted by Melissa on Monday, April 16, 2007 at 5:15 AM
[Reply to this]

  - Web forums broken by popup blockers

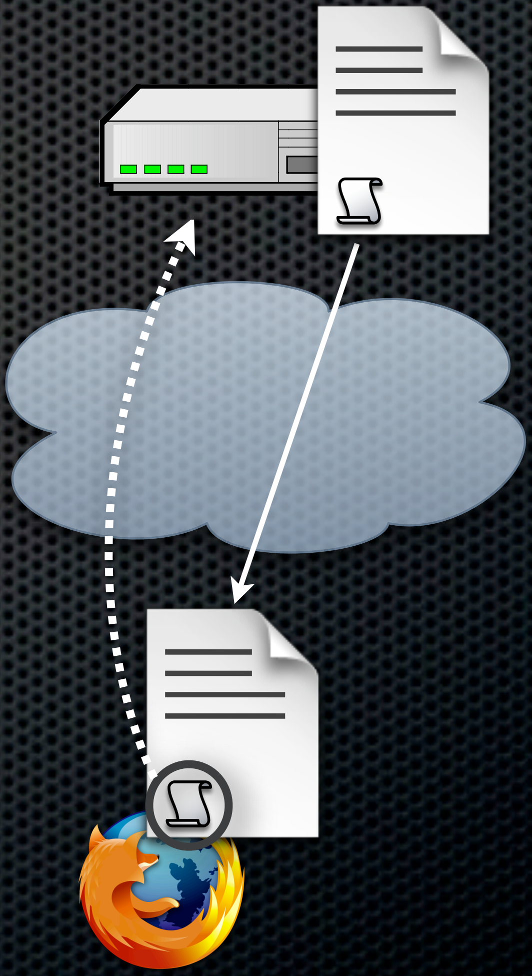
- **Vulnerabilities introduced**
  - Ad blocker code vulnerable to XSS
  - User’s web programs are the victims!



http://vancouver.cs.washington.edu

Web Tripwires for Publishers

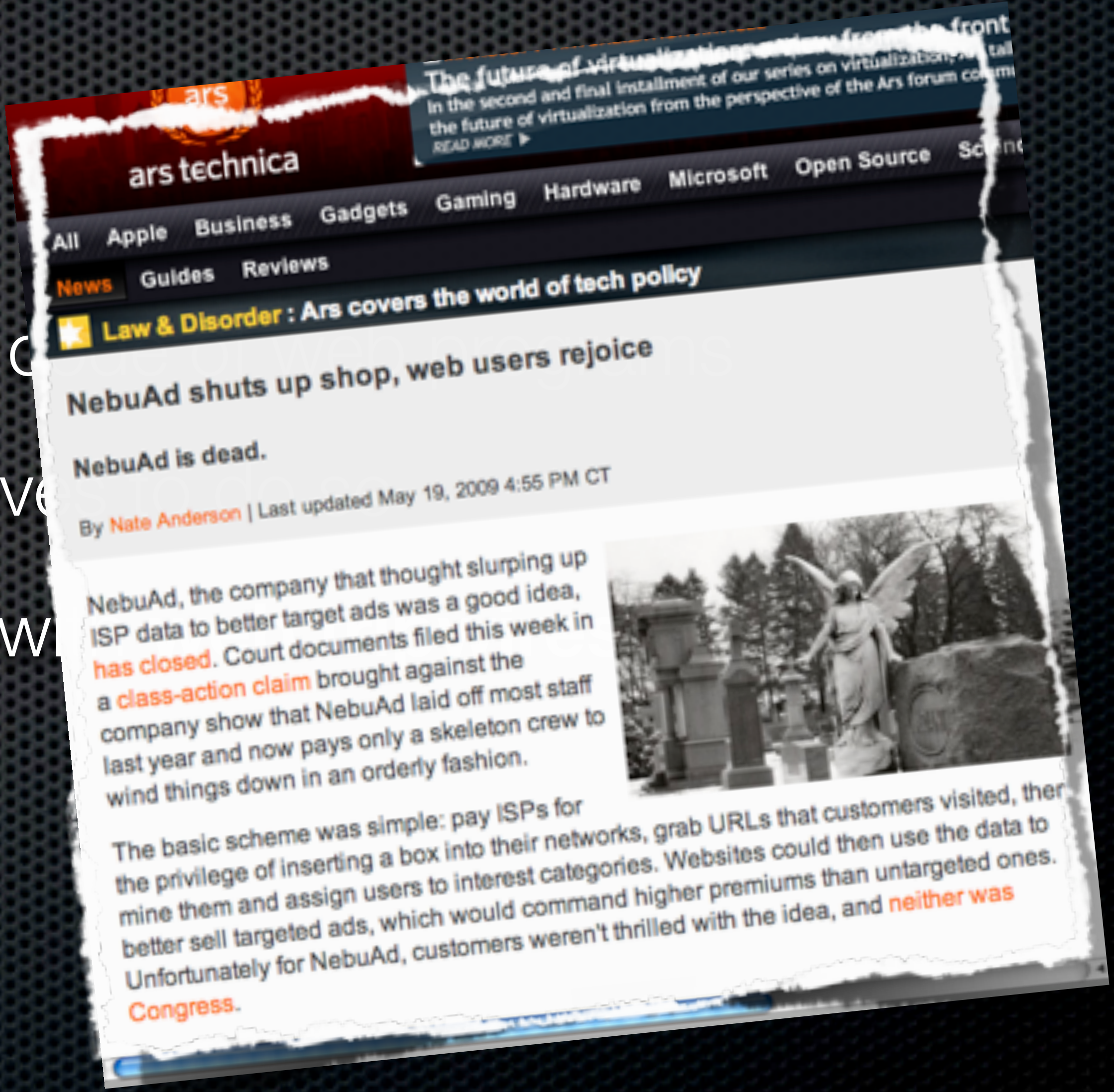
- HTTPS too costly for some sites
- Can detect changes with JavaScript
- Easy for publishers to deploy
  - **Configurable toolkit**
  - **Web tripwire service**



http://vancouver.cs.washington.edu

Summary

- Not safe to blindly patch c
- Many parties with incentive
- Publishers may detect it wi



The future of virtualization: a view from the front

In the second and final installment of our series on virtualization, Ars tal the future of virtualization from the perspective of the Ars forum comm

Law & Disorder : Ars covers the world of tech policy

NebuAd shuts up shop, web users rejoice

NebuAd is dead.

By Nate Anderson | Last updated May 19, 2009 4:55 PM CT

NebuAd, the company that thought slurping up ISP data to better target ads was a good idea, has closed. Court documents filed this week in a class-action claim brought against the company show that NebuAd laid off most staff last year and now pays only a skeleton crew to wind things down in an orderly fashion.

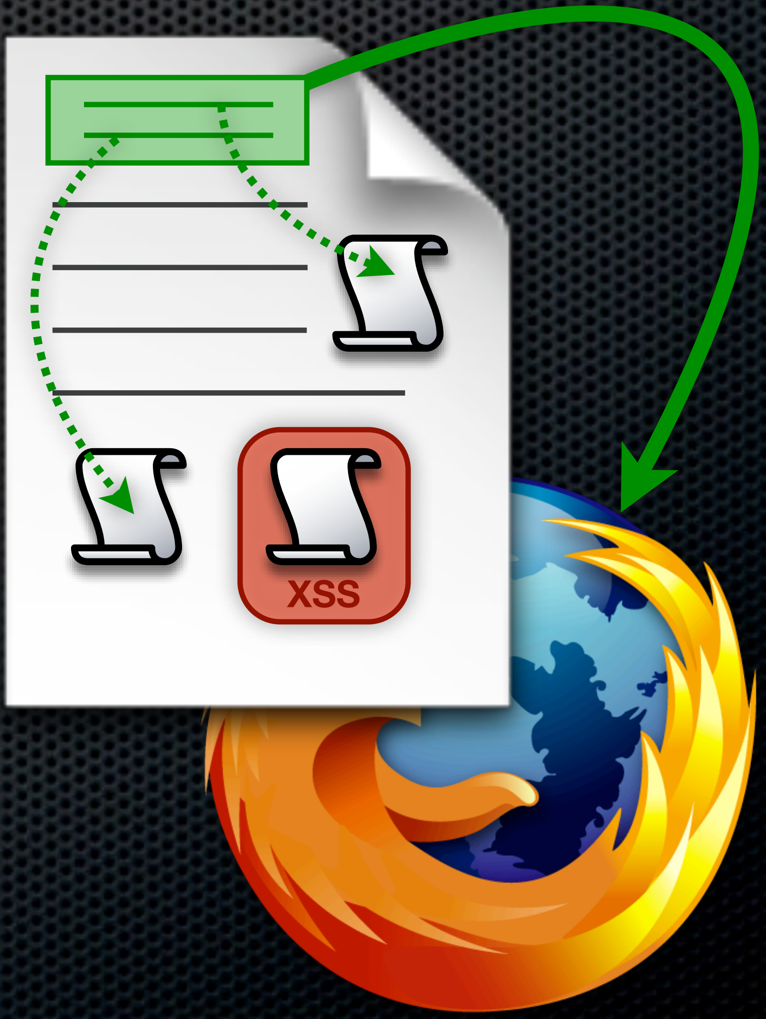
The basic scheme was simple: pay ISPs for the privilege of inserting a box into their networks, grab URLs that customers visited, ther mine them and assign users to interest categories. Websites could then use the data to better sell targeted ads, which would command higher premiums than untargeted ones. Unfortunately for NebuAd, customers weren't thrilled with the idea, and neither was Congress.

Outline

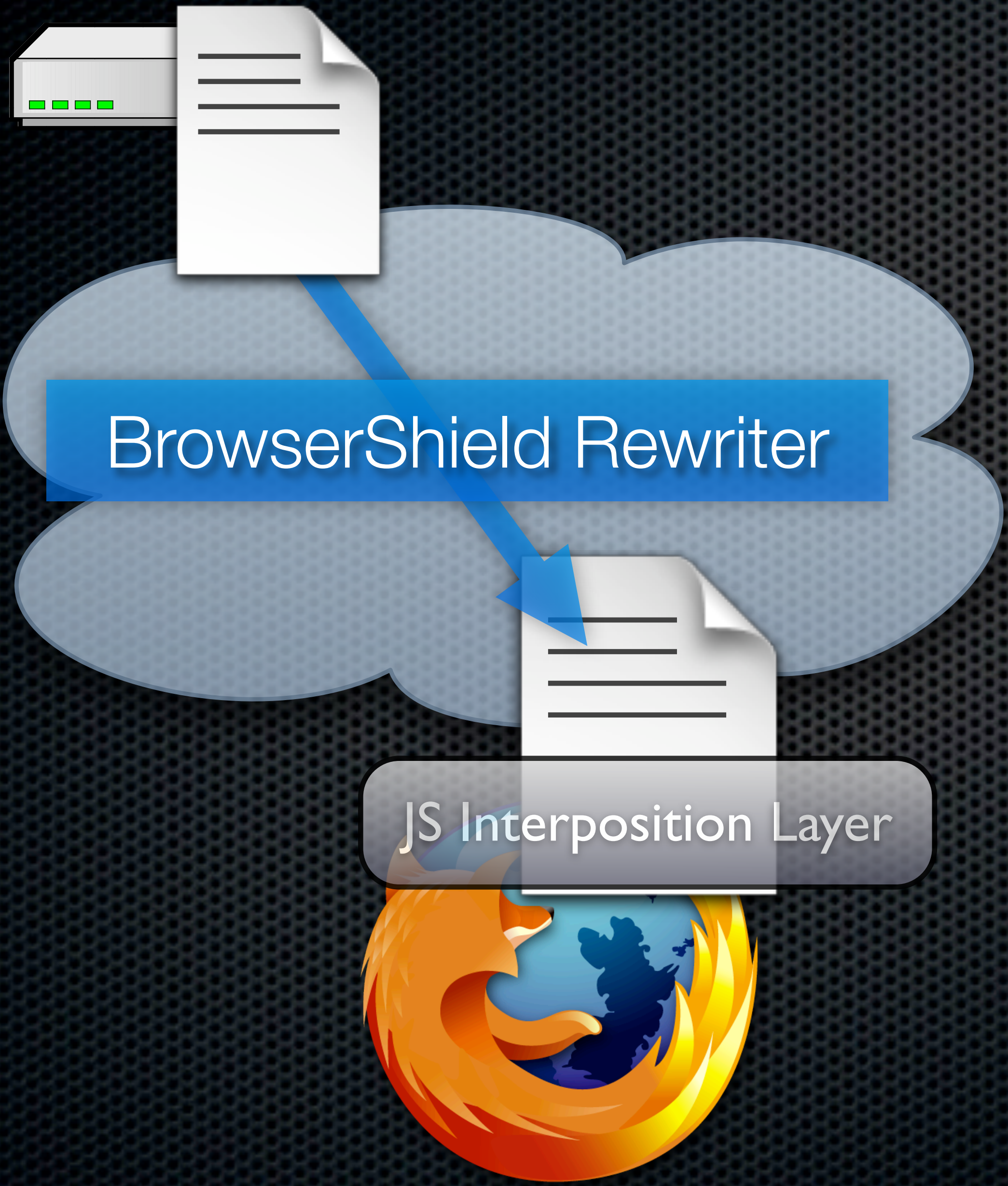
- Browser Architecture
- Web Tripwires
- **Additional Contributions**
- Future Directions

Script Whitelists

- Injected scripts hijack pages
- Server defenses: *fail-open*
- **Authorize code** with whitelists: *fail-closed*
  - Enforced by browser
  - Handles realistic pages



BrowserShield [OSDI '06]



- **Block exploits** of known browser vulnerabilities
    - Interpose to **enforce flexible policies**
- Rewrites JavaScript code in-flight...
- Has influenced Live Labs' Web Sandbox

Thesis:

Adapt lessons from the OS
to improve robustness and security
of web browsers and web content

- **Added support for four architectural principles:**
  1. Identify program boundaries
  2. Isolate programs from each other
  3. Authorize program code
  4. Enforce policies on program behavior

Outline

- Browser Architecture
- Web Tripwires
- Additional Contributions
- **Future Directions**

Future Browsers & Programs

- **Convergence of Browsers and OSes**
  - More powerful features for web programs
  - More effective program definitions
  - Potential for new OS mechanisms
- **Access programs in cloud from diverse devices**
  - Trust models? Customization?

Better Support for Principles

- **Defining explicit boundaries** for web programs
  - e.g., Alternatives to Same Origin Policy
- **Securely + Compatibly isolating** Site Instances
- **Authorizing active code** of any format
- **Enforcing policies** on content, plug-ins, extensions

Conclusion

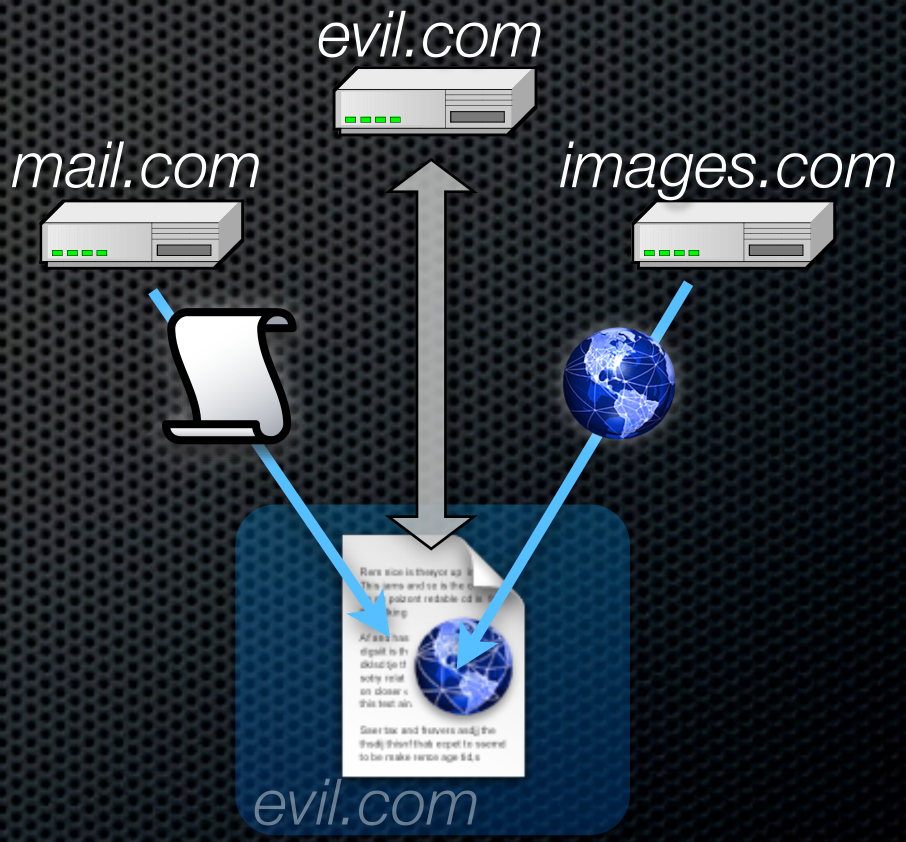
- Web is becoming an **application platform**
  - Browser architectures must **support programs**
  - Web publishers must **protect content**
- **Great opportunity to reshape the web**

Compatibility Compromises

- **Coarse granularity**
  - Some logical apps grouped together (instances help)
- **Imperfect isolation**
  - Shared cookies, some window-level JS calls
- **Not a secure boundary**
  - Must still rely on renderer to prevent certain leaks

Relevant for security?

- **Pages are free to embed objects from any site**
  - Scripts, images, plugins
  - Carry user's credentials
  - *Inaccessible info within each Site Instance*
- **Compatibility makes us rely on internal logic**

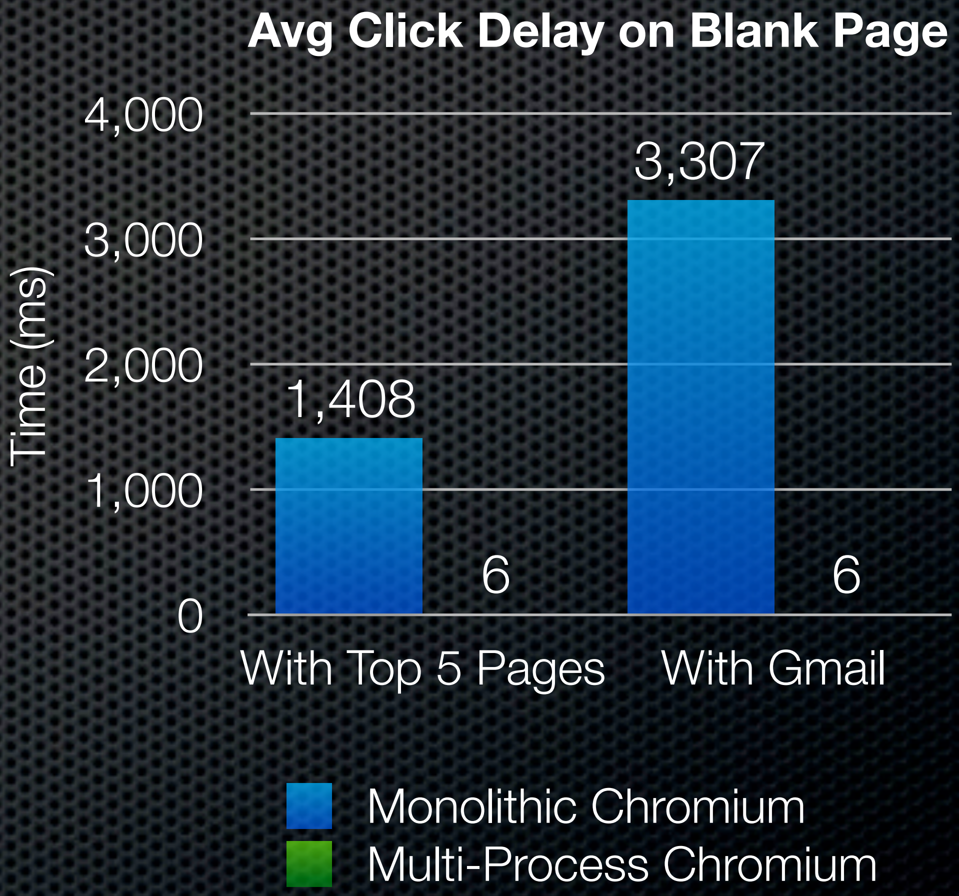


Implementation Caveats

- **Sites may sometimes share processes**
  - Not all cross-site navigations change processes
  - Frames still in parent process
  - Process limit (20), then randomly re-used

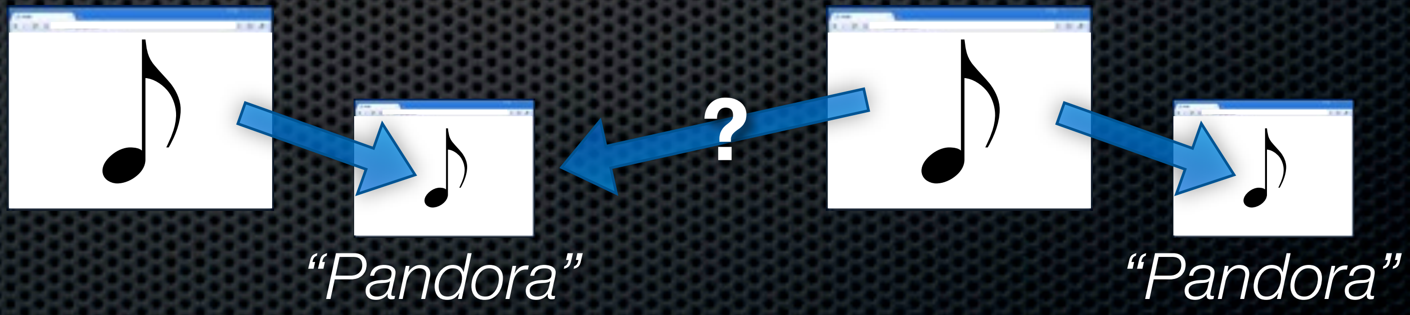
Performance Isolation

- **Responsive** while other web programs working



Compatibility Evaluation

- No known compat bugs due to architecture
- Some minor behavior changes
  - e.g., **Narrower scope of window names:** browsing instance, not global



Related Architecture Work

- **Internet Explorer 8**
  - Multi-process architecture, no program abstractions
- **Gazelle**
  - Like Chromium, but values security over compatibility
- **Other research: OP, Tahoma, SubOS**
  - Break compatibility (isolation too fine-grained)