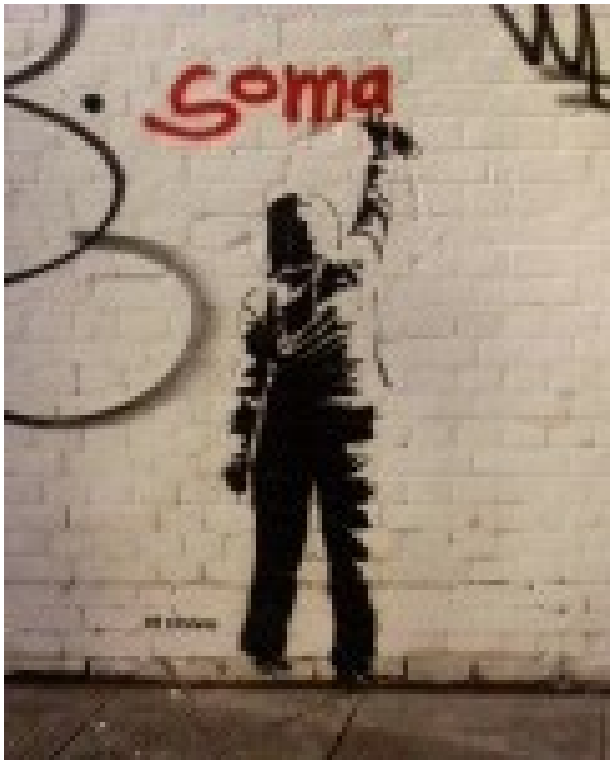
Carleton
UNIVERSITY

SOMA:
Mutual Approval for Included Content On Web Pages

Terri Oda, **Glenn Wurster**,
P. C. van Oorschot, Anil Somayaji

SOMA

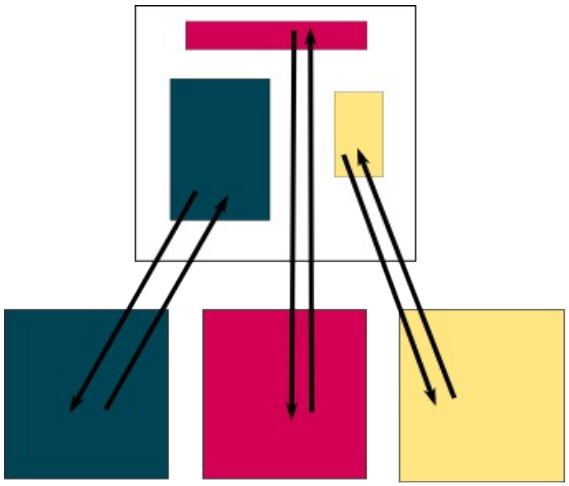


- Same Origin Mutual Approval
- Tighten the JavaScript Same Origin policy to prevent additional attacks
- Extension to web browsers
  - Obey simple policies set by site operators

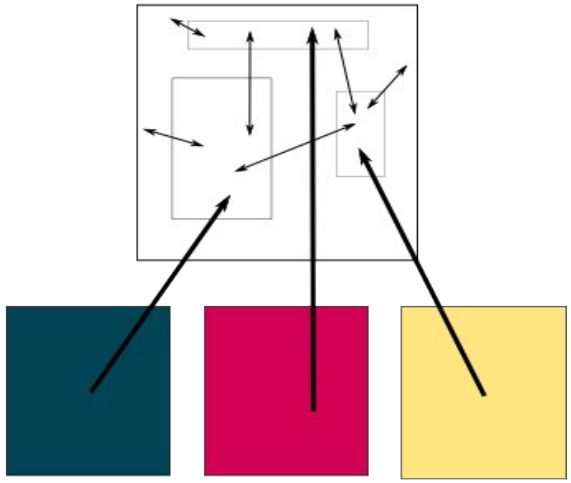
http://flickr.com/photos/kenturamon/168978666/

Same Origin Policy

- All JavaScript code has full access to:
  - Run/Overwrite all other JavaScript code
  - Read/Write to other content from the **document** origin
- Same Origin Policy restricts access to content from other domains



Incorrect View



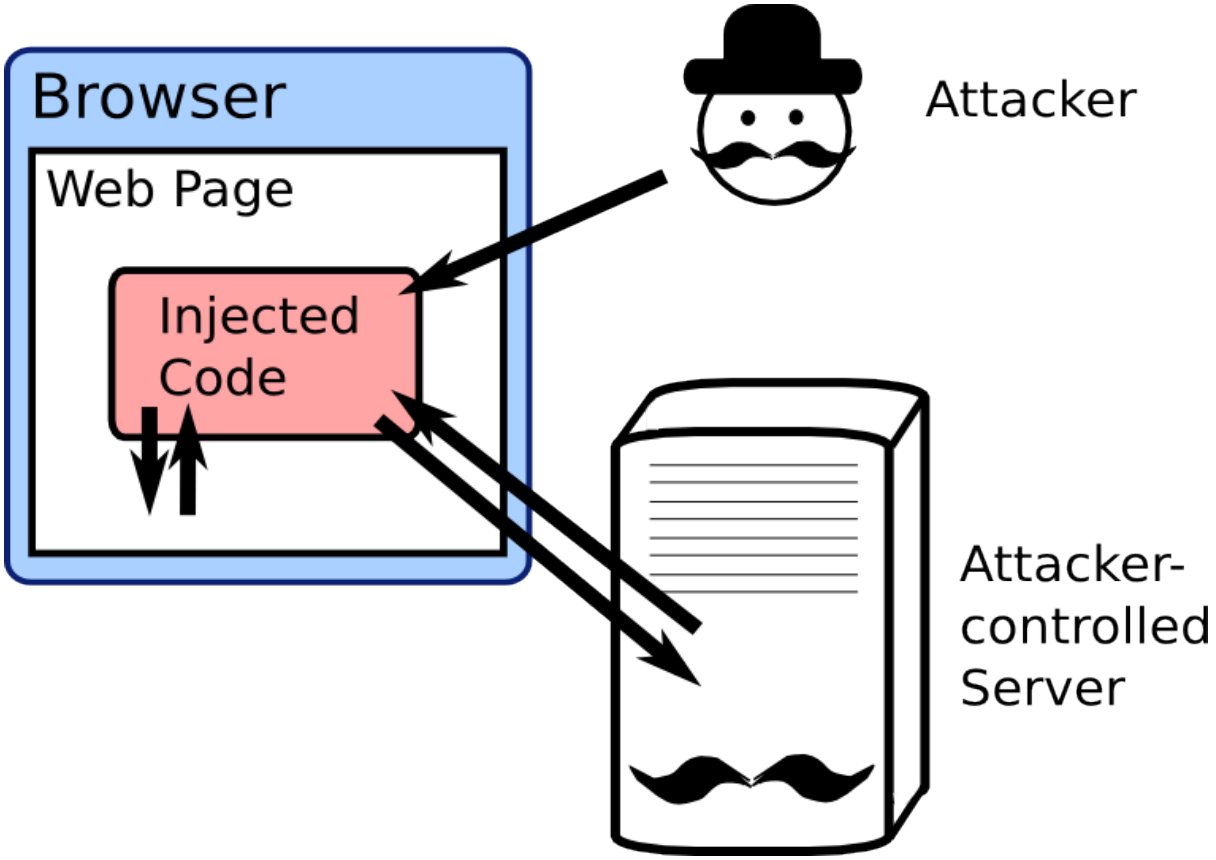
Correct View

Same Origin Policy

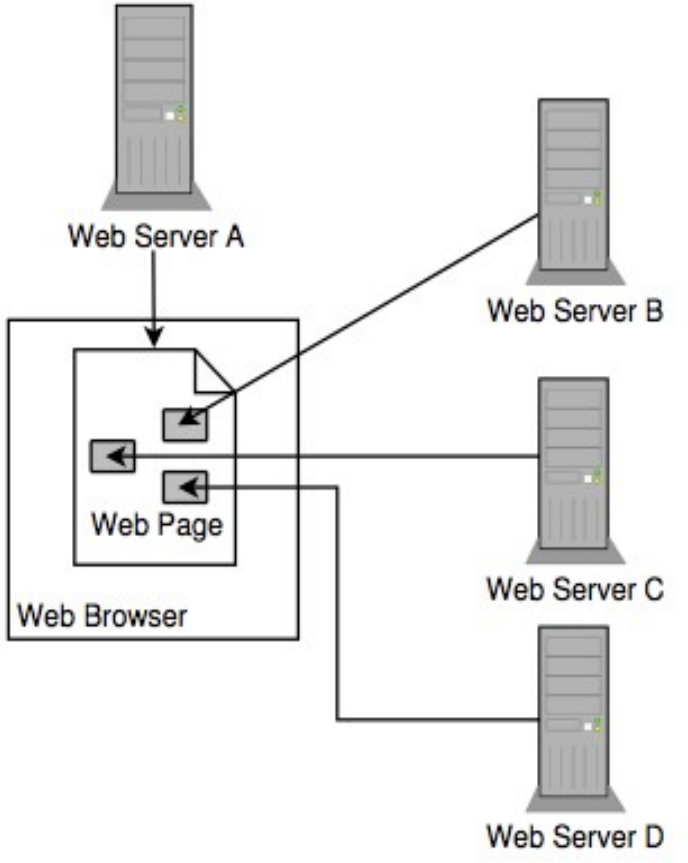
- Same Origin policy restricts read and modify access
- Fetching of content is unrestricted

Content Type	Permissions				
	Fetch	Read	Modify	Execute	Display
Images	YES	SO	SO	NO	YES
HTML	YES	SO	SO	NO	YES
JavaScript	YES	SO	YES	YES	NO
Audio/Video	YES	Plugin Dependant		NO	YES

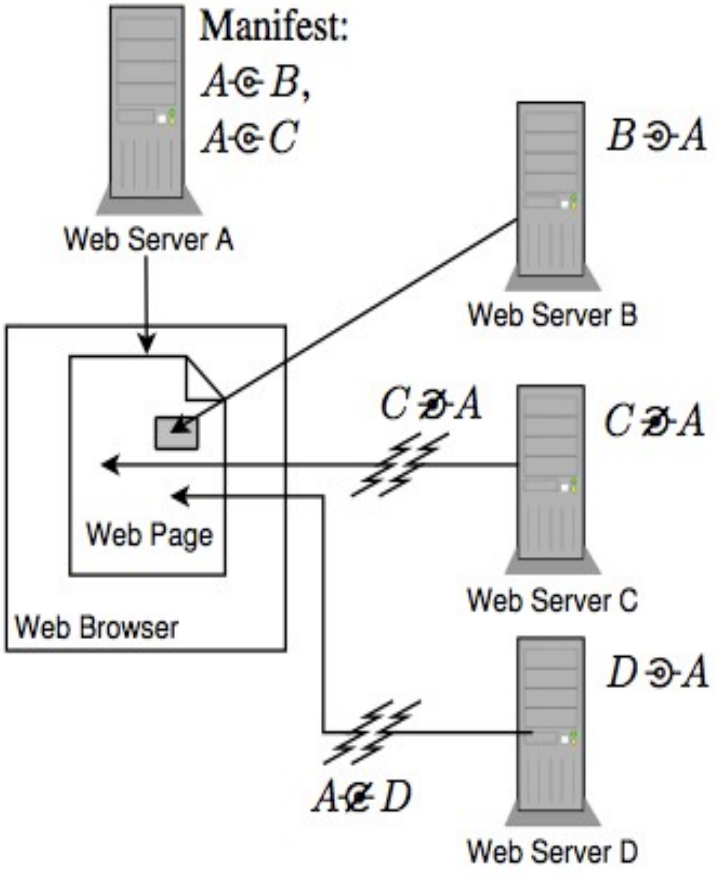
Sample Web Attack



Inclusions



Inclusions allowed with Same Origin



Inclusions allowed with SOMA

SOMA Manifests

1. A file on the origin domain (/`soma-manifest`)
2. Lists domains approved by origin site

Possible Manifest States

(given by site A)

Server Response	Meaning	Symbol
No Manifest	All sites approved	$A \subset B$
B in Manifest	Content from B allowed	$A \subset B$
B not in Manifest	Content from B not allowed	$A \not\subset B$

For some domain B

SOMA Approvals

1. Script on content provider site (`/soma-approval`)
2. Responds to approval requests
   - Based on origin page domain

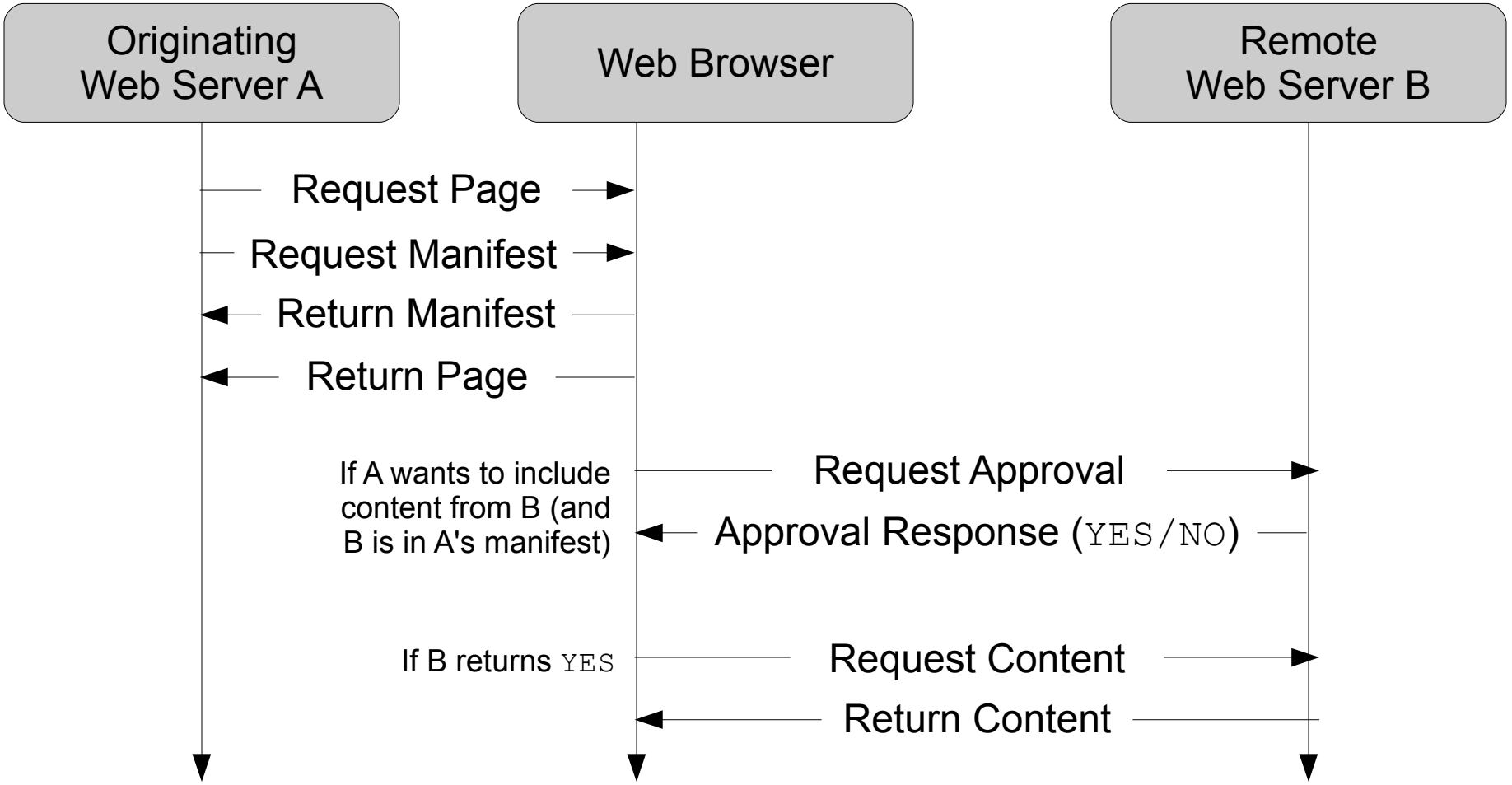
Possible Approval Responses

(by site B)

Server Response	Meaning	Symbol
File Not Found	All sites approved	B ⊸ A
`YES`	Can include content into A's page	B ⊸ A
`NO`	Can NOT include content into A's page	B ⊸̸ A

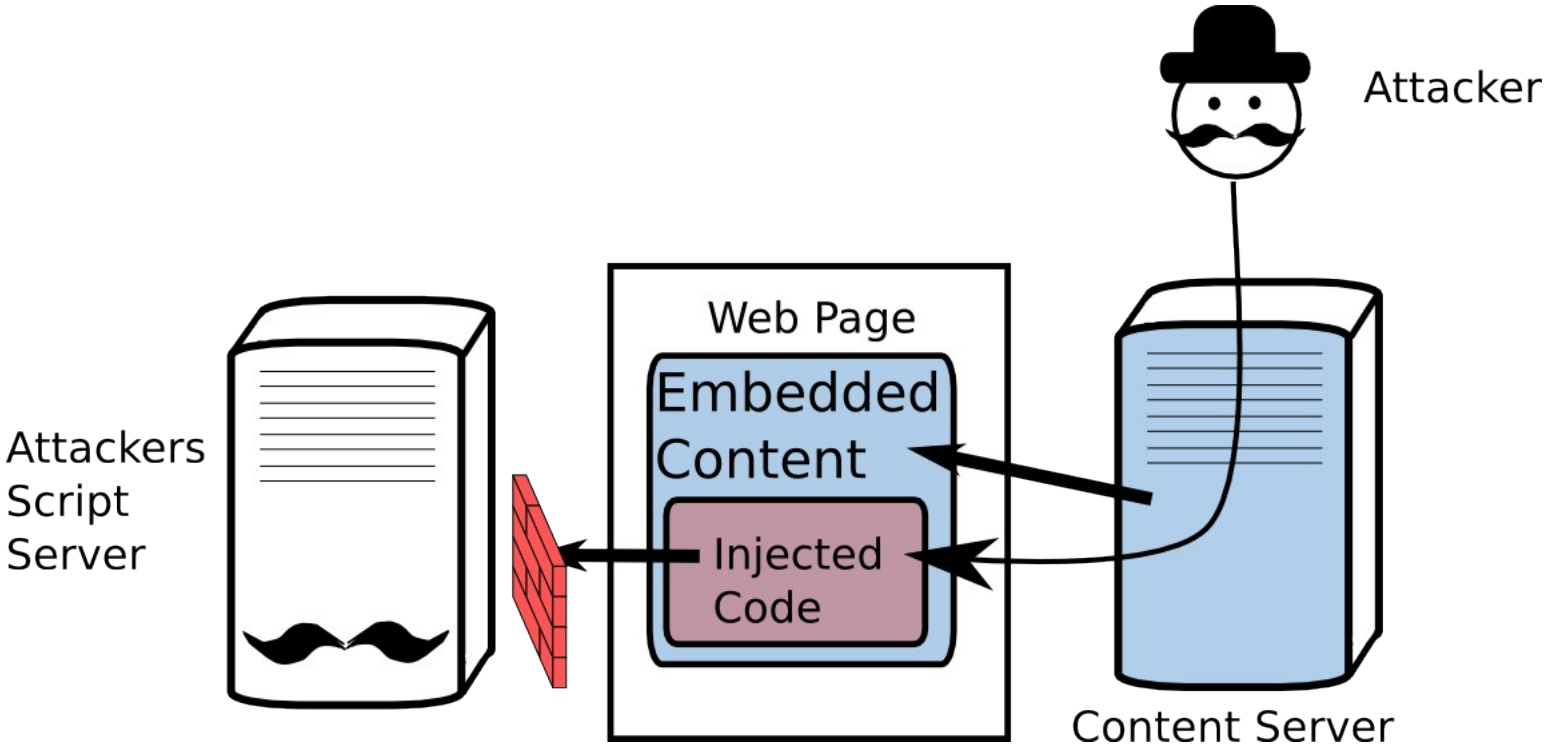
For some domain A

SOMA Message Flow



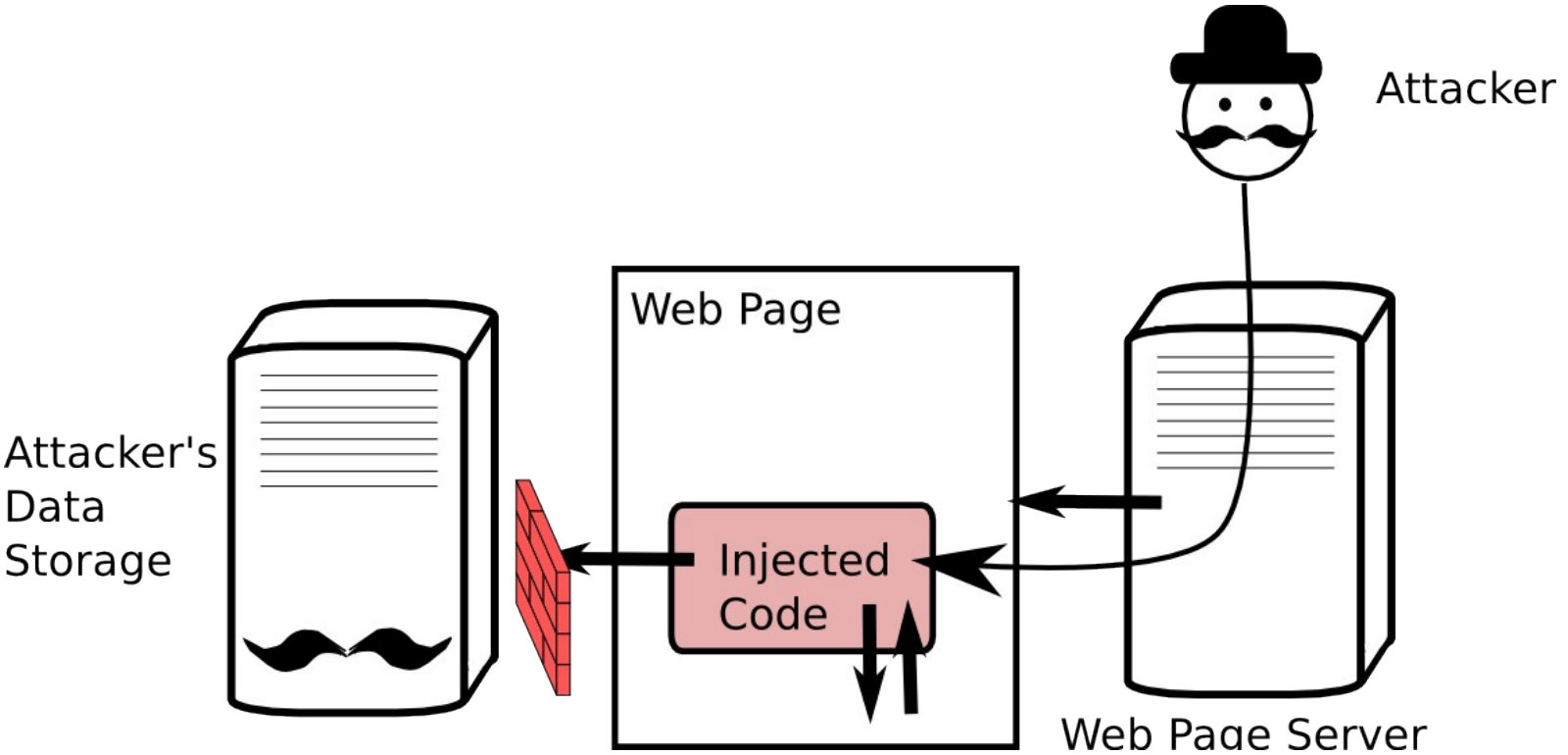
Cross Site Scripting

- Any script can include other scripts (from any site)
- Inclusion blocked by SOMA Manifest



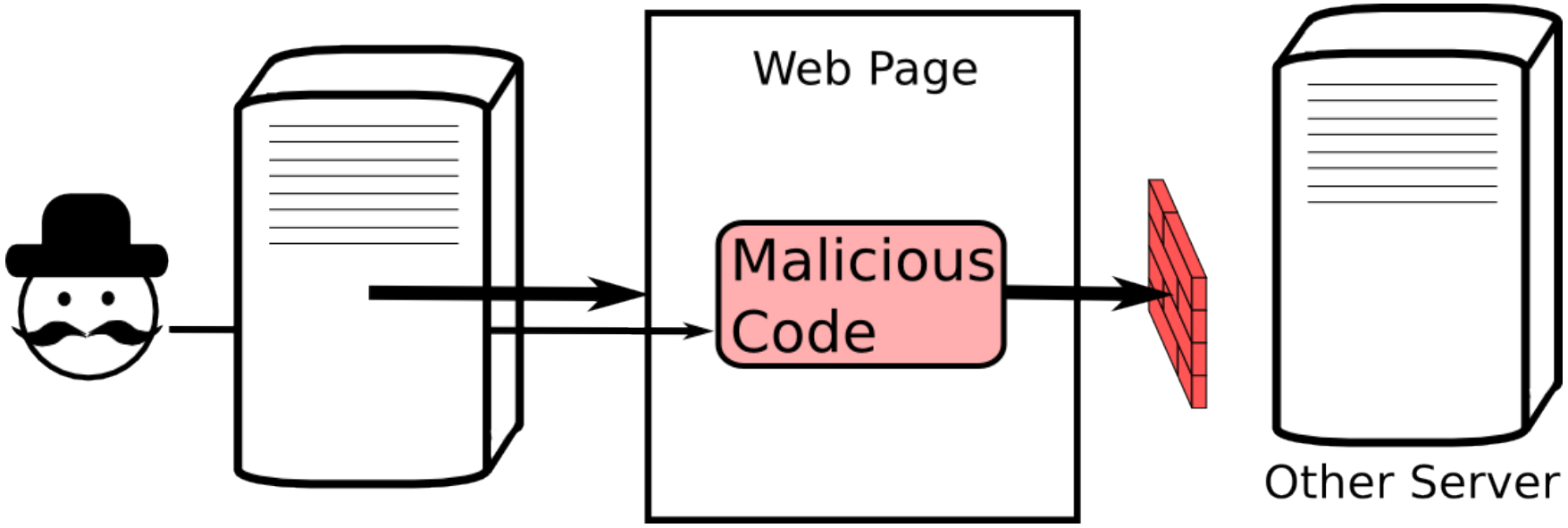
Unrestricted Outbound Communication

- Any script can read content from the **document** origin
- Transmission blocked by SOMA Manifest



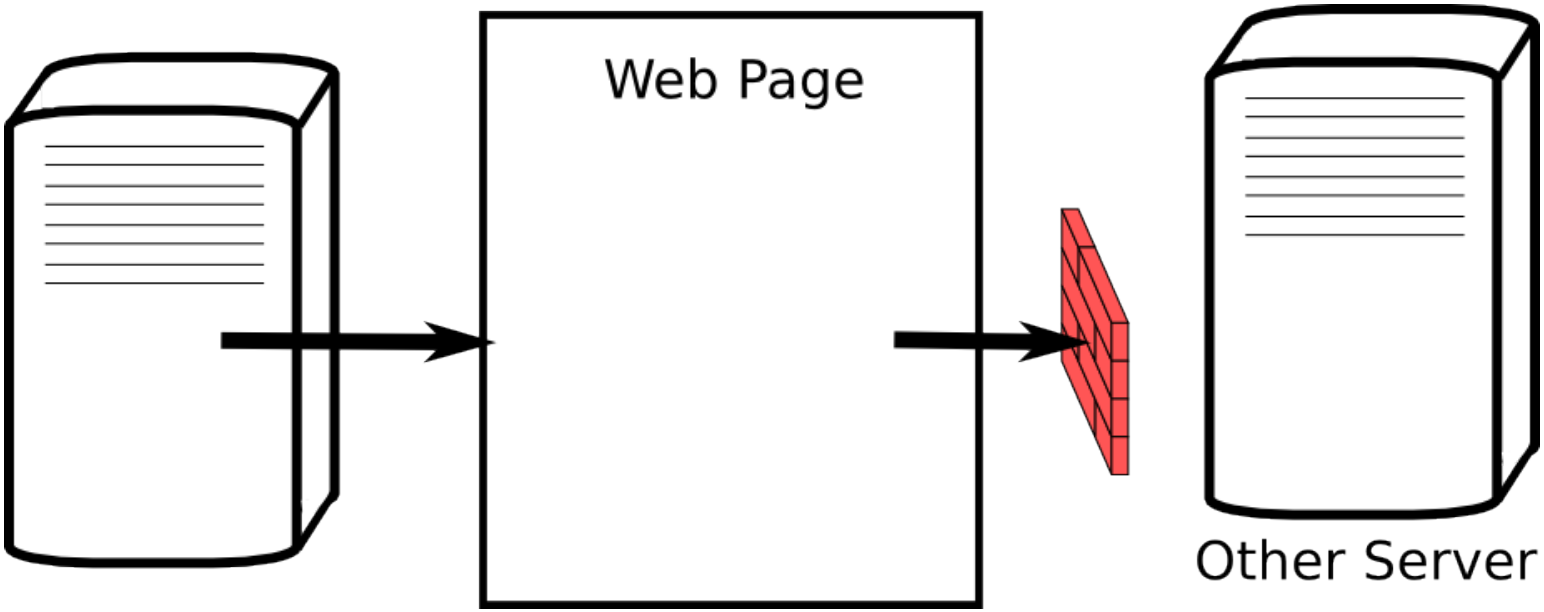
Cross Site Request Forgery

- A script can make requests to any domain
- Request blocked by SOMA Approval



Bandwidth Stealing

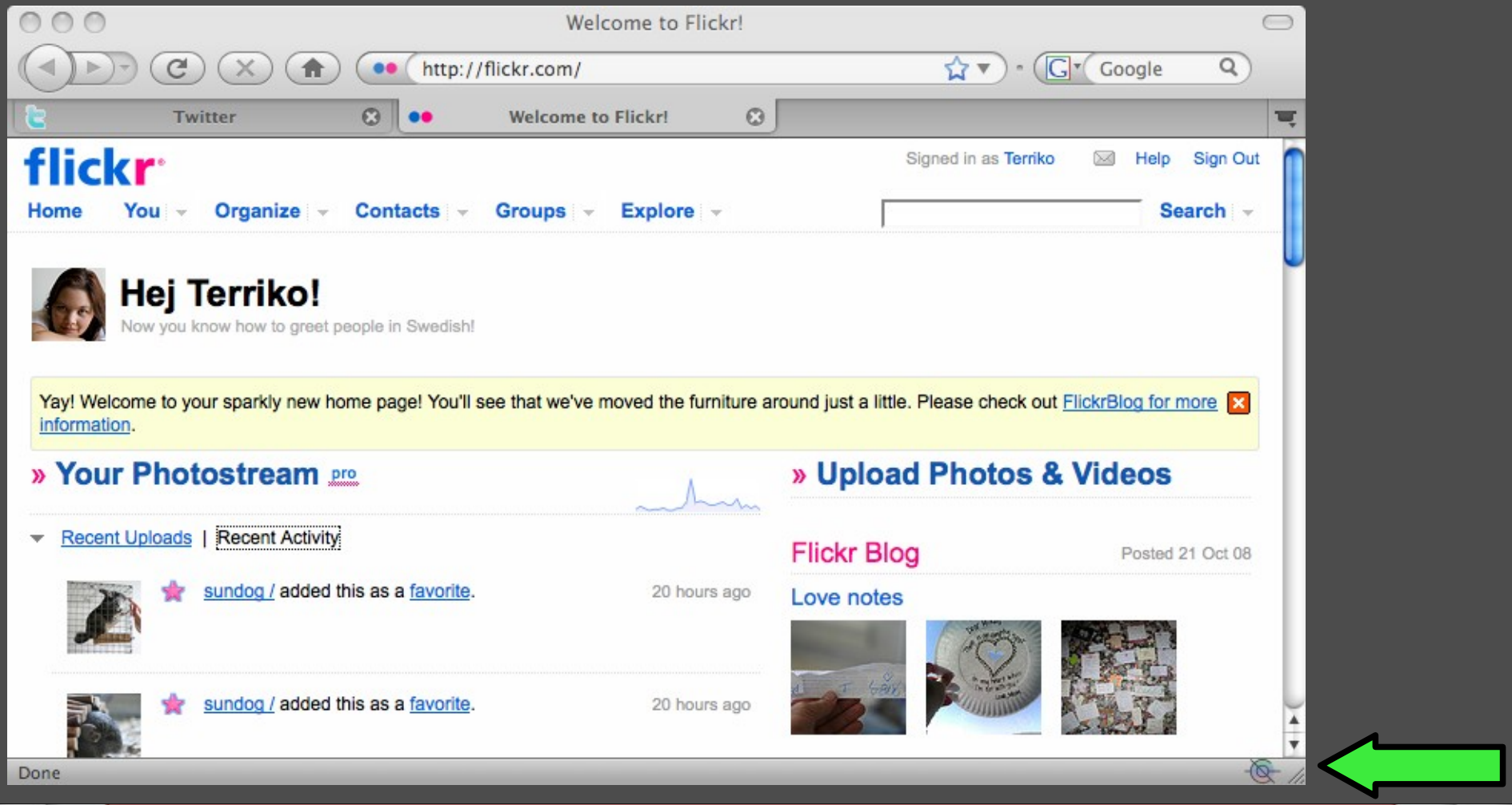
- A document can include content from anywhere
- Inclusion blocked by SOMA Approval



SOMA Prototype

- Mozilla Firefox 2 Add-on
  - also compatible with Firefox 3
  - can be downloaded and tried out
    - http://ccsl.carleton.ca/software/soma
- Fully backwards compatible
  - current websites appear unchanged
- Stops attacks discussed earlier
- Icon in statusbar indicates that SOMA is running

Screenshot of Prototype



Deployment

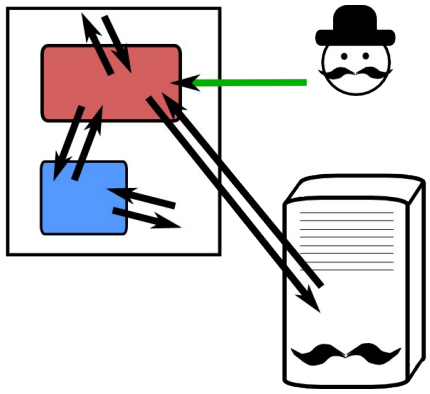
- Need:
  - minor modifications to browser
    - Mozilla SOMA Add-on implementation code is 12k
  - policy on origin & content providers (ideally)
    - some protection if either side provides policy
- Requires some additional network overhead
  - fetch manifest from origin
  - fetch approval from each content provider before fetching content
- Deployment is incremental

Performance

- Approvals overhead:
  - adds one additional round trip
  - estimated additional page load time is 5.58%
  - estimate probably overstated:
    - We used average content response size: 10459 bytes
    - soma-approval response size: 4 bytes (0.1% overhead)
      - independent of site complexity
- Manifest size:
  - checked front page of top 500 Alexa sites
  - average: 5.45 domains per site (5.3 stdev)

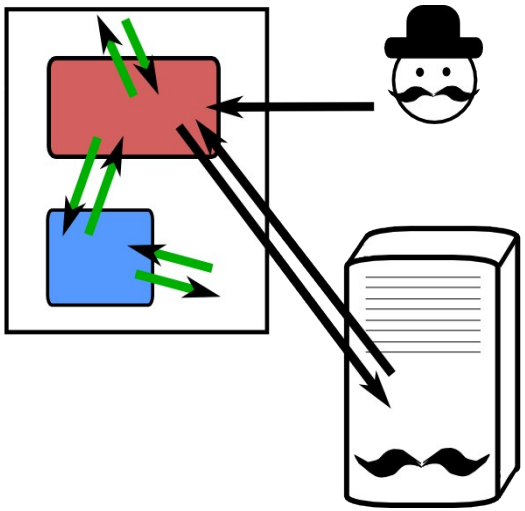
Complementary Work: Existing Code Injection Prevention

- Do careful input checking
  - risk of interactions with web page
  - difficult to do well
  - done by web programmer in source code
- Detect known code injection attacks
  - XSS, CSRF, SQL Injection
  - risk of false positives/missing new attacks
  - can be done by 3rd party tool
    - eg: web application firewalls



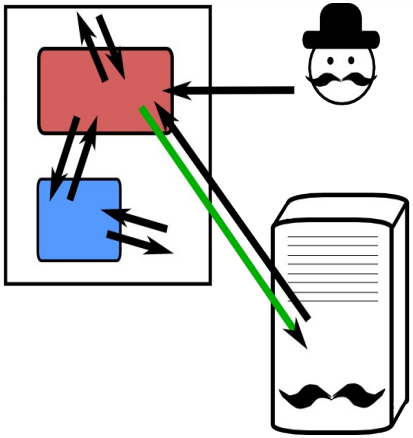
Complementary Work: Mashups

- A mashup is a web application which combines information and code from different sources
- There has been work on ways to make them more secure
  - better separation between components
  - communication between different contexts

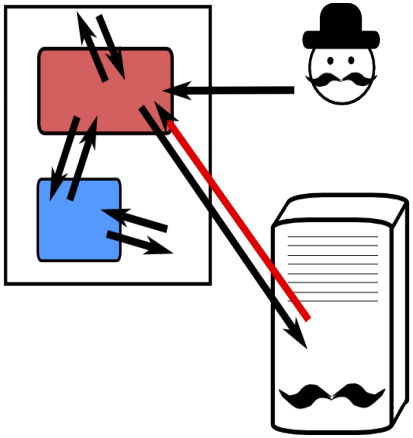


- Mashup work focuses on interactions within the page
  - SOMA focuses on interactions with external servers
- Requires use of tools by skilled web developers

Related Work: Tahoma and Flash



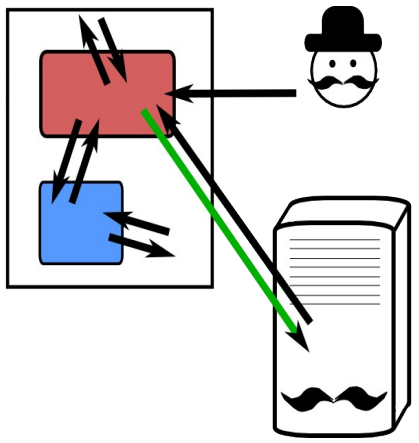
- Tahoma [Cox 2006]
  - SOMA Manifest for VM's



- Flash's `crossdomain.xml`
  - SOMA approvals for Flash

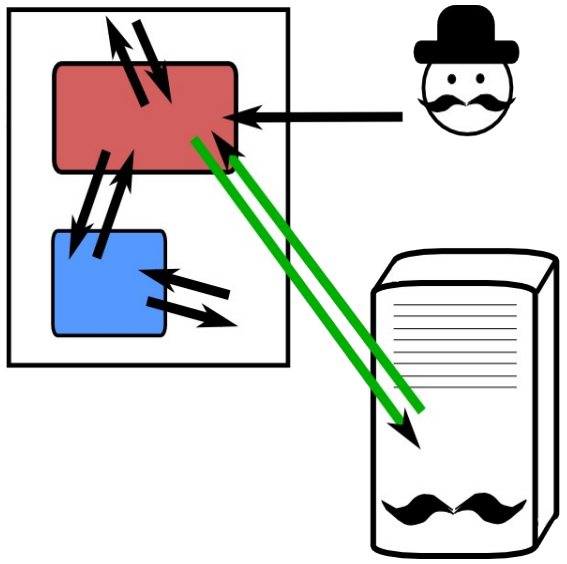
Related Work: Mozilla's Content Security Policy

- First version (“Site Security Policy”) similar to SOMA
- Most recent version has only manifest
  - Does not protect against cross site request forgery
- Other major differences:
  - policy is per-resource
  - more complex syntax required



SOMA Benefits

1. Incrementally deployable (with incremental benefit)
2. No configuration/usage burden on end users
3. Required changes/configuration are done by site operators
4. Changes are relatively simple to understand and easy to implement
5. Gives server operators the ability to specify which sites can interact with their content



Thanks!

- Carleton Computer Security Laboratory:
  - http://ccsl.carleton.ca
- SOMA Firefox Add-On (and more info):
  - http://ccsl.carleton.ca/software/soma