Improving the Security and Robustness of Modern Web Browsers

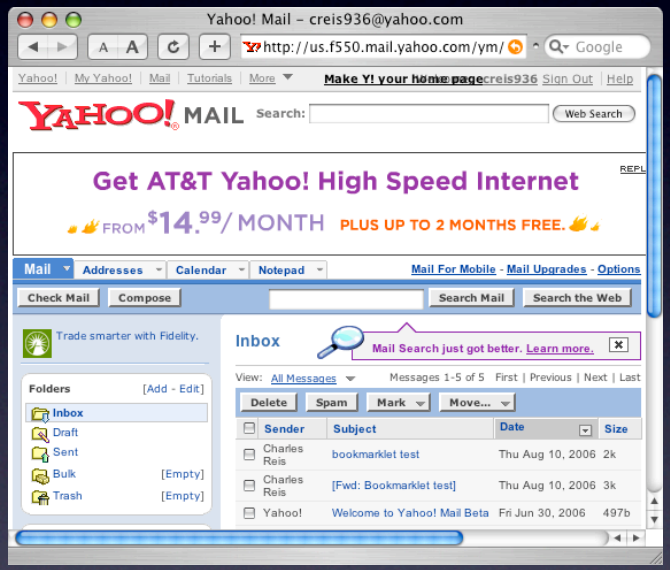
Charles Reis

General Examination

May 14, 2007

Is Browsing Safe?

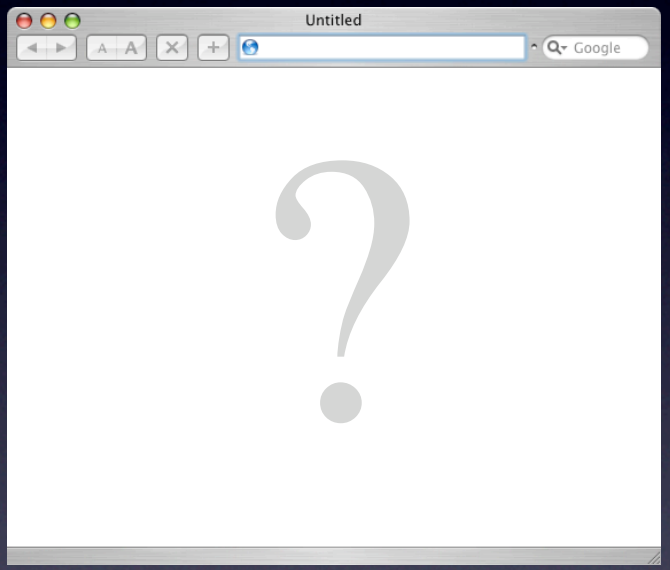
Is Browsing Safe?



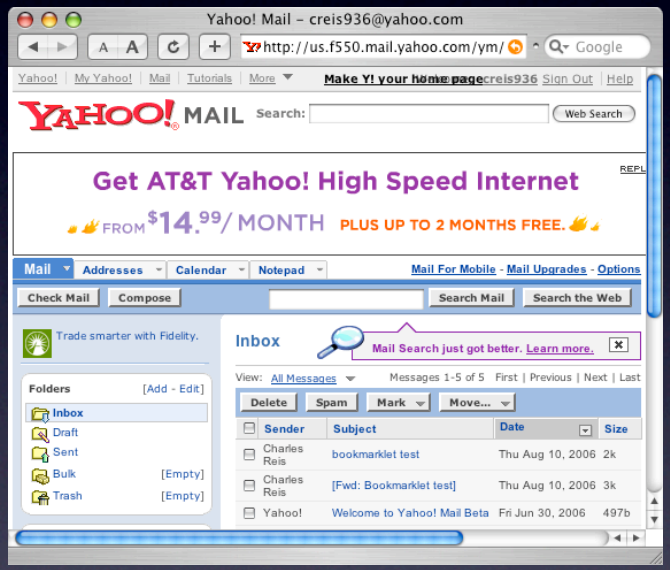
Web Mail

Movie Rentals

Search Results



Is Browsing Safe?

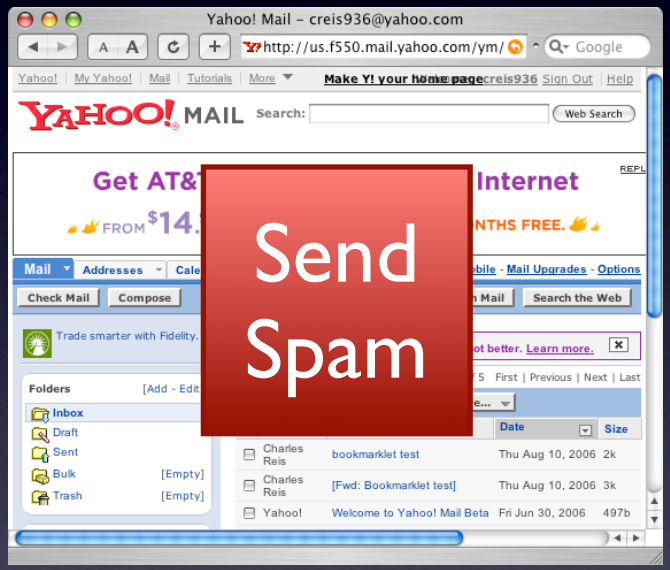


Web Mail

Movie Rentals

Search Results

Is Browsing Safe?

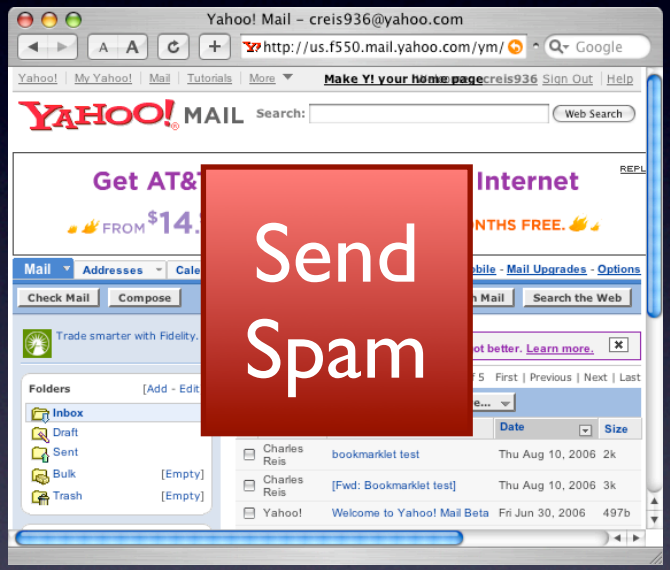


Web Mail

Movie Rentals

Search Results

Is Browsing Safe?



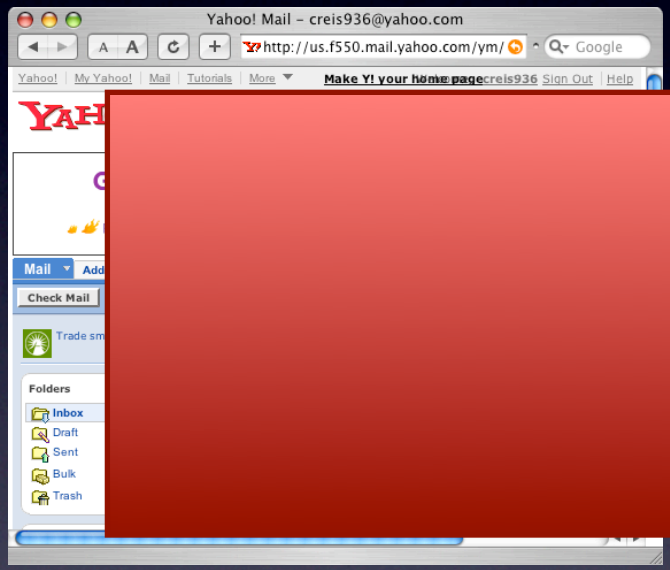
Web Mail

Movie Rentals

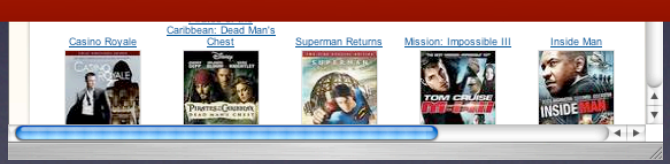
Search Results

Is Browsing Safe?

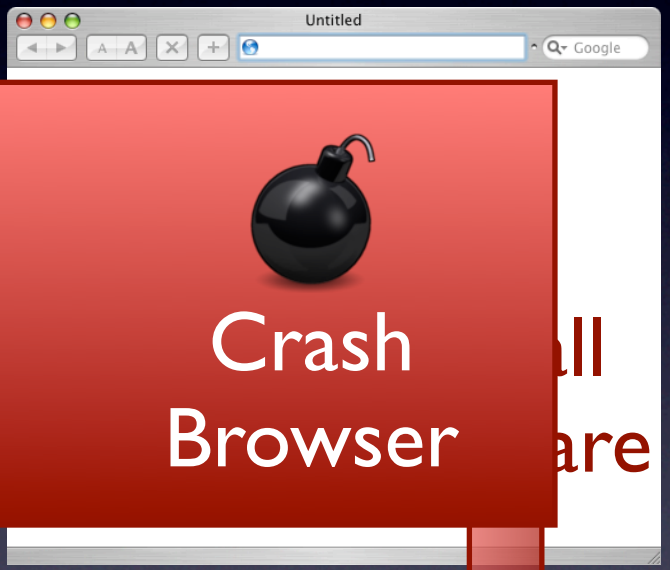
Search Results



Web Mail



Movie Rentals

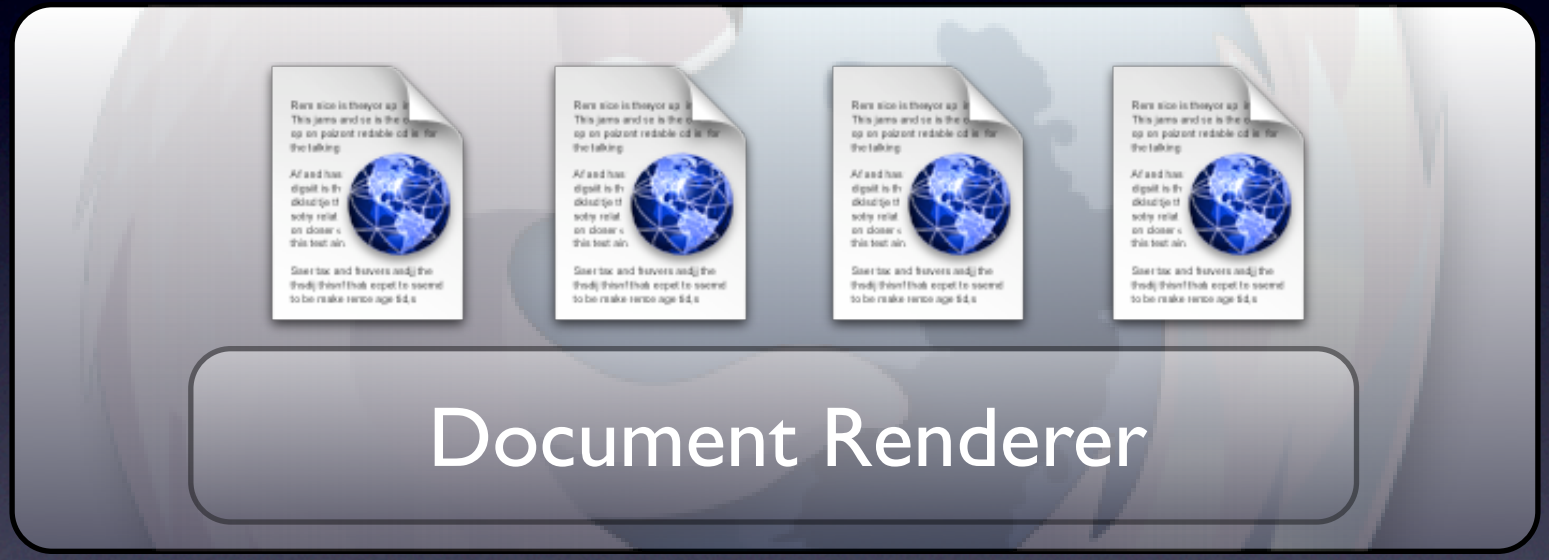


Crash Browser

How did we get here?

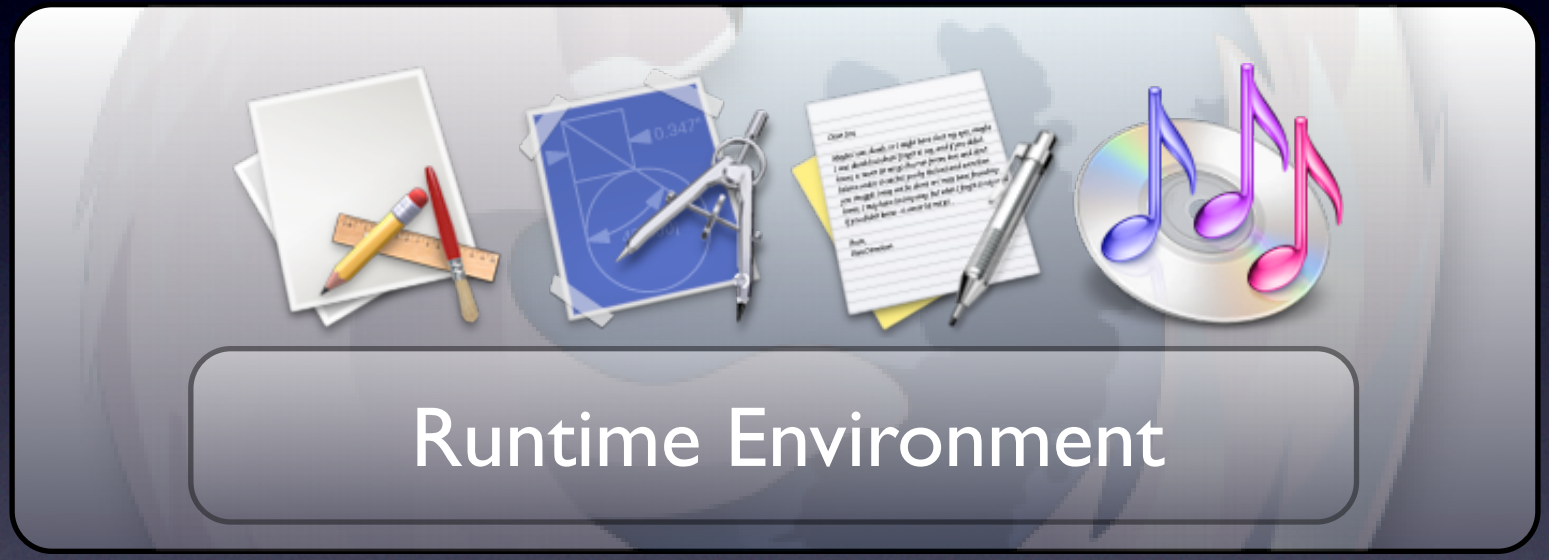
How did we get here?

- Browsers have evolved



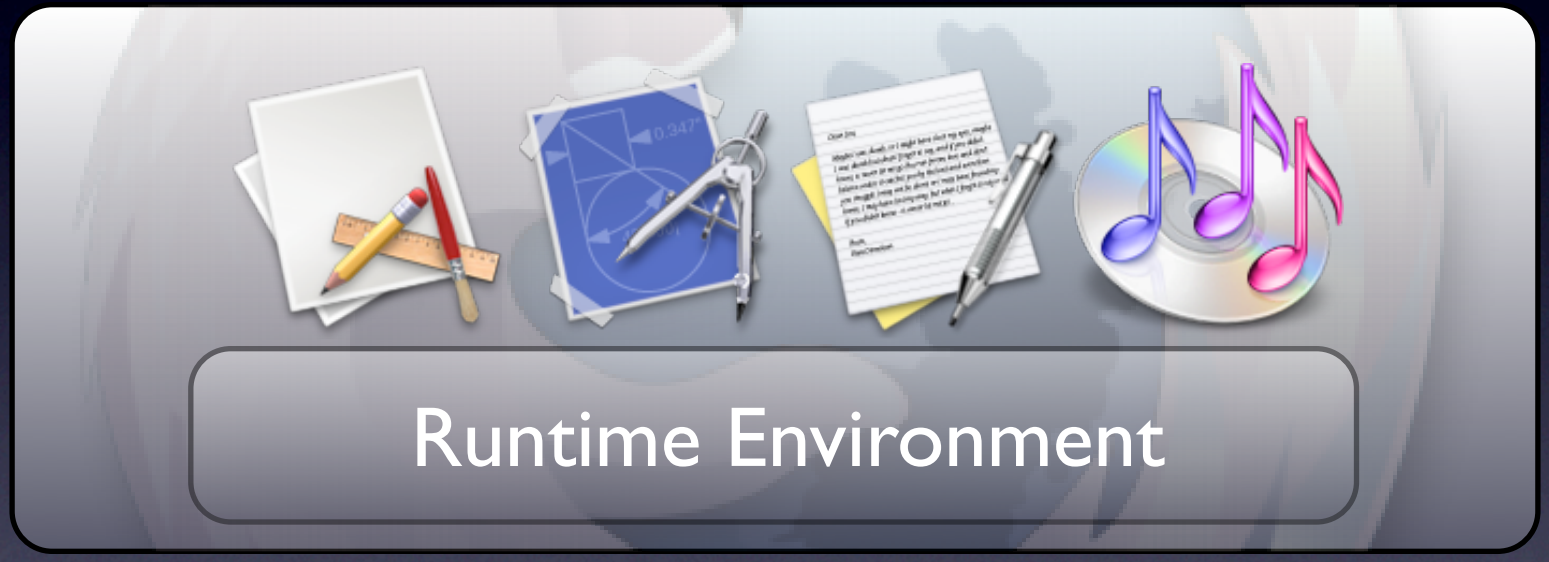
How did we get here?

- Browsers have evolved



How did we get here?

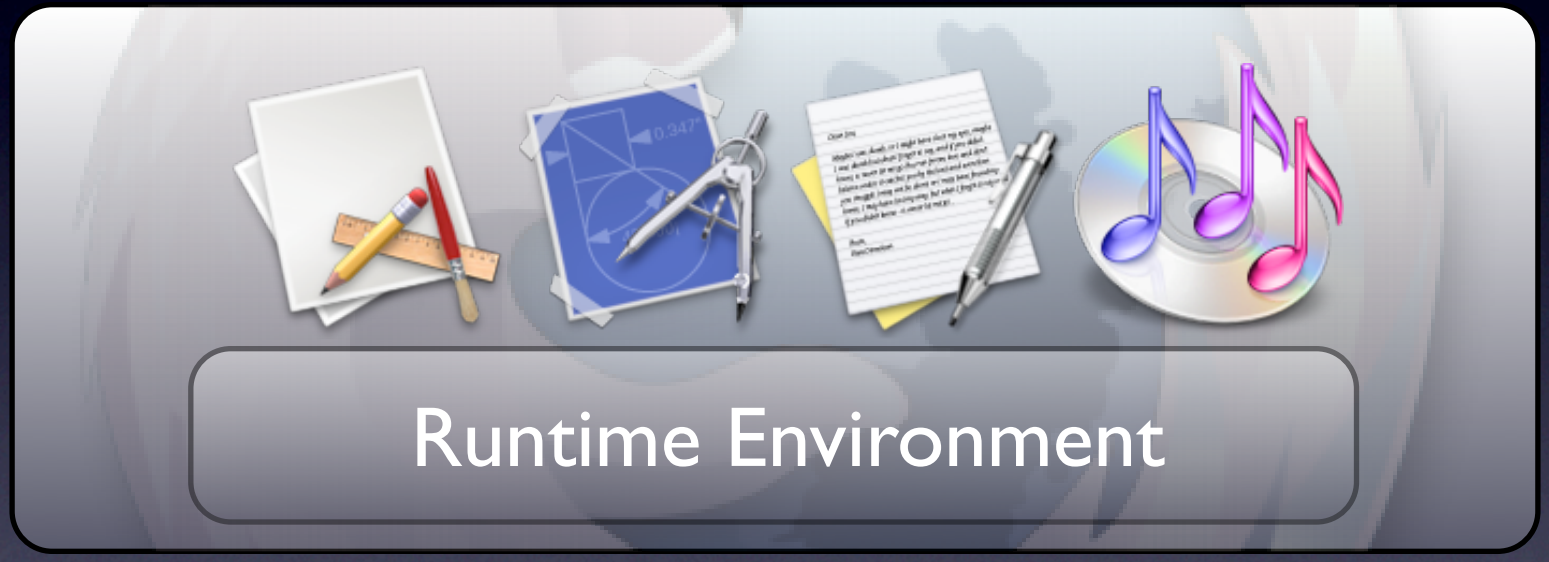
- Browsers have evolved



- Now analogous to OS

How did we get here?

- Browsers have evolved



- Now analogous to OS
- Convenience has trumped security

Hypothesis

Mechanisms from OS research can improve the security and robustness of modern web browsers.

Hypothesis

Mechanisms from OS research can improve the security and robustness of modern web browsers.

- Focus on **Isolation** and **Interposition**

Hypothesis

Mechanisms from OS research can improve the security and robustness of modern web browsers.

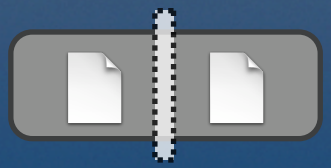
- Focus on **Isolation** and **Interposition**
- Evaluate **Safety**, **Backwards Compatibility**, and **Efficiency**

Outline

Motivation

Threats

Isolation

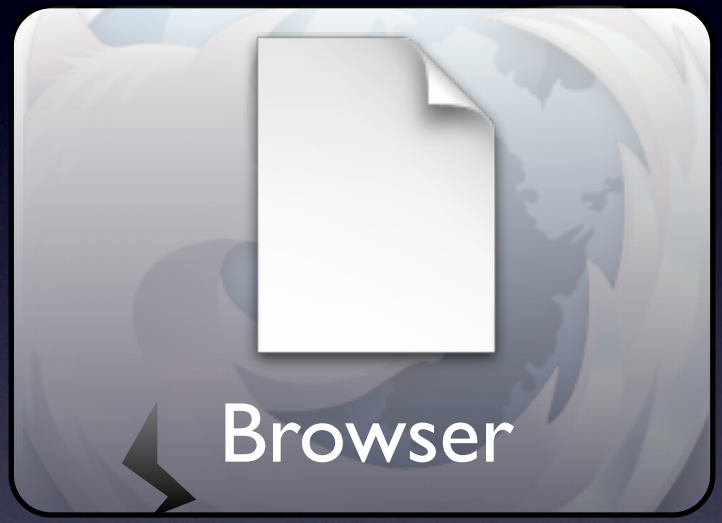


Interposition

Future Work

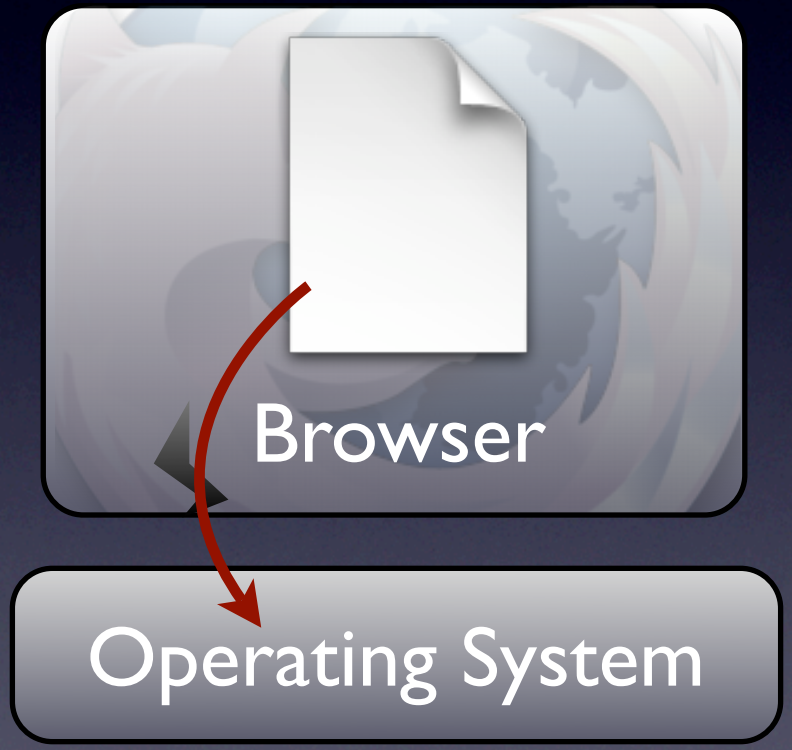
Conclusion

1. Browser Vulnerabilities



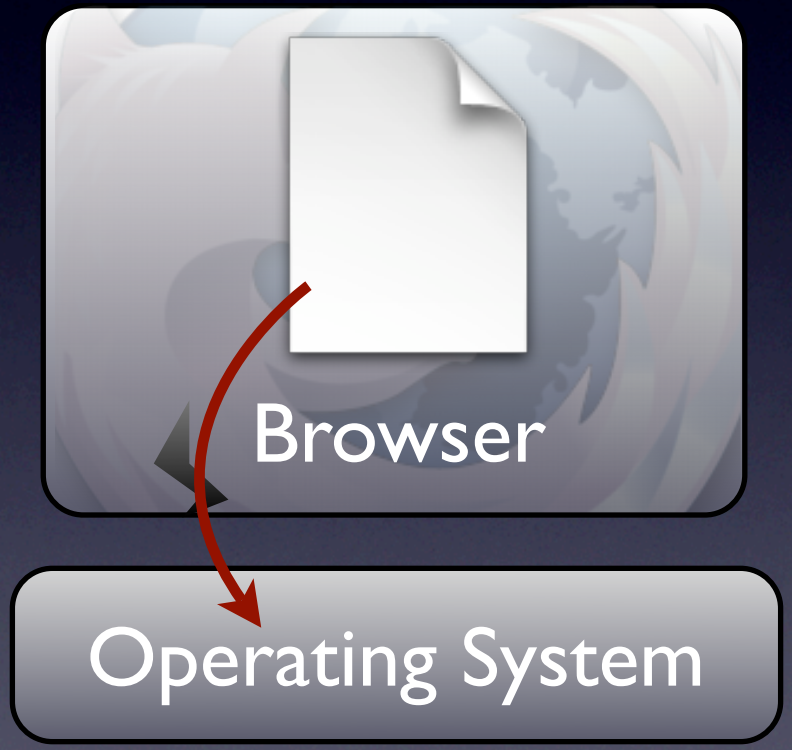
Operating System

1. Browser Vulnerabilities



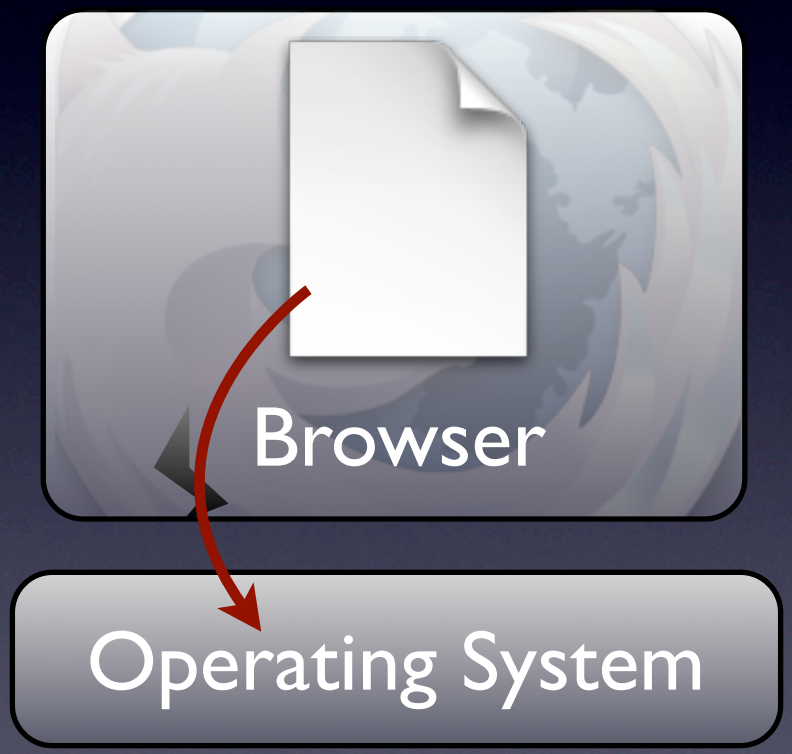
- Can run arbitrary code
- Windows .ANI bug

1. Browser Vulnerabilities



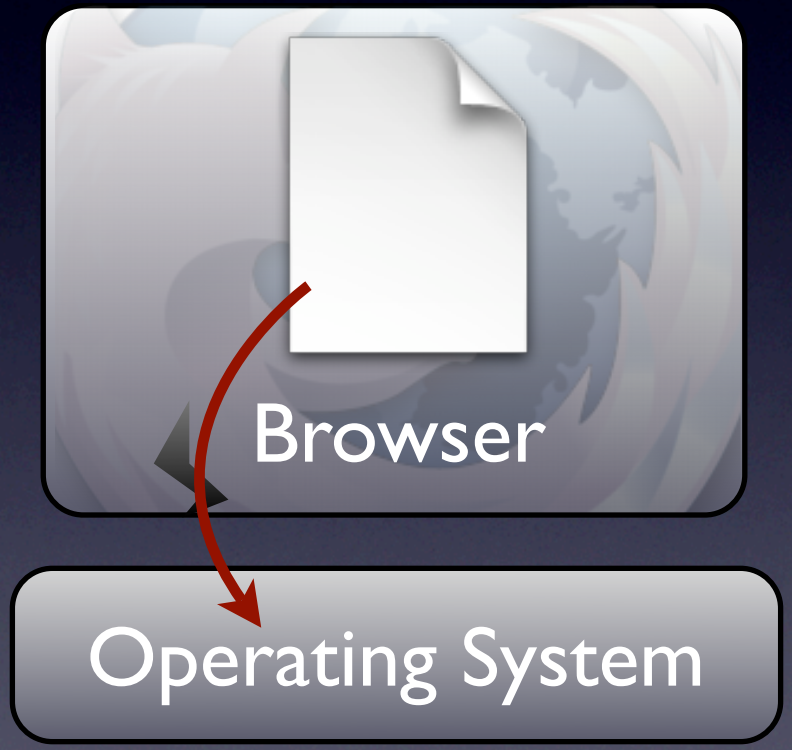
- Can run arbitrary code
  - Windows .ANI bug
- Patches get delayed

1. Browser Vulnerabilities



- Can run arbitrary code
  - Windows .ANI bug
- Patches get delayed
- **Existing Proposals**:
  VM sandboxes *[Tahoma]*,
  pre-screening *[SpyProxy]*,
  legal teams *[HoneyMonkey]*

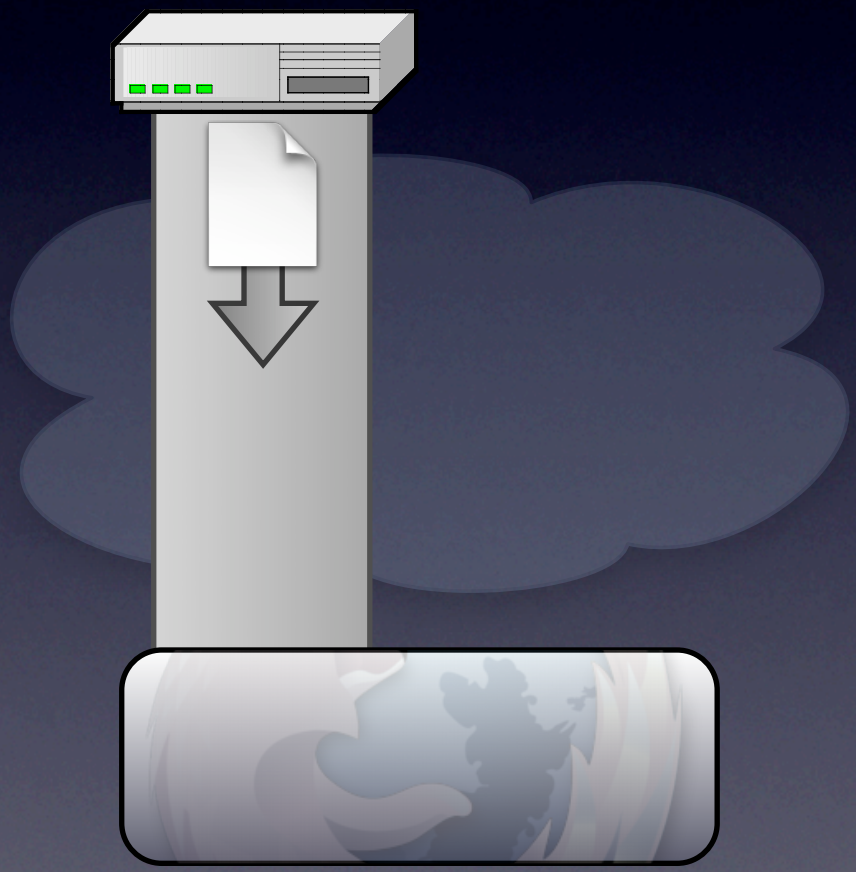
1. Browser Vulnerabilities



- Can run arbitrary code
  - Windows .ANI bug
- Patches get delayed
- **Existing Proposals**: VM sandboxes *[Tahoma]*, pre-screening *[SpyProxy]*, legal teams *[HoneyMonkey]*
- **Proposal**: filter exploits

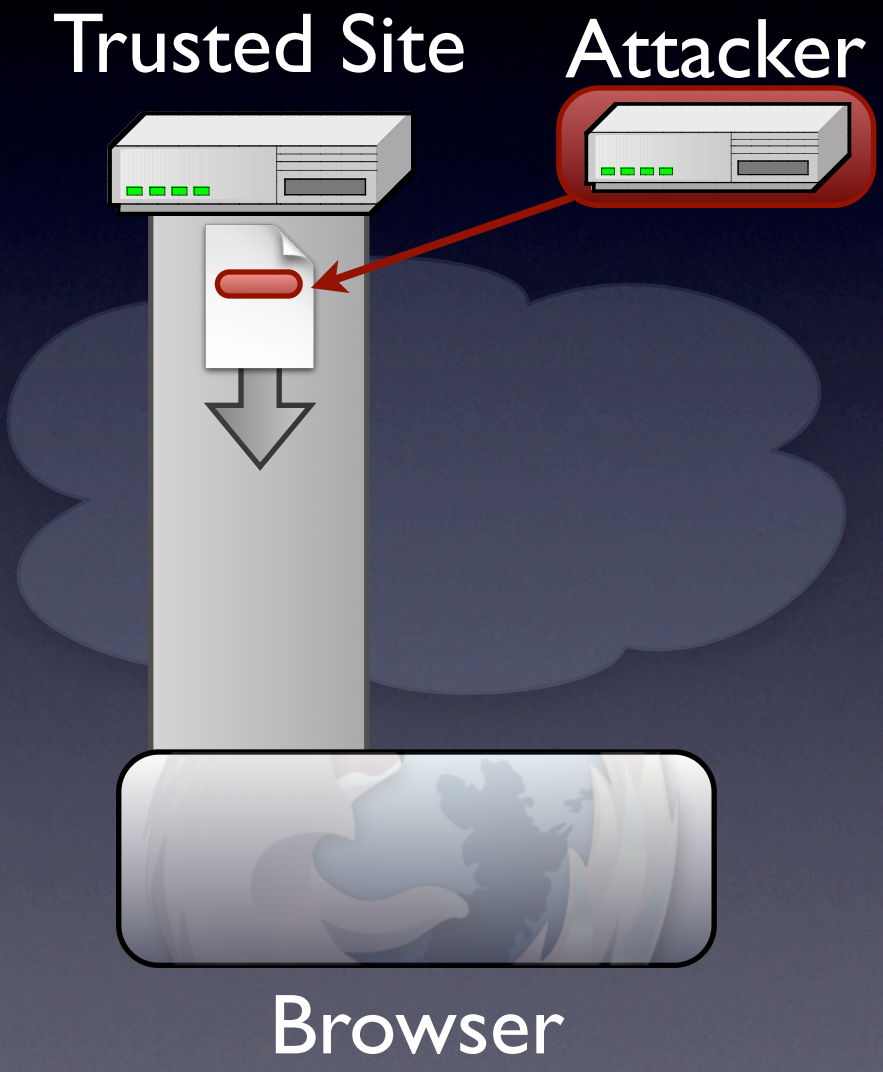
2. Script Injection (XSS)

Trusted Site

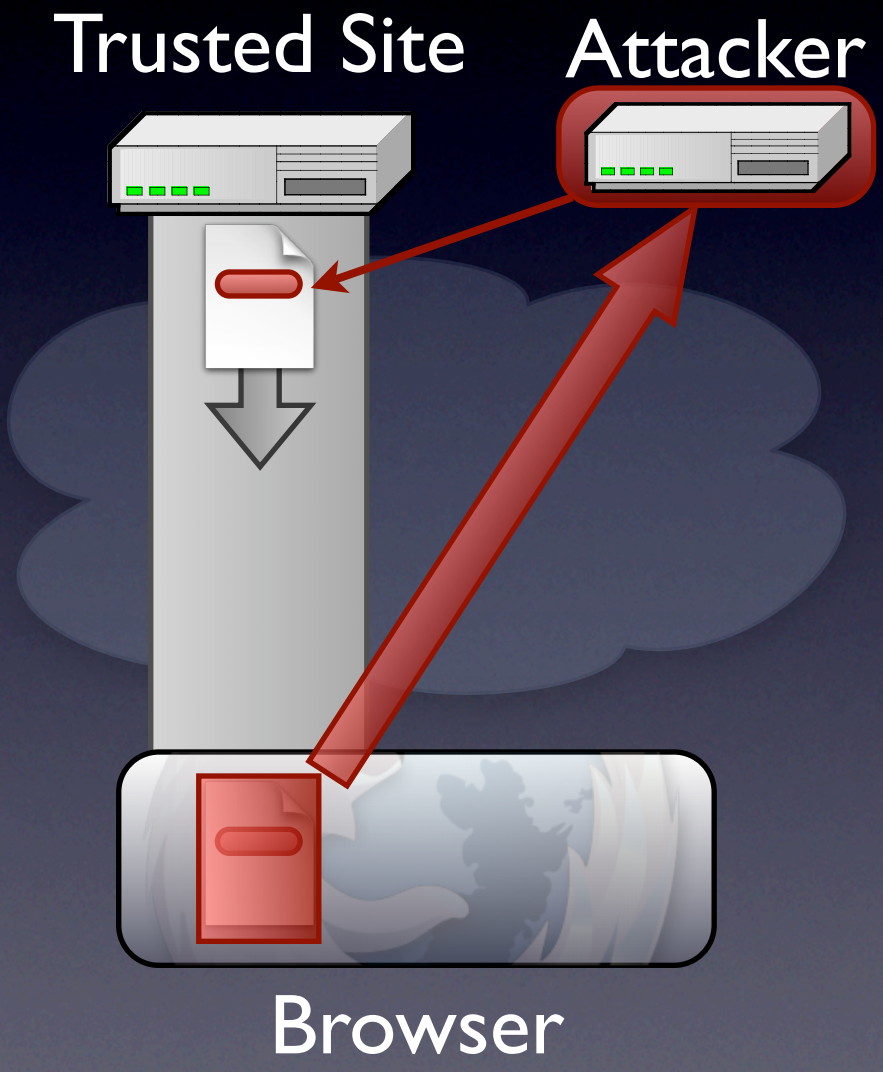


Browser

2. Script Injection (XSS)

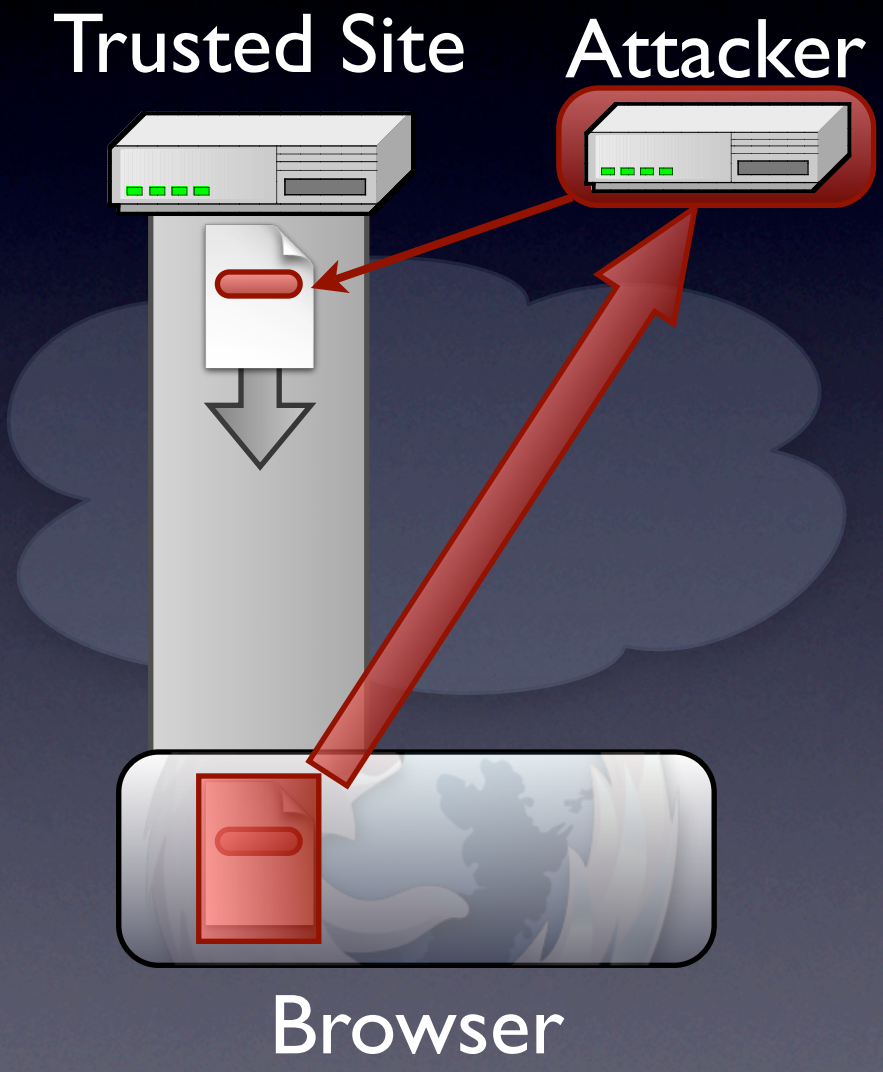


2. Script Injection (XSS)



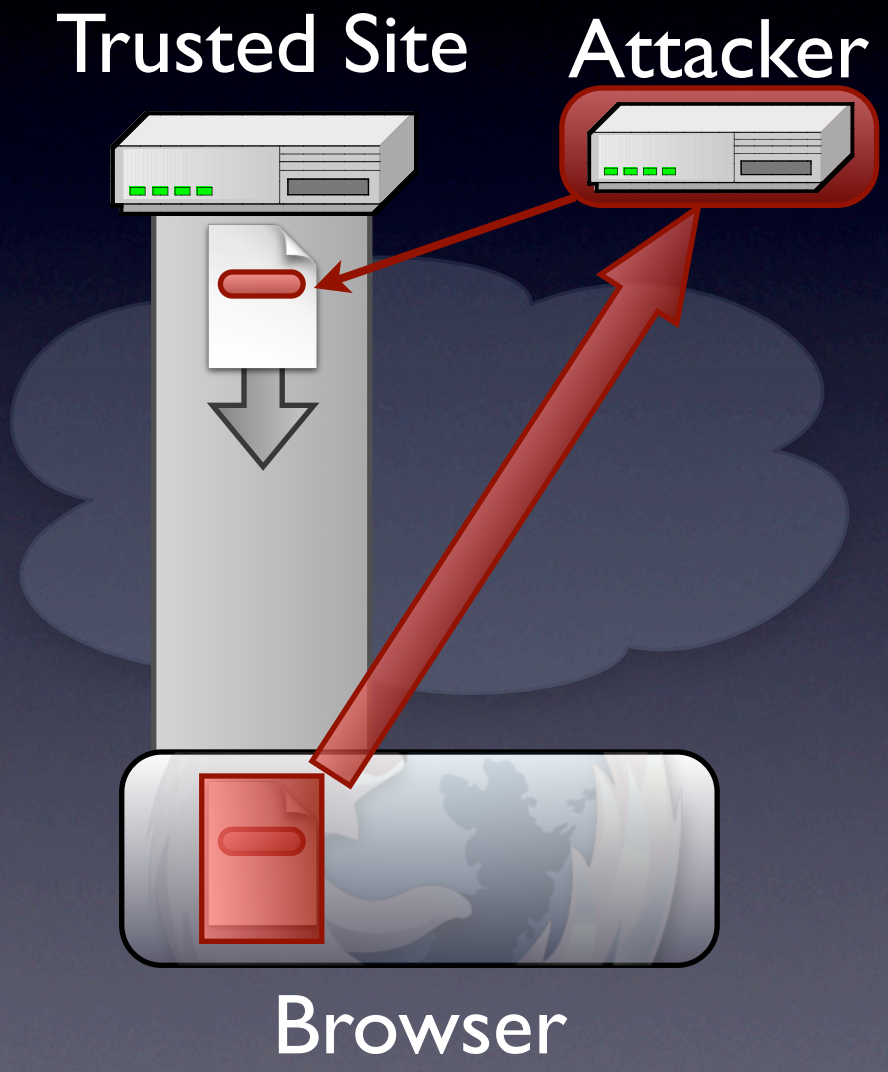
- Can subvert trusted sites
- Yahoo Mail, MySpace

2. Script Injection (XSS)



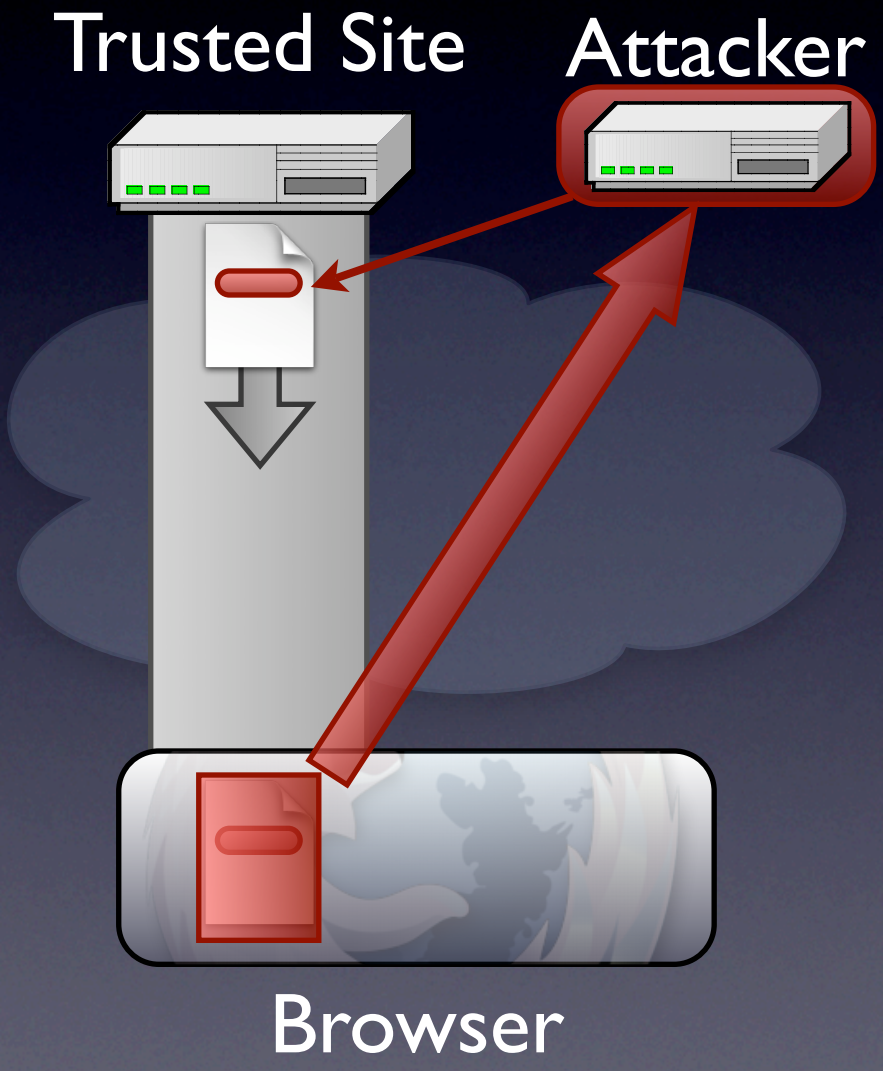
- Can subvert trusted sites
  - Yahoo Mail, MySpace
- Input validation is hard

2. Script Injection (XSS)



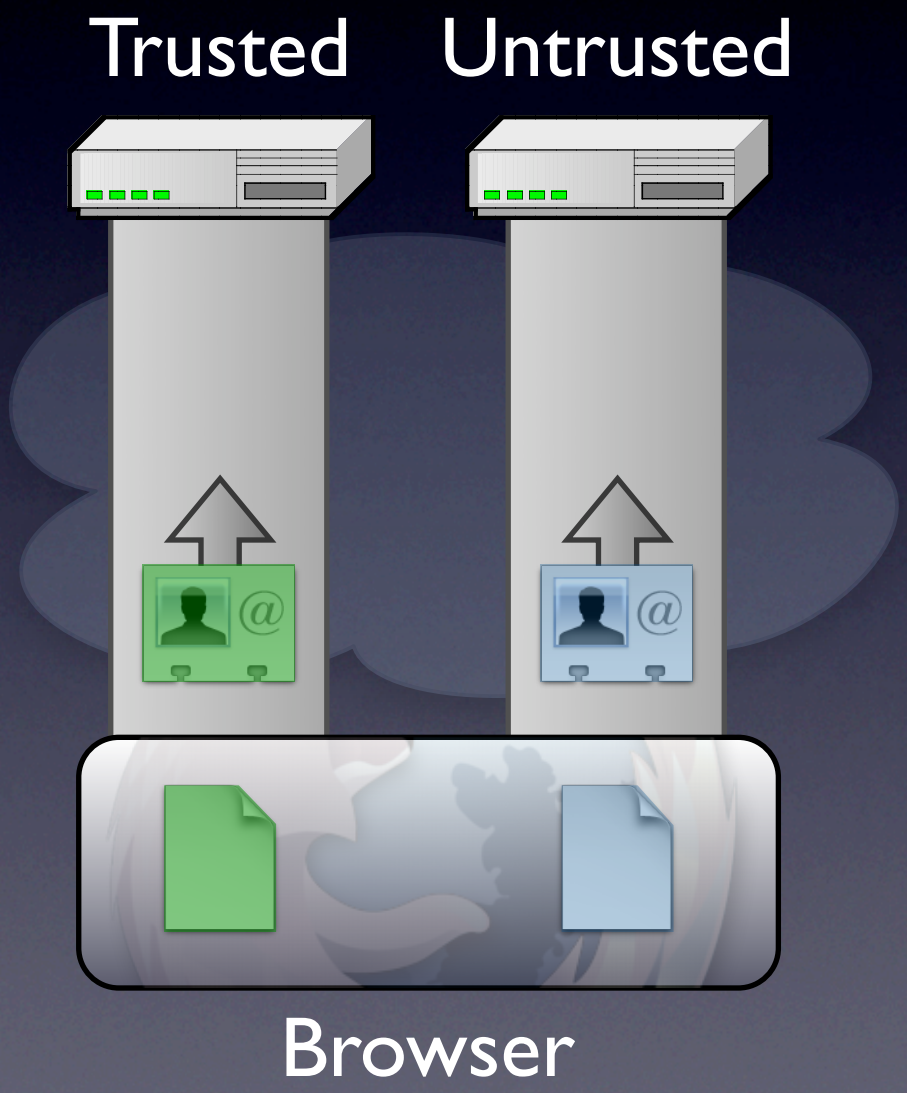
- Can subvert trusted sites
  - Yahoo Mail, MySpace
- Input validation is hard
- **Existing Proposals**: Server-side analysis *[Xie 06]*, client firewalls *[Noxes]*

2. Script Injection (XSS)

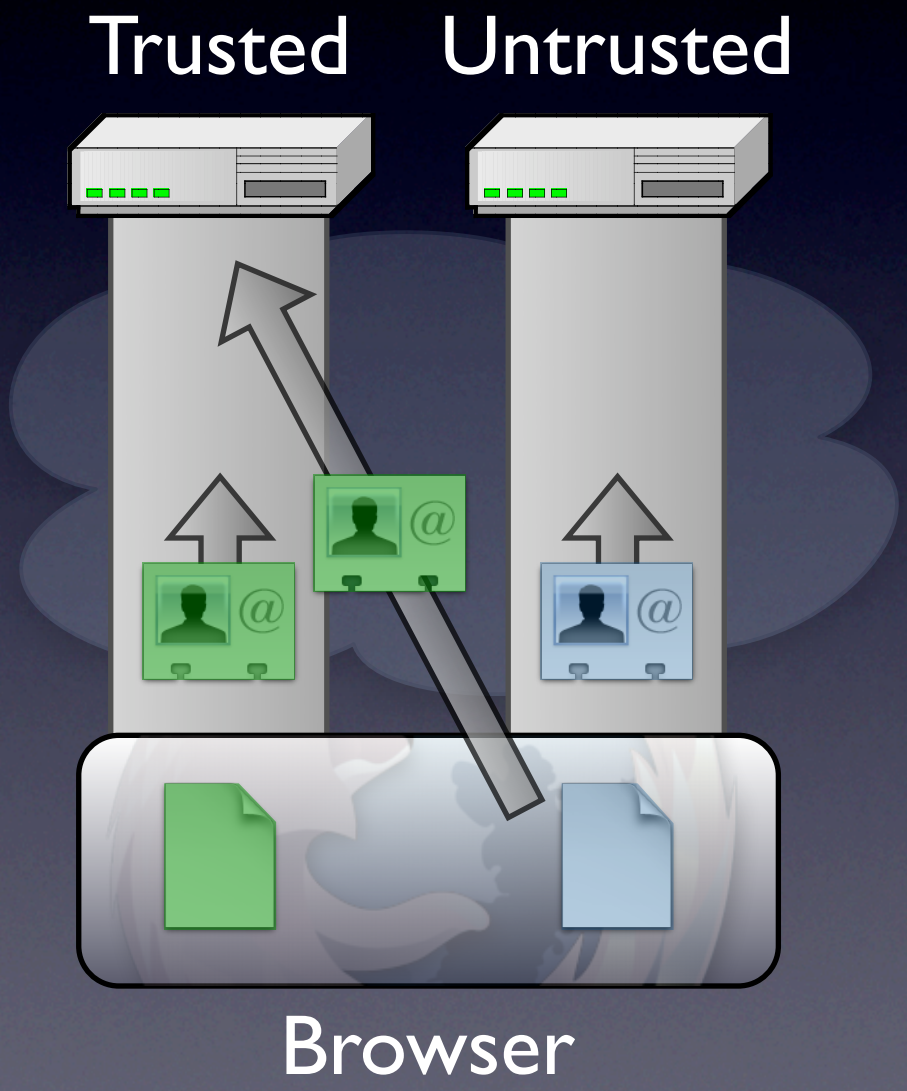


- Can subvert trusted sites
  - Yahoo Mail, MySpace
- Input validation is hard
- **Existing Proposals**: Server-side analysis *[Xie 06]*, client firewalls *[Noxes]*
- **Proposal**: whitelists

3. Cross-Site Request Forgery (CSRF)

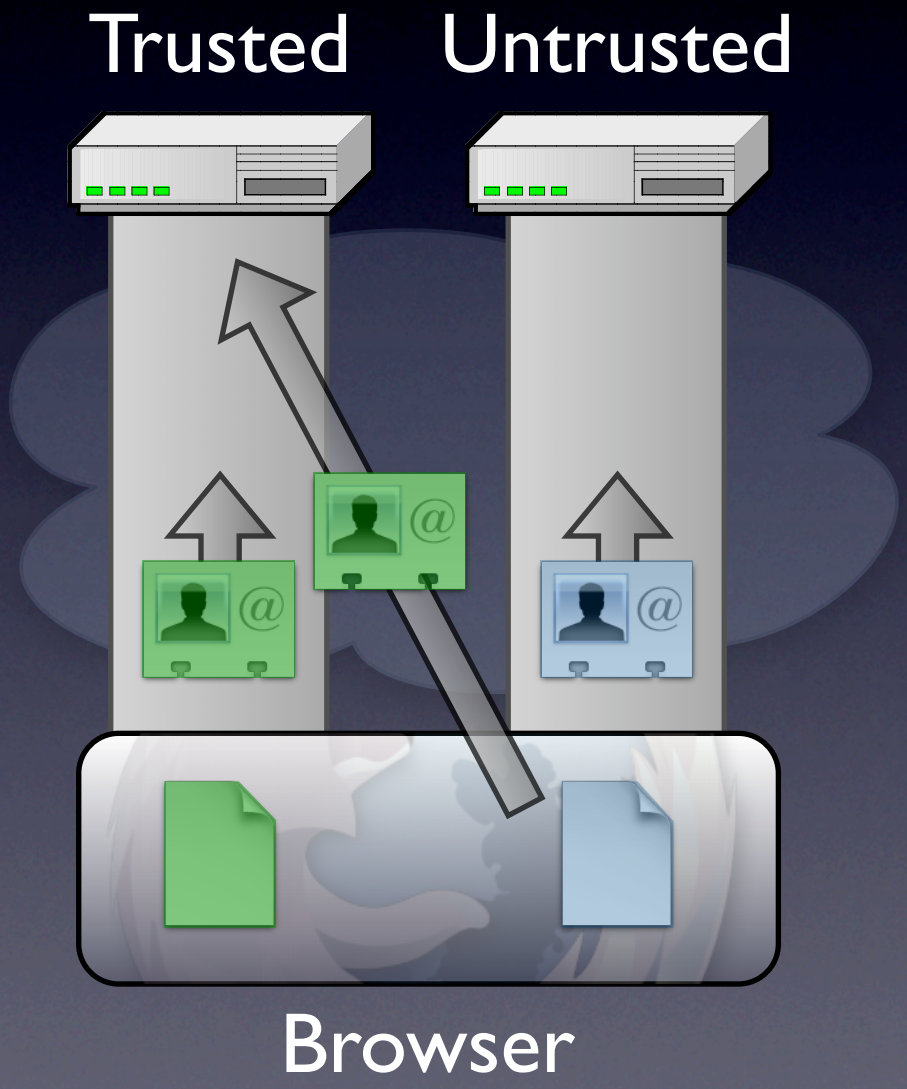


3. Cross-Site Request Forgery (CSRF)



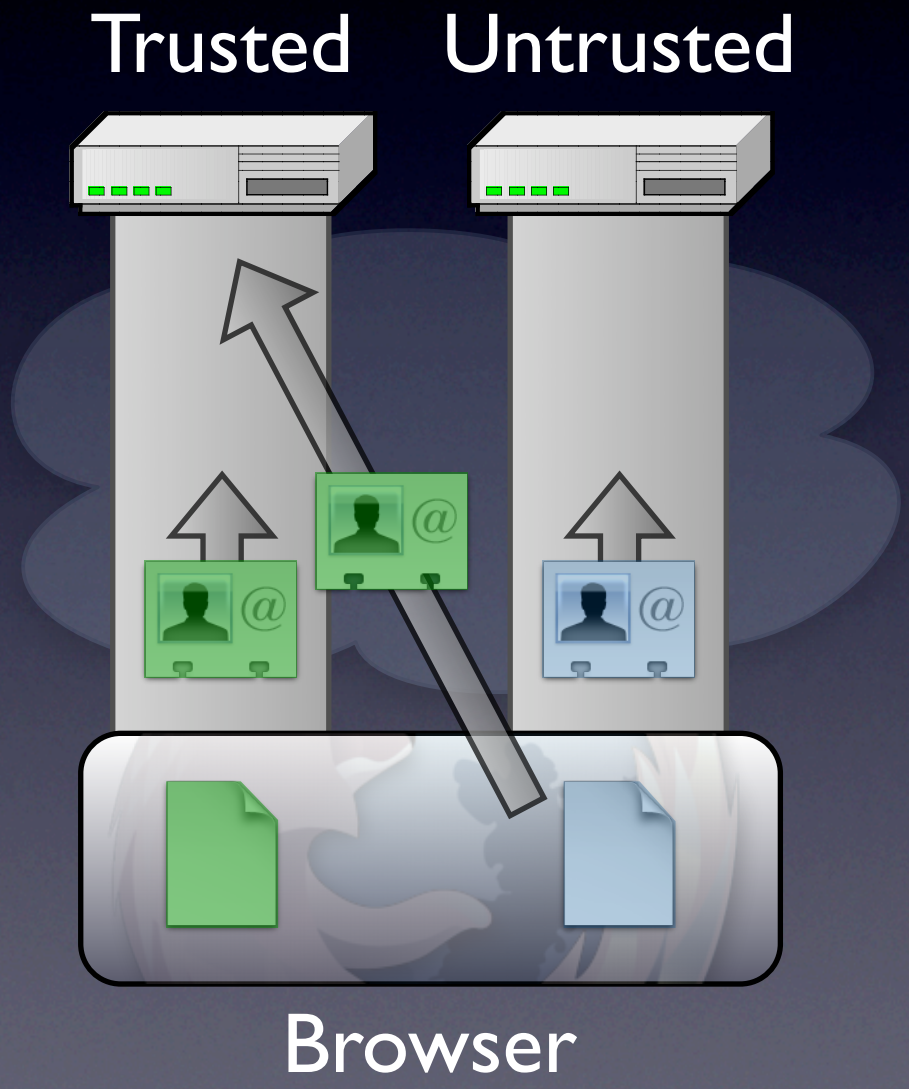
- Can abuse user credentials
- Netflix, GMail

3. Cross-Site Request Forgery (CSRF)



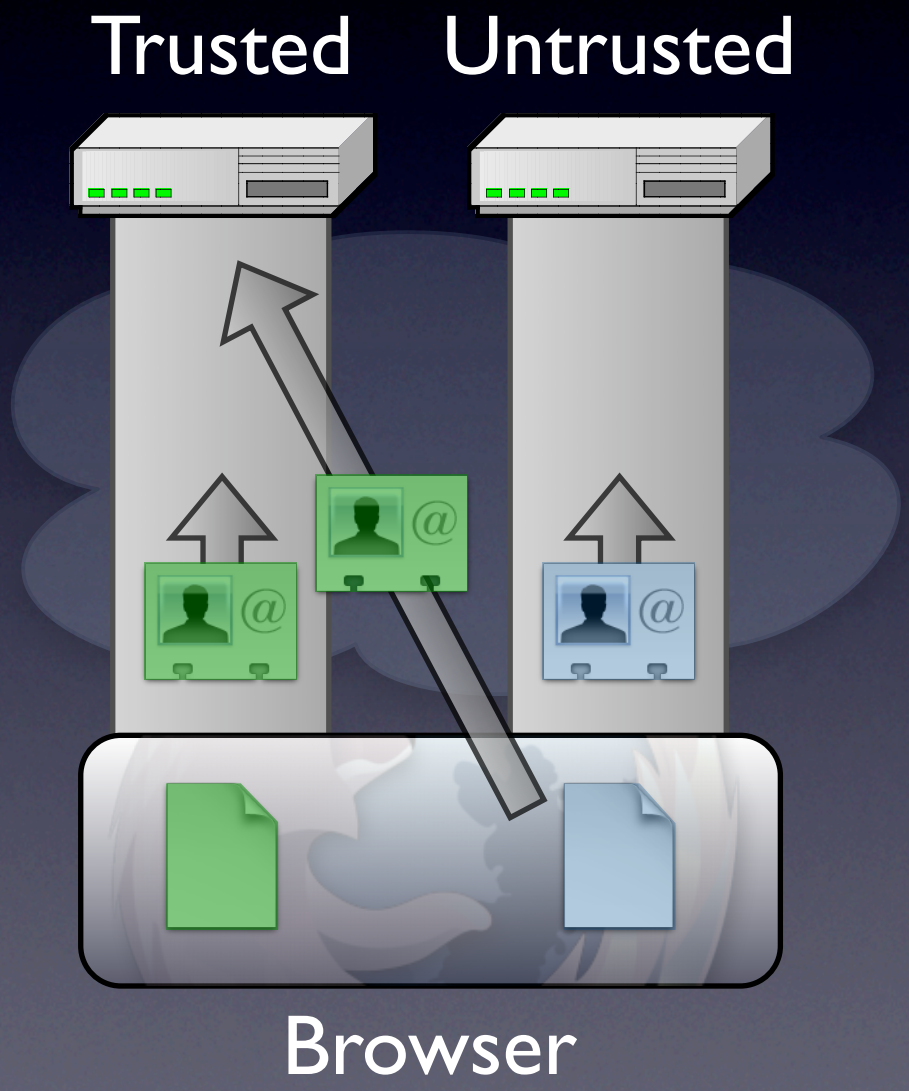
- Can abuse user credentials
  - Netflix, GMail
- Add tokens to web forms

3. Cross-Site Request Forgery (CSRF)



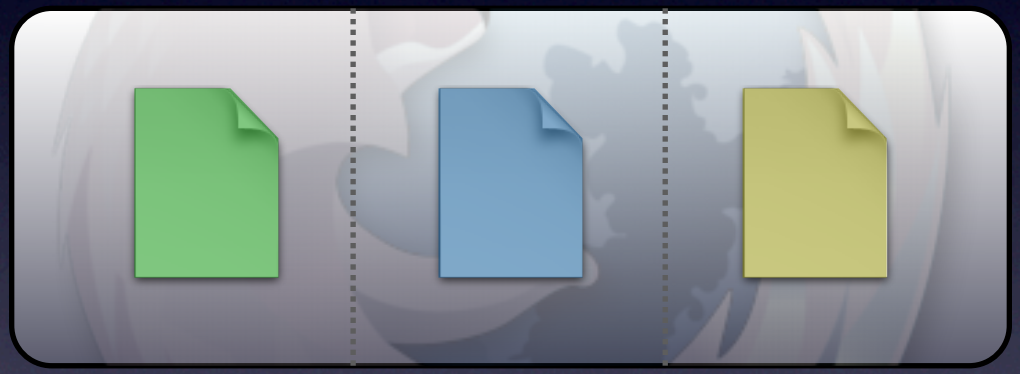
- Can abuse user credentials
  - Netflix, GMail
- Add tokens to web forms
- **Existing Proposals**: Proxies *[Jovanovic 06]*, block suspicious requests *[RequestRodeo]*

3. Cross-Site Request Forgery (CSRF)



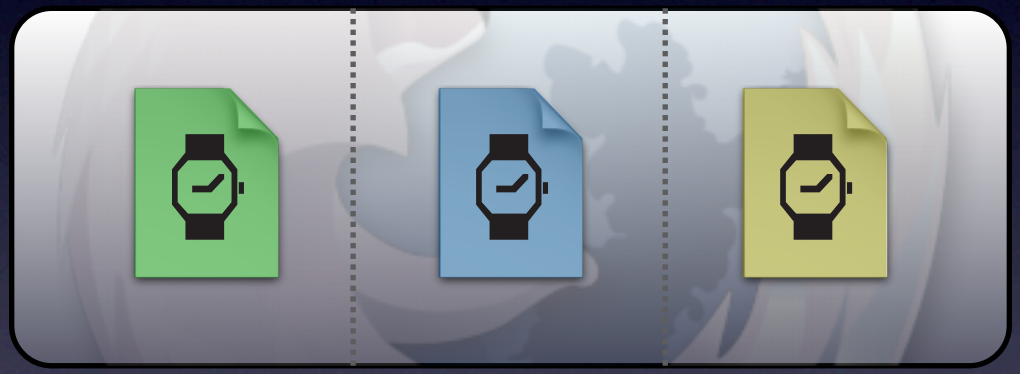
- Can abuse user credentials
  - Netflix, GMail
- Add tokens to web forms
- **Existing Proposals**: Proxies *[Jovanovic 06]*, block suspicious requests *[RequestRodeo]*
- **Proposal**: browser sessions

4. Resource Contention



Browser

4. Resource Contention



Browser

4. Resource Contention

Browser

- Unresponsiveness, crashes

4. Resource Contention

Browser

- Unresponsiveness, crashes
- **Existing Proposals**:
  Separate VMs with manifests
  [Tahoma]

4. Resource Contention

Browser

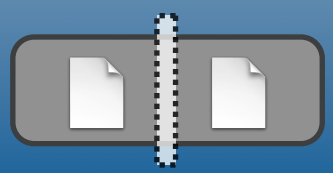
- Unresponsiveness, crashes
- **Existing Proposals**: Separate VMs with manifests *[Tahoma]*
- **Proposal**: OS processes

Outline

Motivation

Threats

Isolation

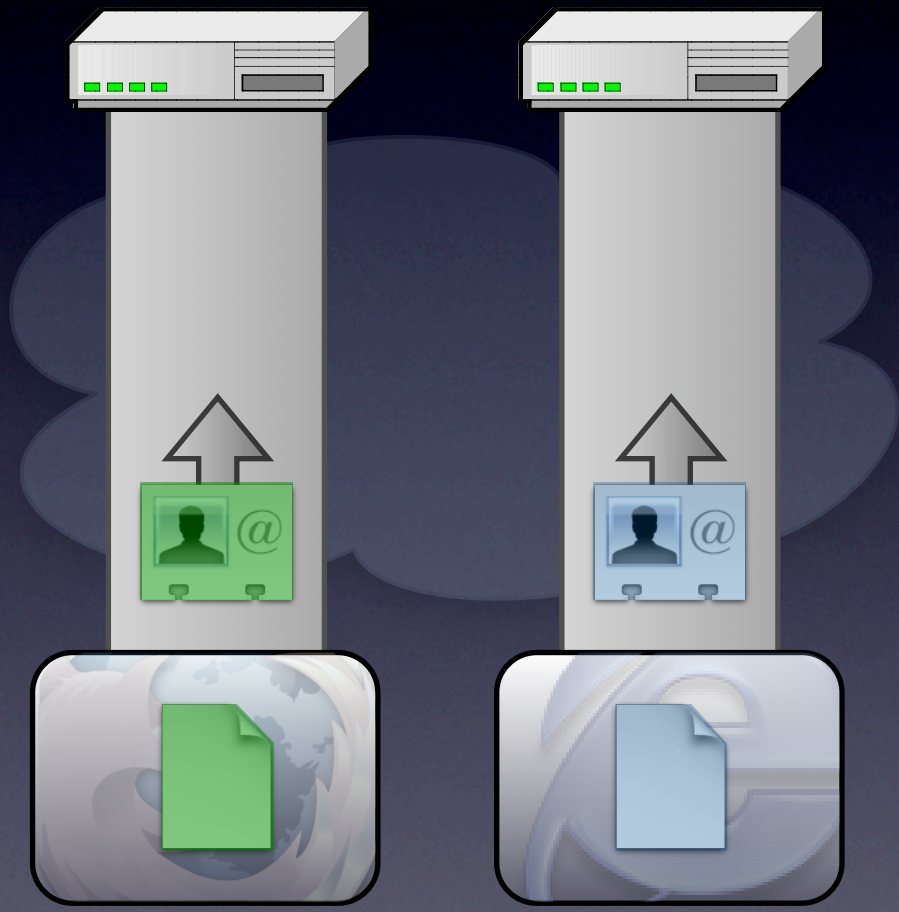


Interposition

Future Work

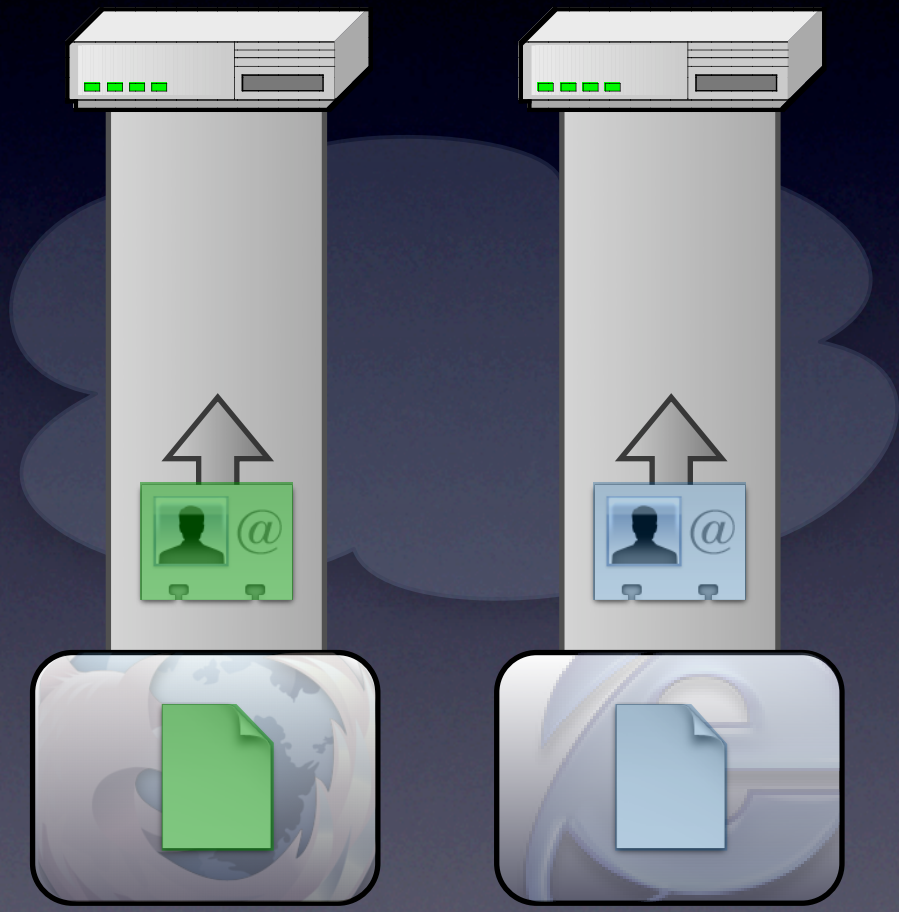
Conclusion

Isolating Web Content



Isolating Web Content

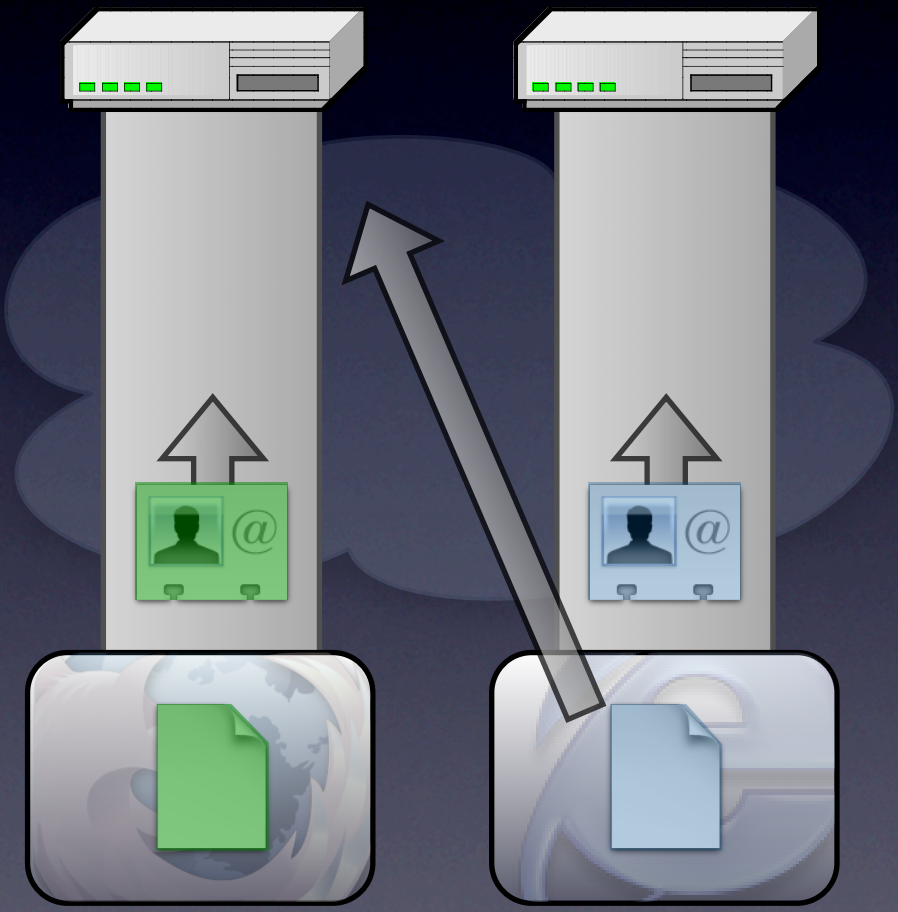
- Some threats addressed by using multiple browsers



Isolating Web Content

- Some threats addressed by using multiple browsers

3. *CSRF*

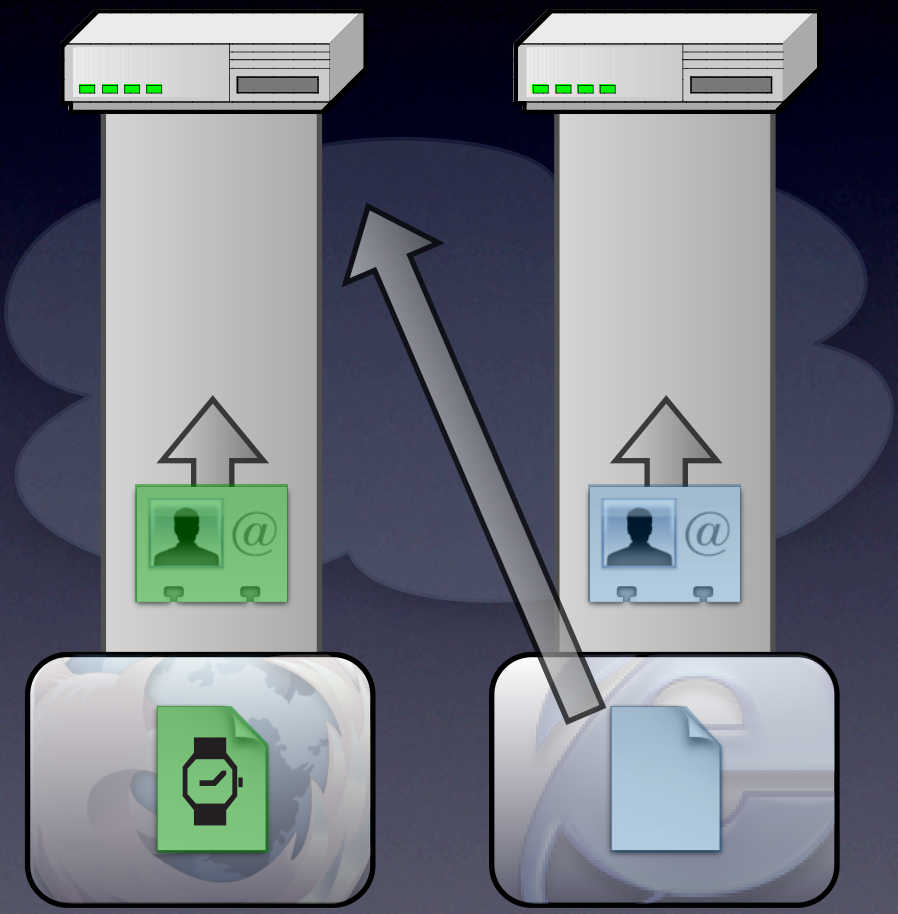


Isolating Web Content

- Some threats addressed by using multiple browsers

3. *CSRF*

4. *Resource contention*



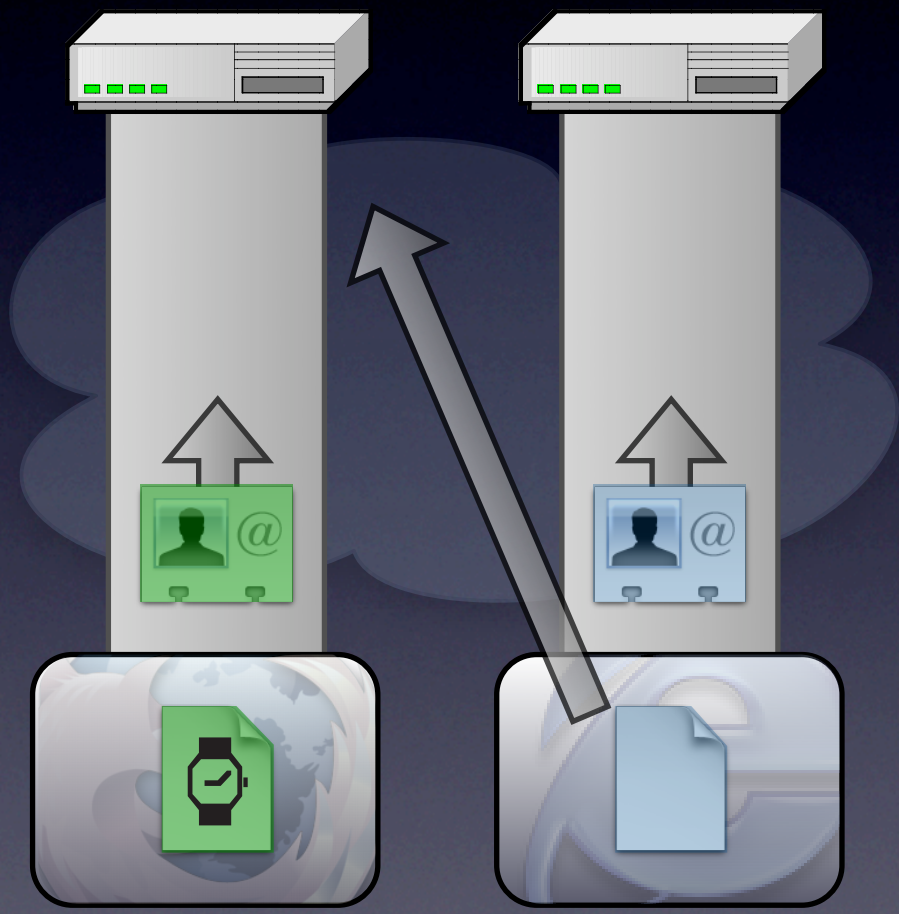
Isolating Web Content

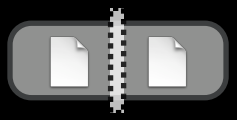
- Some threats addressed by using multiple browsers

3. *CSRF*

4. *Resource contention*

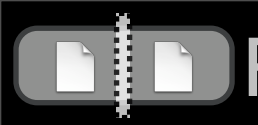
Goal: capture this idea within a single browser





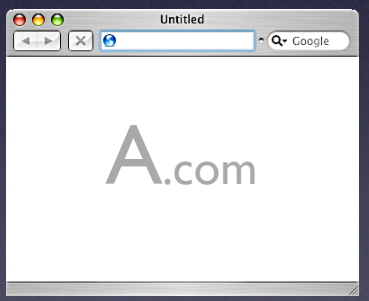
Isolation in OS

- Processes
- Lightweight fault domains
  [Wahbe 93, Swift 03]
- Covert channels
  [Lampson 73]



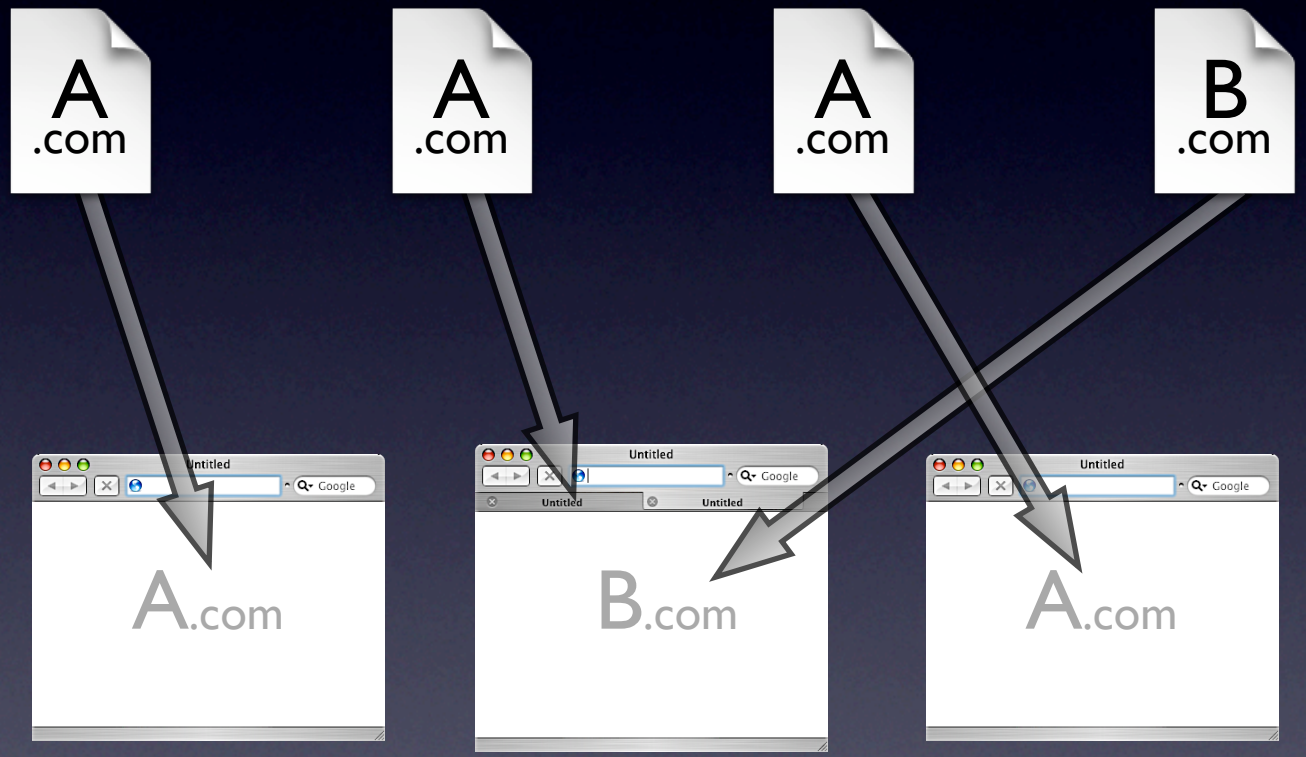
Proposal:

Decompose Browser



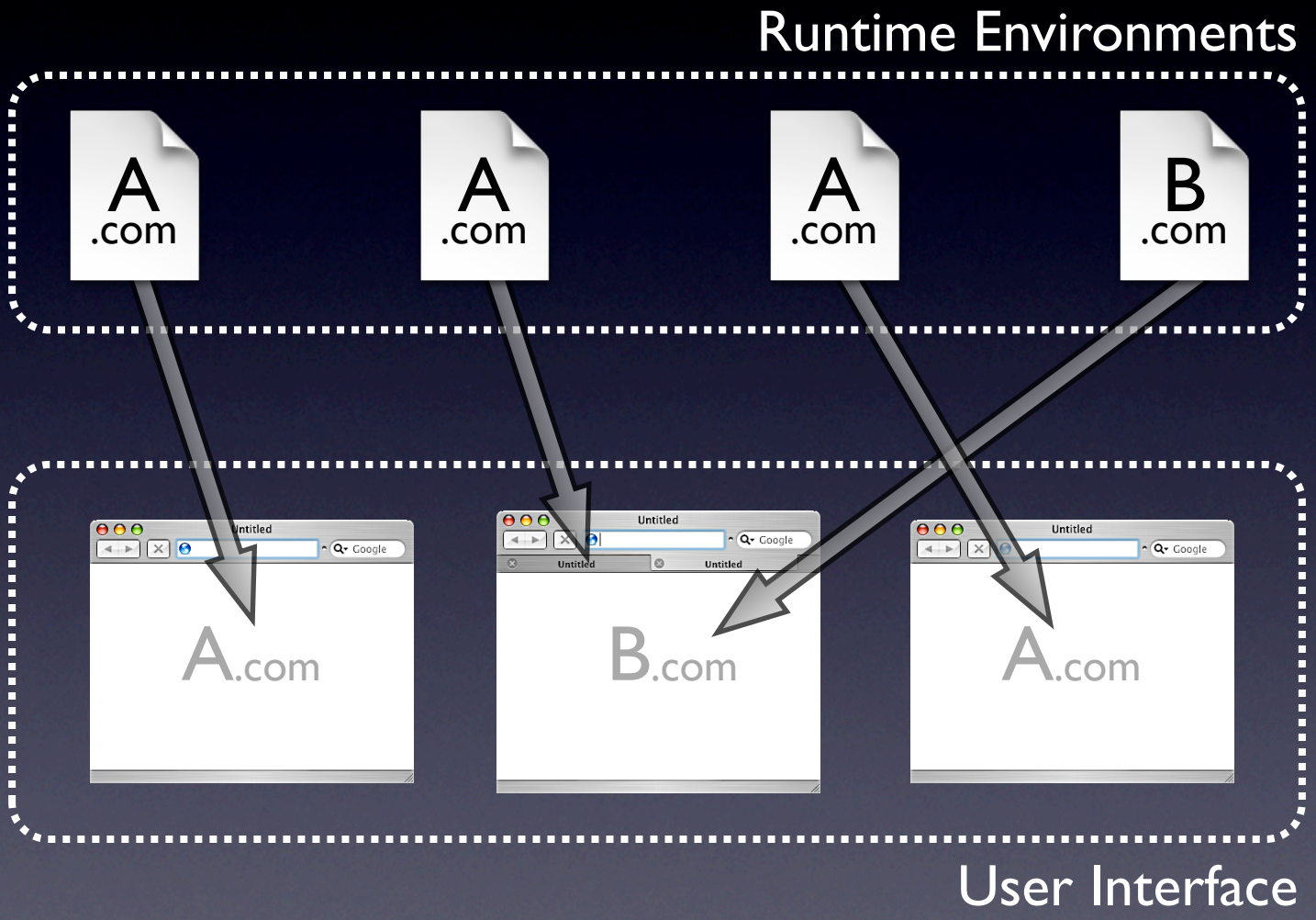
Proposal:

Decompose Browser



Proposal:

Decompose Browser



Proposal:

Host

Host

- All documents from a hostname

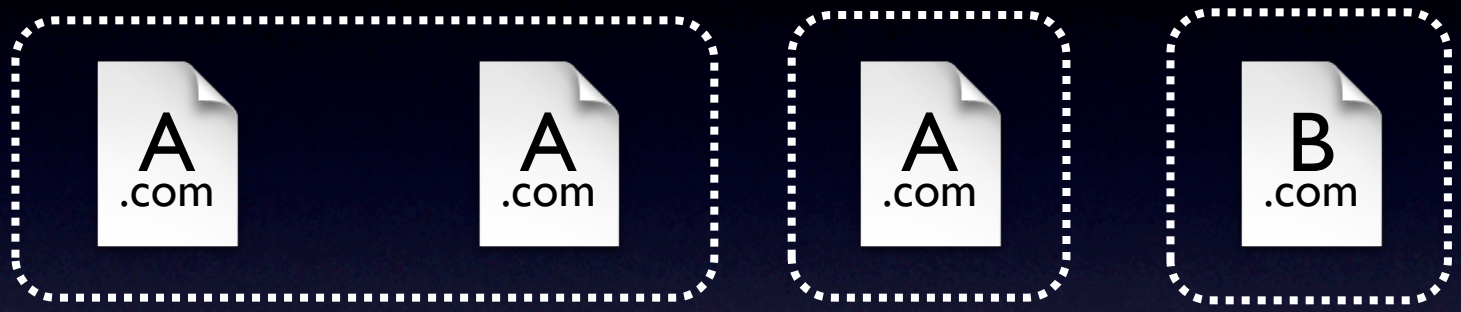
Host

- All documents from a hostname
- **Preliminary work:** process per host

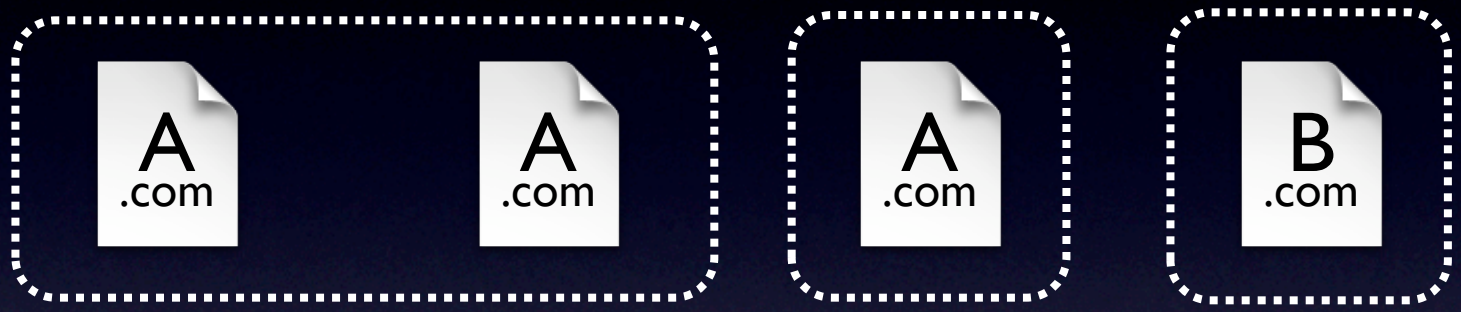
Host

- All documents from a hostname
- **Preliminary work:** process per host
- **Proposal:** partition of storage
  - Cache *[Felten 00]*, visited links *[Jackson 06]*, persistent cookies

Session

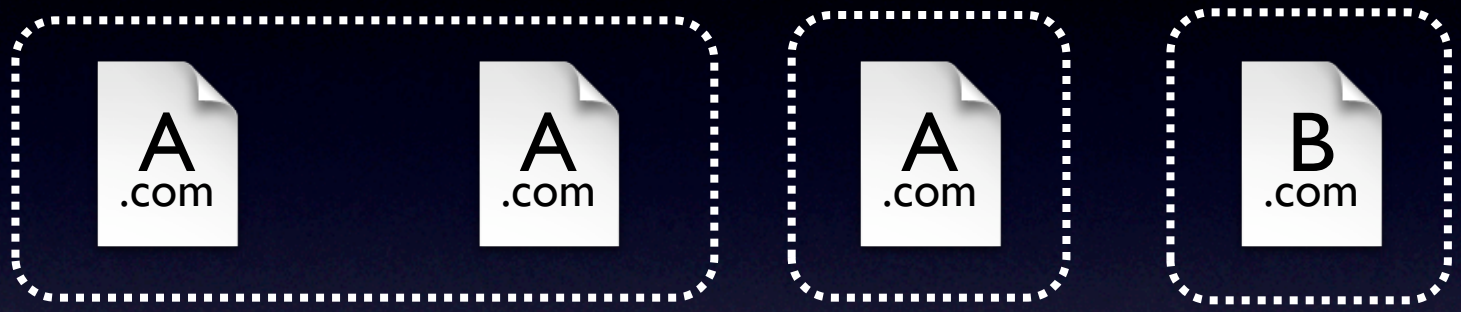


Session



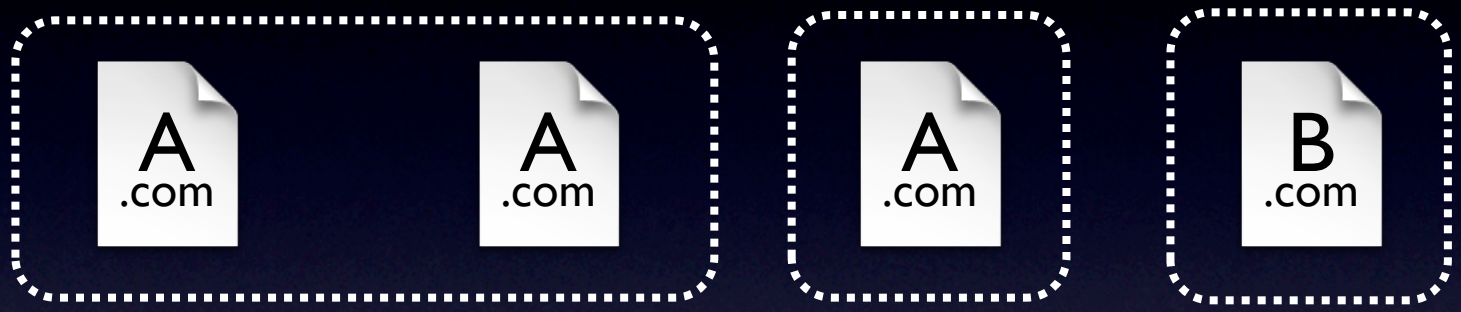
- All documents from a host with either:
  - Navigational relationship
  - Parent-child relationship

Session



- All documents from a host with either:
  - Navigational relationship
  - Parent-child relationship
- Separate **session cookies** *(no exp. date)*

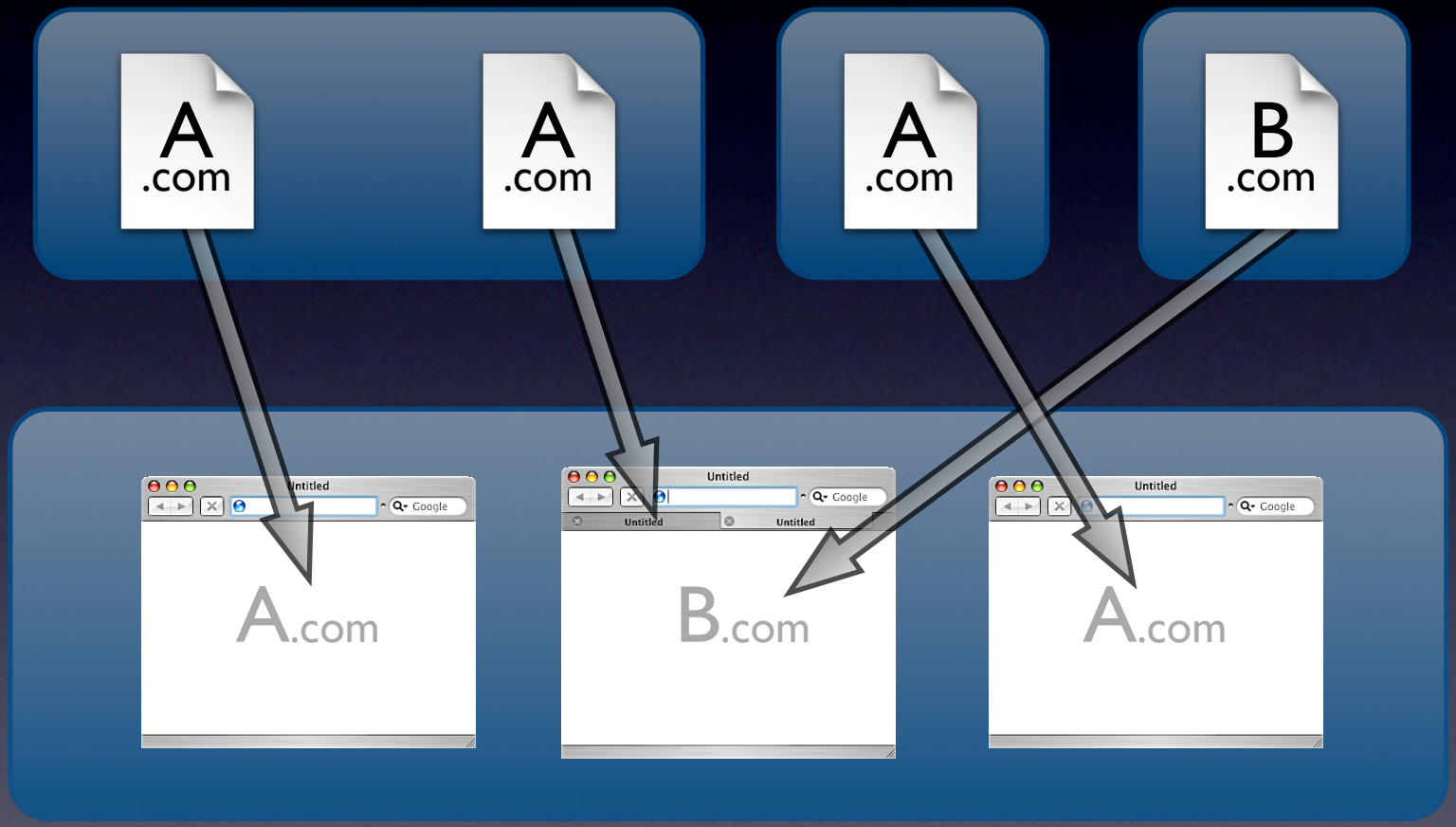
Session



- All documents from a host with either:
  - Navigational relationship
  - Parent-child relationship
- Separate **session cookies** *(no exp. date)*
- Separate runtime environments

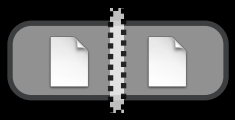
Proposal:

Isolation with Processes



Methodology

- Implement with KDE web browser
- Challenges:
  - What to isolate or share across sessions?
  - How to keep overhead low?
  - Will use of cookies need to change?



Proposal:

Evaluation

Evaluation

- **Safety**: test CSRF and contention attacks
  - Use both crafted and observed pages

Evaluation

- **Safety**: test CSRF and contention attacks
  - Use both crafted and observed pages
- **Back Compat**: test popular content
  - Compare loaded objects, JS errors
  - Characterize use of cookies

Evaluation

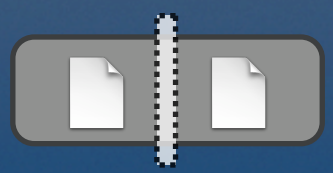
- **Safety**: test CSRF and contention attacks
  - Use both crafted and observed pages
- **Back Compat**: test popular content
  - Compare loaded objects, JS errors
  - Characterize use of cookies
- **Efficiency**: overhead of sessions, calls
  - Are lightweight domains needed?

Outline

Motivation

Threats

Isolation



Interposition

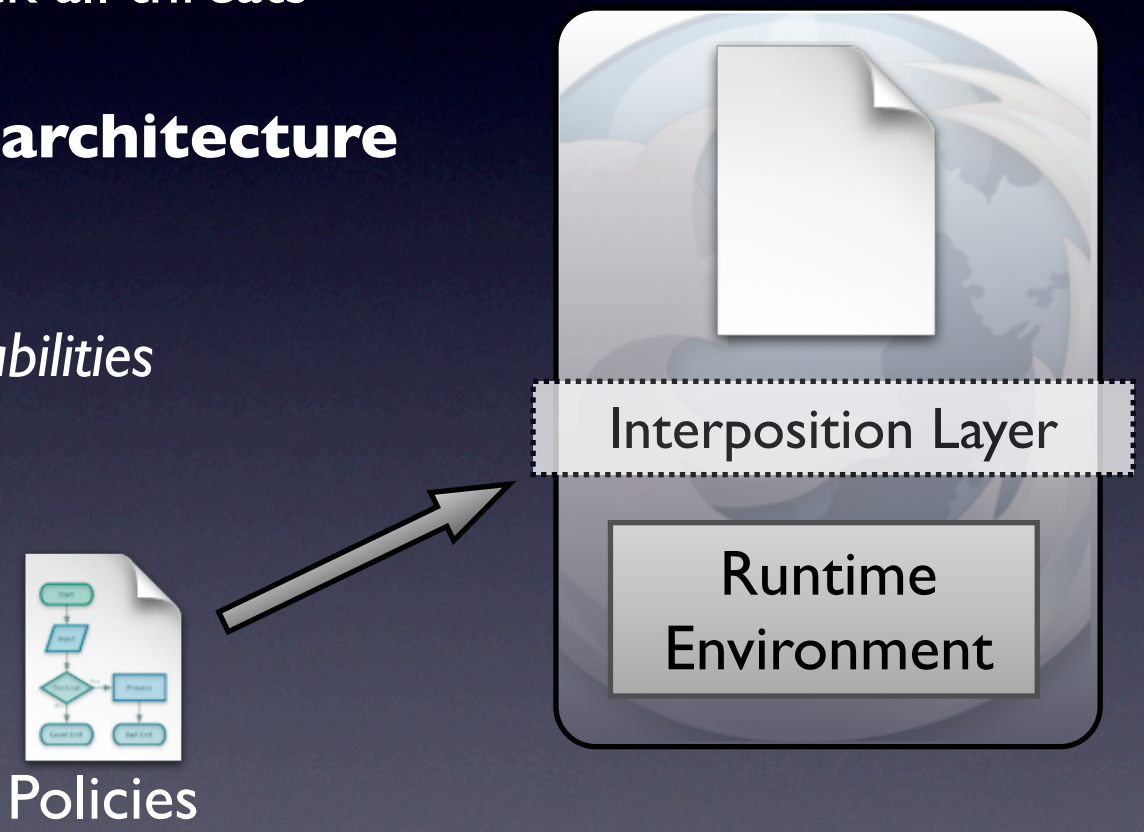
Future Work

Conclusion

Interposition & Policies

- Fixed policy doesn't block all threats
- **Extensible security architecture**
  [Wallach 97]

  1. *Defend browser vulnerabilities*
  2. *Block XSS attacks*

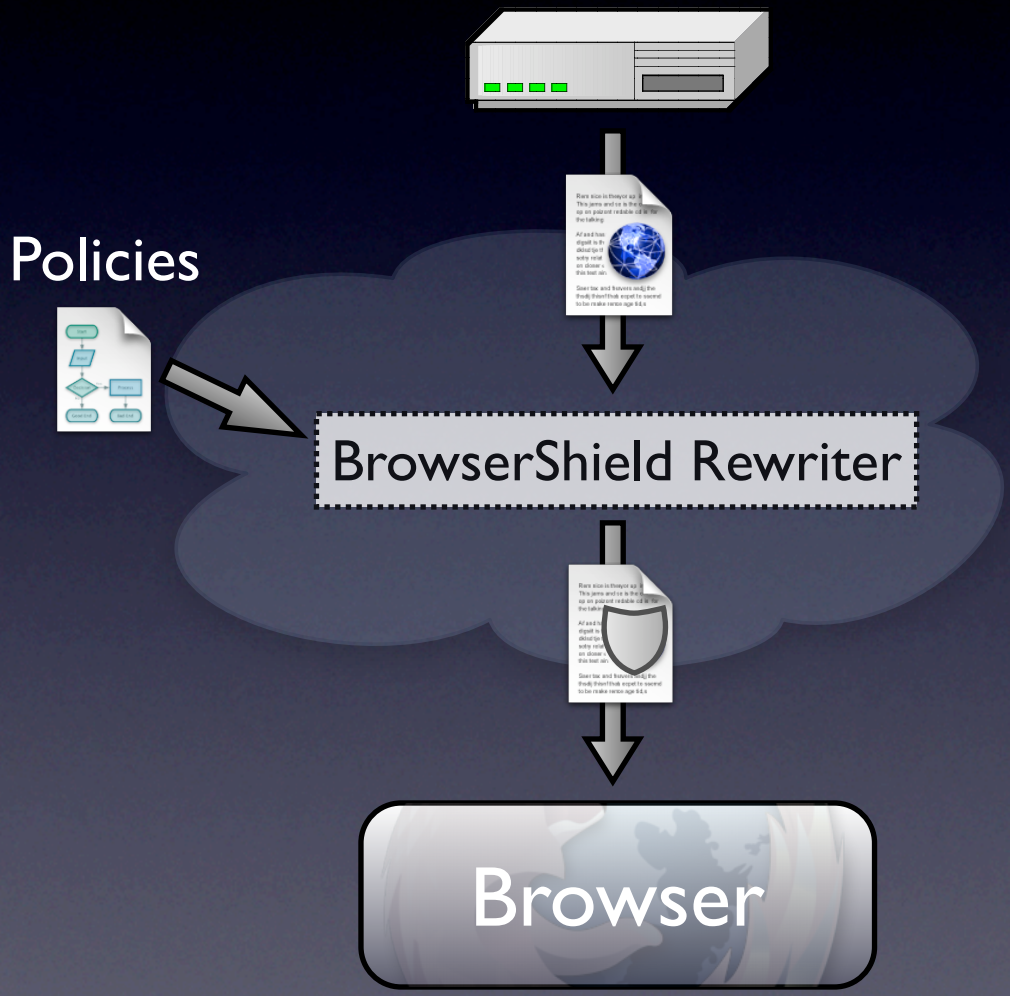


Interposition in OS

- System call interposition
  [Goldberg 96, Garfinkel 04]
- Code rewriting
  [Erlingsson 00]
- Vulnerability Filtering
  [Wang 04]

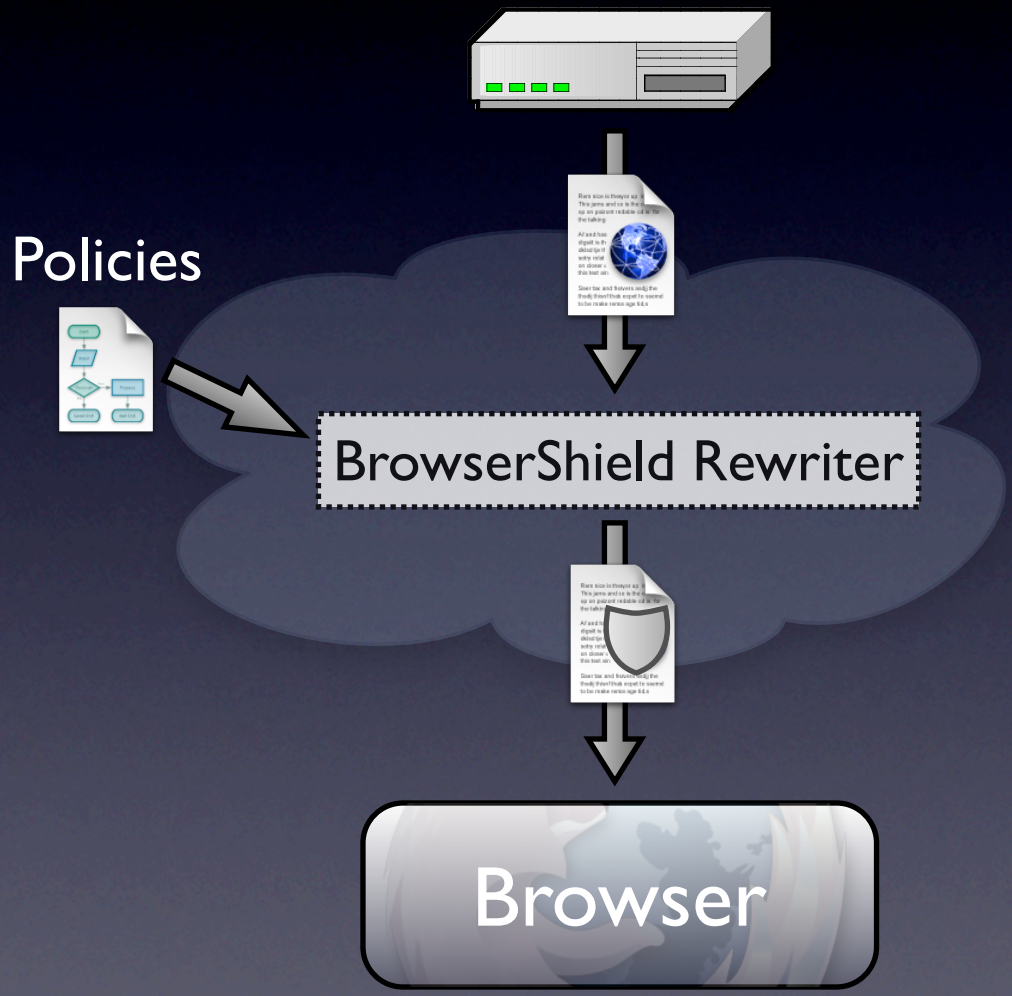
Preliminary Work:

BrowserShield



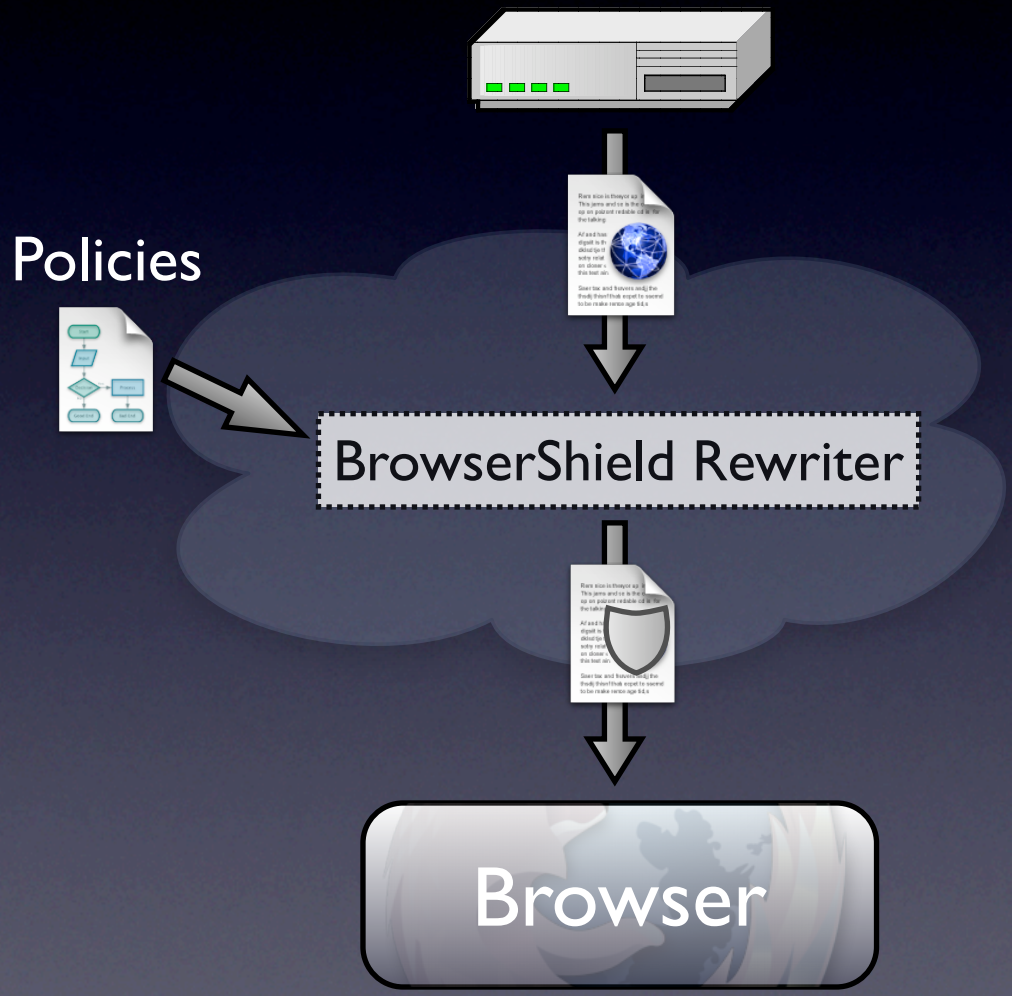
Preliminary Work:

BrowserShield



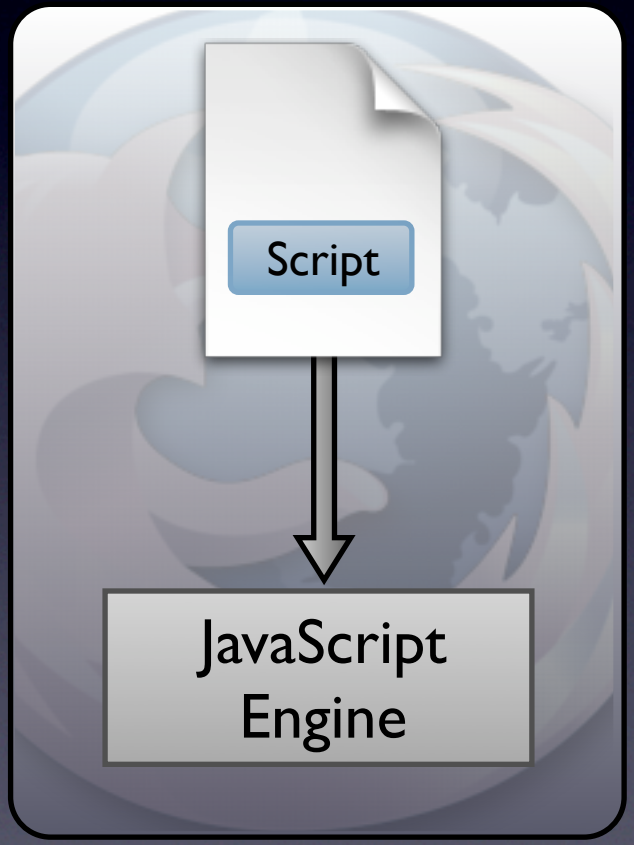
- **Code Rewriting**
  - Interpose on HTML and JavaScript code

BrowserShield



- **Code Rewriting**
  - Interpose on HTML and JavaScript code
- **Vulnerability Policies**
  - Block all exploits of known vulnerabilities

Script Whitelists



Script Whitelists

- **Intercept JavaScript**
  - Independent of quirky HTML parsing

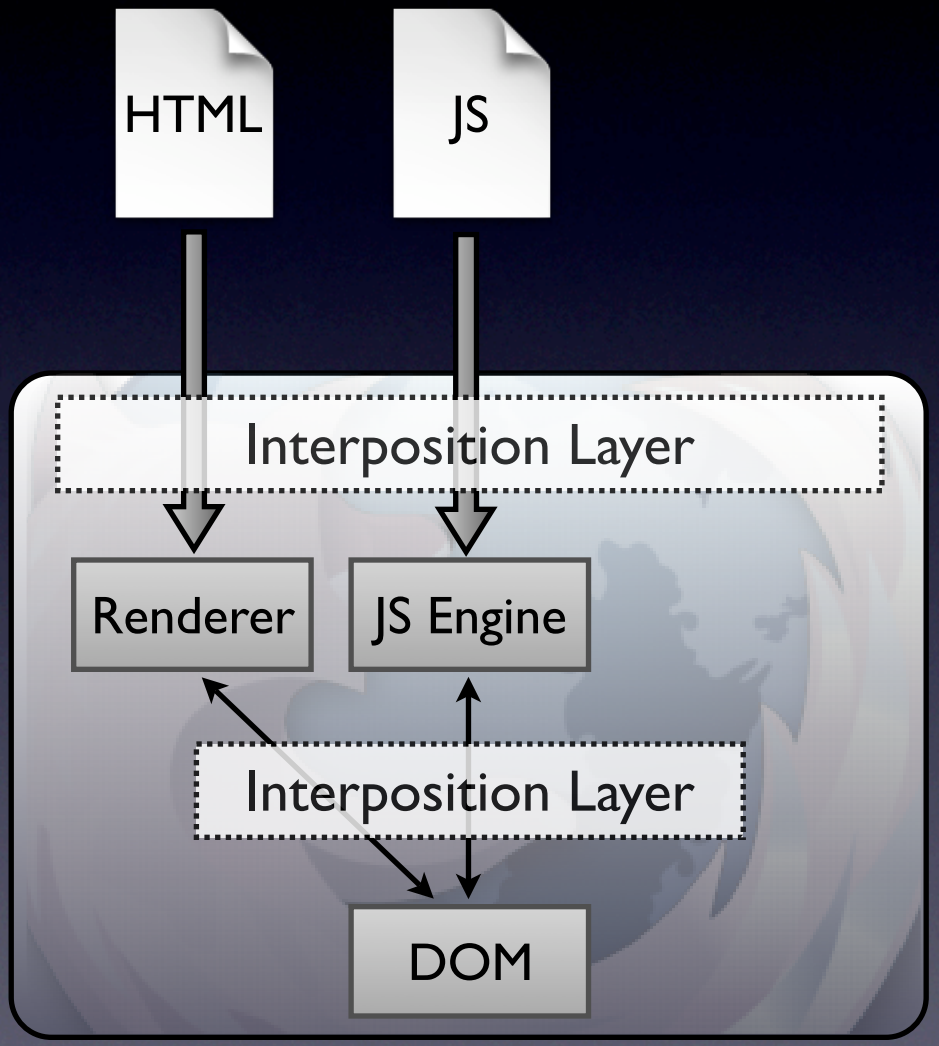
Preliminary Work:

Script Whitelists

- **Intercept JavaScript**
  - Independent of quirky HTML parsing
- **Page provides whitelist**
  - Prevents XSS attacks
  - Practical to deploy

Proposal:

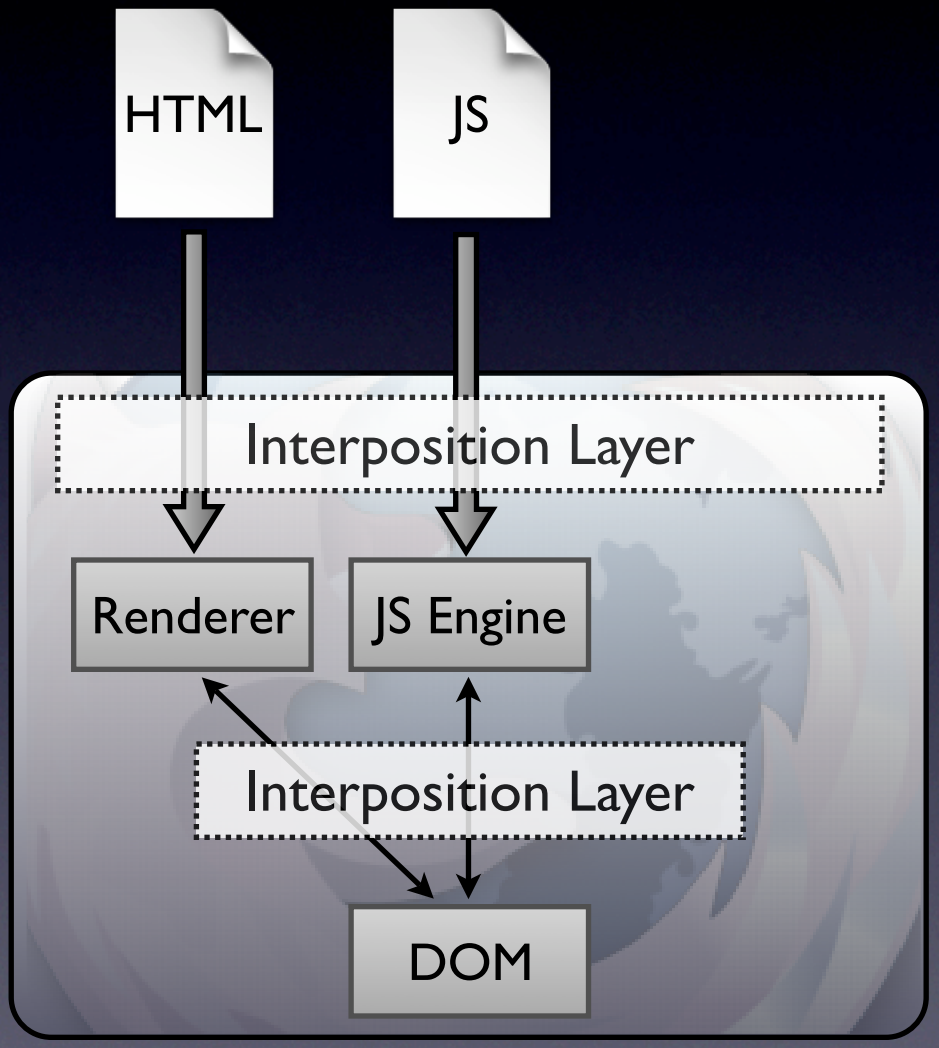
Interposition Layer



Proposal:

Interposition Layer

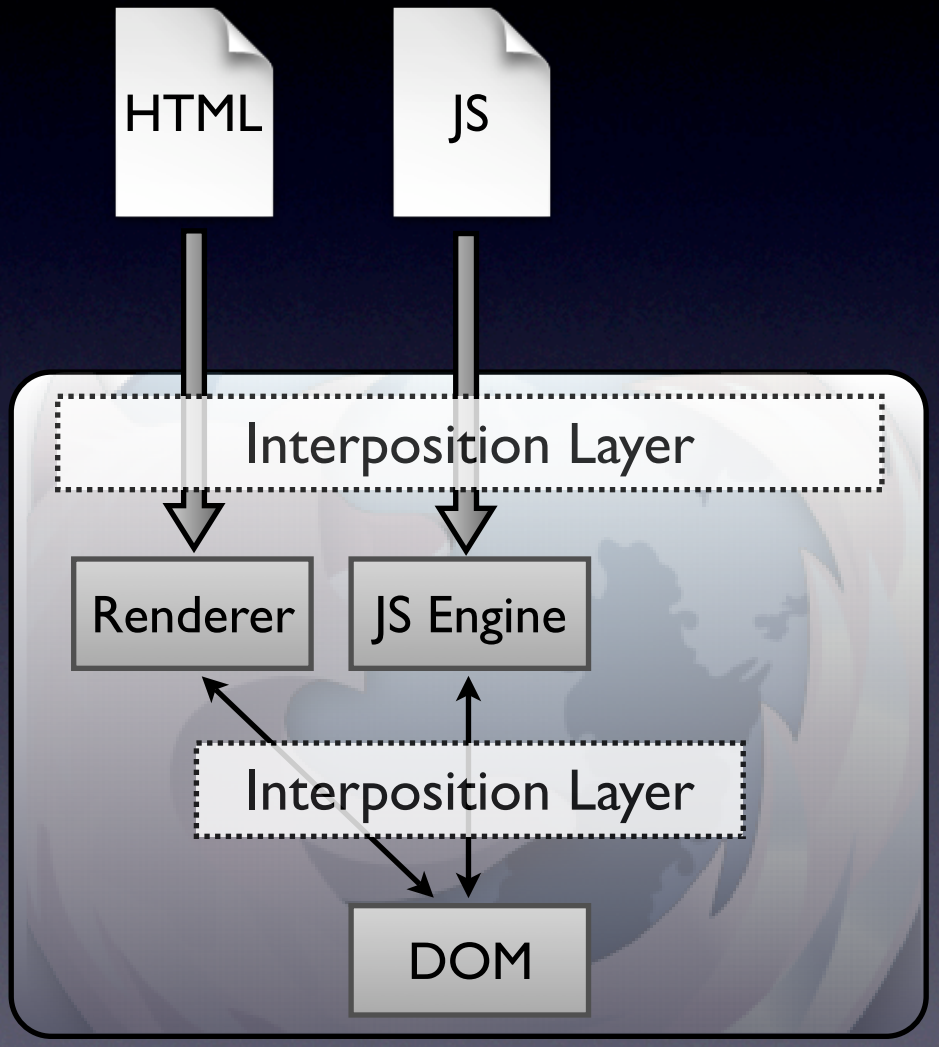
- Interpose *within* browser



Proposal:

Interposition Layer

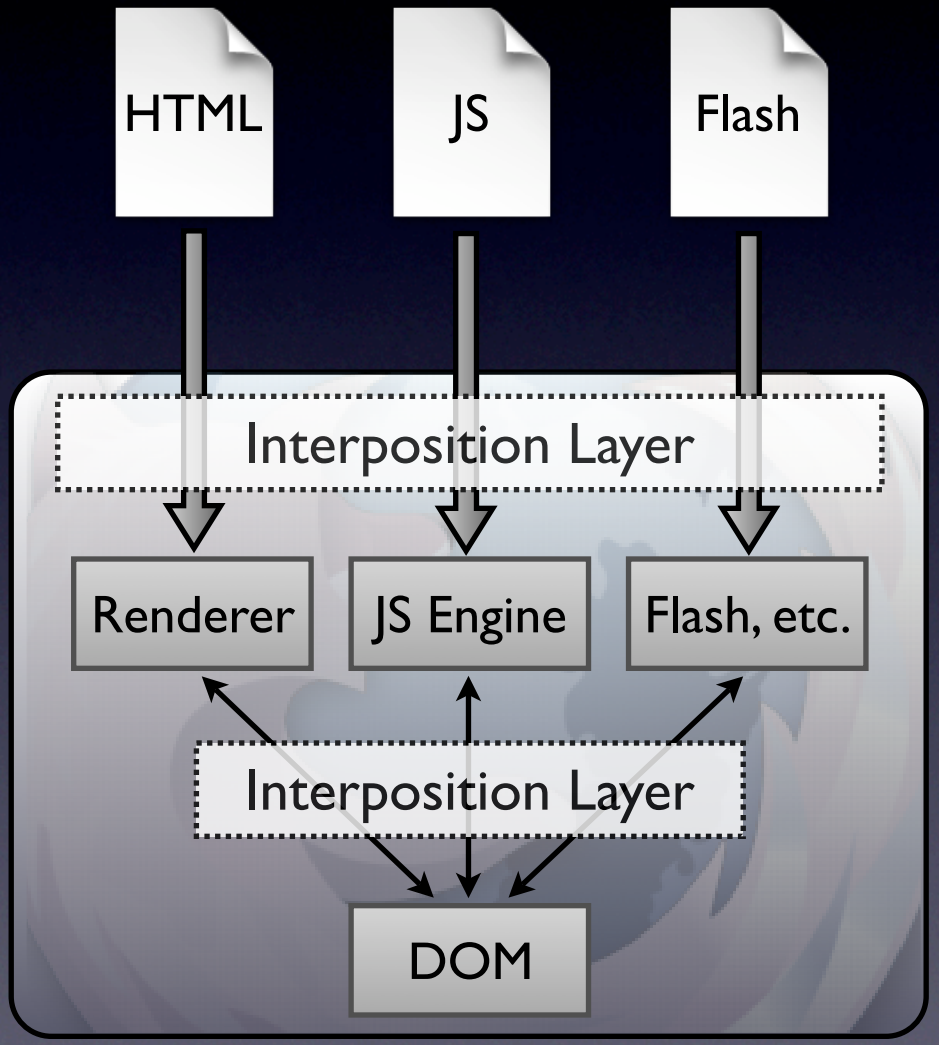
- Interpose *within* browser
- Uniform policies for web content of all formats



Proposal:

Interposition Layer

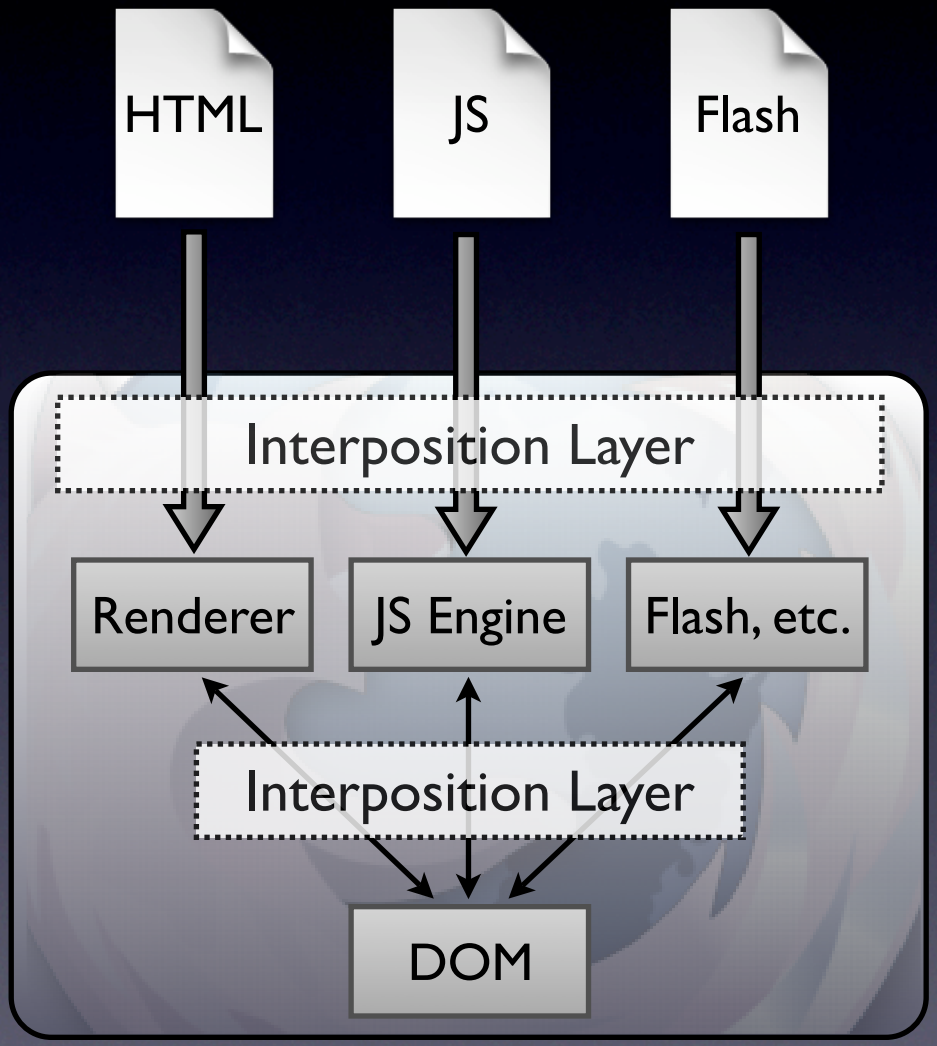
- Interpose *within* browser
- Uniform policies for web content of all formats



Proposal:

Interposition Layer

- Interpose *within* browser
- Uniform policies for web content of all formats
- Expose *hooks* for policies:
  - Raw input (e.g., HTML), DOM access, Communication



Proposal:

Security Policies

- **Policy hierarchy**
  - Browser, extensions, sessions, documents
- **Build sample policies**
  - Vulnerability shields
  - Script whitelists

Proposal:

Methodology

- Implement in Firefox (extension API)
- Challenges:
  - How to specify policies?
  - Expressive with low overhead?
  - How much can plugins be confined?

Proposal:

Evaluation

Evaluation

- **Safety**: can it support safer policies?
  - Test defense against exploits, XSS

Evaluation

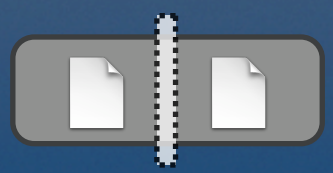
- **Safety**: can it support safer policies?
  - Test defense against exploits, XSS
- **Back Compat**: do policies break pages?
  - Same popular content tests

Evaluation

- **Safety**: can it support safer policies?
  - Test defense against exploits, XSS
- **Back Compat**: do policies break pages?
  - Same popular content tests
- **Efficiency**: overhead of layer and policies?
  - Micro/macro benchmarks

Outline

- Motivation
- Threats
- Isolation



- Interposition

- **Future Work**
- Conclusion

Future Directions

Future Directions

- **Communication** between sites
  - *Better support for "mashups"*

Future Directions

- **Communication** between sites
  - *Better support for "mashups"*
- Support for new **phishing defenses**
  - *Visual indicators of sessions*

Future Directions

- **Communication** between sites
  - *Better support for "mashups"*
- Support for new **phishing defenses**
  - *Visual indicators of sessions*
- Platform for deploying **security research**
  - *Distribute as policies*

Conclusion

- OS mechanisms can improve web browsers
  - **Isolation** prevents interference
  - **Interposition** allows flexible policies
- Will prevent threats with few costs