The Same-Origin Saga

Brendan Eich
AppSec USA 2012

A long time ago in a galaxy far, far away...

Netscape 2

- 1995 platform push against looming MSFT threat
- Frames, framesets, plugins (first in Netscape 1.1)
- JavaScript, in ten days in May
  - `window.open(`*`url`*`, `*`name`*`, `*`options`*`)`
  - `window.location, document.cookie`
  - `link.click(), form.submit()`
  - `javascript:` (generated docs, bookmarklets)
  - `document.write(s)` still tops for ad insertion

Origin

- Take the base URL of a loaded document
- Take the *`scheme://host:port`* prefix
- Label all code and data with origin
- Check "same origin" by string compare
- Originally, all `file:` URLs were same-origin!
  - Fixed to use file pathname as origin
  - Similarly for `imap:`, `mailbox:`, `news:`

Netscape 3

- `<script src="url"></script>`
- Cross-site `src` URL allowed -- just like `img`
- Like `#include` in C, remote script code given same origin as loading HTML
- `document.domain` for changing effective origin to join a super-domain

Apply Murphy's Law, Iterate

- Miserable XSS history elided...

Are browser vendors too conservative?

- Features are not free!
  - Simplicity as a selling point
  - Rely on addons for niche functionality
- Breakage is **very** expensive
  - Web sites slow to adapt
  - Switching costs are low

Browser Not Supported

The browser you are using is not supported by this application. If you wish to use this application, please use one of the links below to download and install the current version of a supported browser:

Microsoft Internet Explorer

Netscape

Firefox

No browser wins by taking overlarge risks

mozilla

No browser wins by taking overlarge risks

- Previous slide from Collin Jackson's USENIX Security 2011 invited talk, titled "**Crossing the Chasm: Pitching Security Research to Mainstream Browser Vendors**" [slides]

mozilla

No browser wins by taking overlarge risks

- Previous slide from Collin Jackson's USENIX Security 2011 invited talk, titled "**Crossing the Chasm: Pitching Security Research to Mainstream Browser Vendors**" [slides]

- Features are **not** free

mozilla

No browser wins by taking overlarge risks

- Previous slide from Collin Jackson's USENIX Security 2011 invited talk, titled "**Crossing the Chasm: Pitching Security Research to Mainstream Browser Vendors**" [slides]

- Features are **not** free

- Mozilla commissioned optional CPython integration work by Mark Hammond

mozilla

No browser wins by taking overlarge risks

- Previous slide from Collin Jackson's USENIX Security 2011 invited talk, titled "**Crossing the Chasm: Pitching Security Research to Mainstream Browser Vendors**" [slides]
- Features are **not** free
- Mozilla commissioned optional CPython integration work by Mark Hammond
  - Used by only one embedder

mozilla

No browser wins by taking overlarge risks

- Previous slide from Collin Jackson's USENIX Security 2011 invited talk, titled "**Crossing the Chasm: Pitching Security Research to Mainstream Browser Vendors**" [slides]
- Features are **not** free
- Mozilla commissioned optional CPython integration work by Mark Hammond
  - Used by only one embedder
  - No CPython distribution for Windows or Mac OS X

mozilla

No browser wins by taking overlarge risks

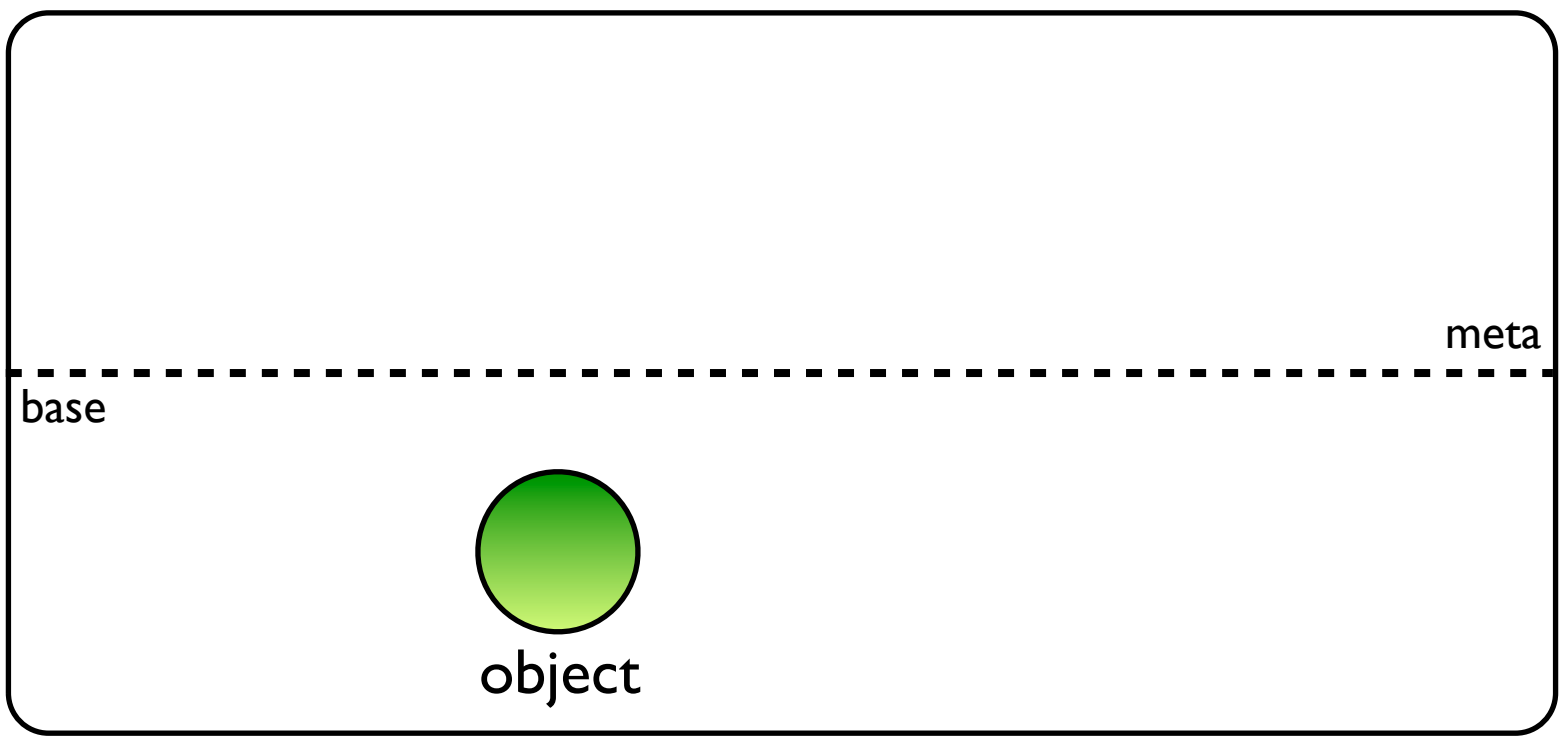
- Previous slide from Collin Jackson's USENIX Security 2011 invited talk, titled "**Crossing the Chasm: Pitching Security Research to Mainstream Browser Vendors**" [slides]
- Features are **not** free
- Mozilla commissioned optional CPython integration work by Mark Hammond
  - Used by only one embedder
  - No CPython distribution for Windows or Mac OS X
  - High sunk and recurring engineering costs

mozilla

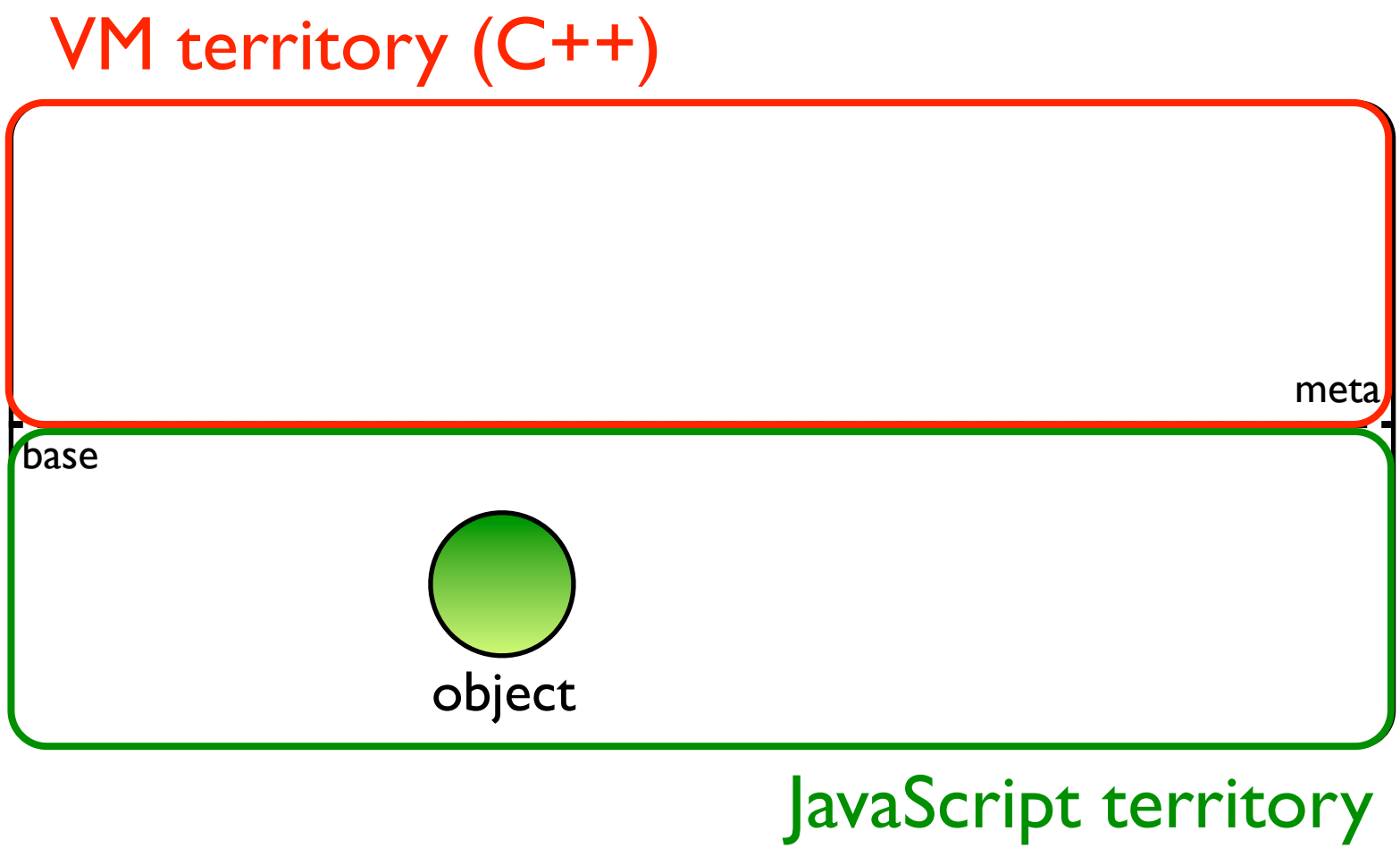
Hopeful Developments

- JS is almost OCap (Object Capability)
- Browsers use OCap under the hood
- ES5 and ES6 strengthen JS's OCap nature
  - Better Caja/SES efficiency
  - `"use strict"` pseudo-pragma
- OWASP best practices, Origin header, etc. help outside of JS attack vectors

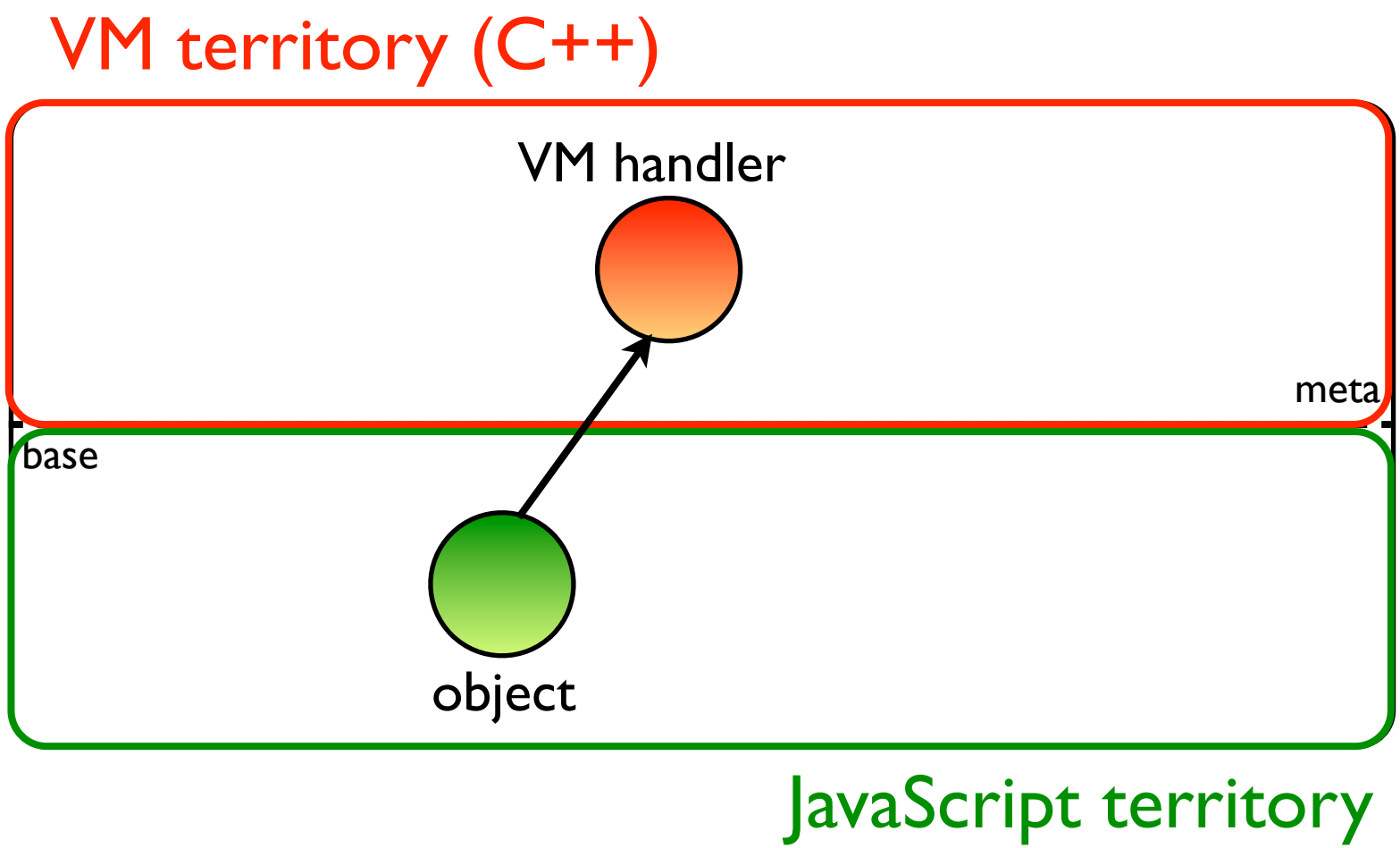
Selective Interception



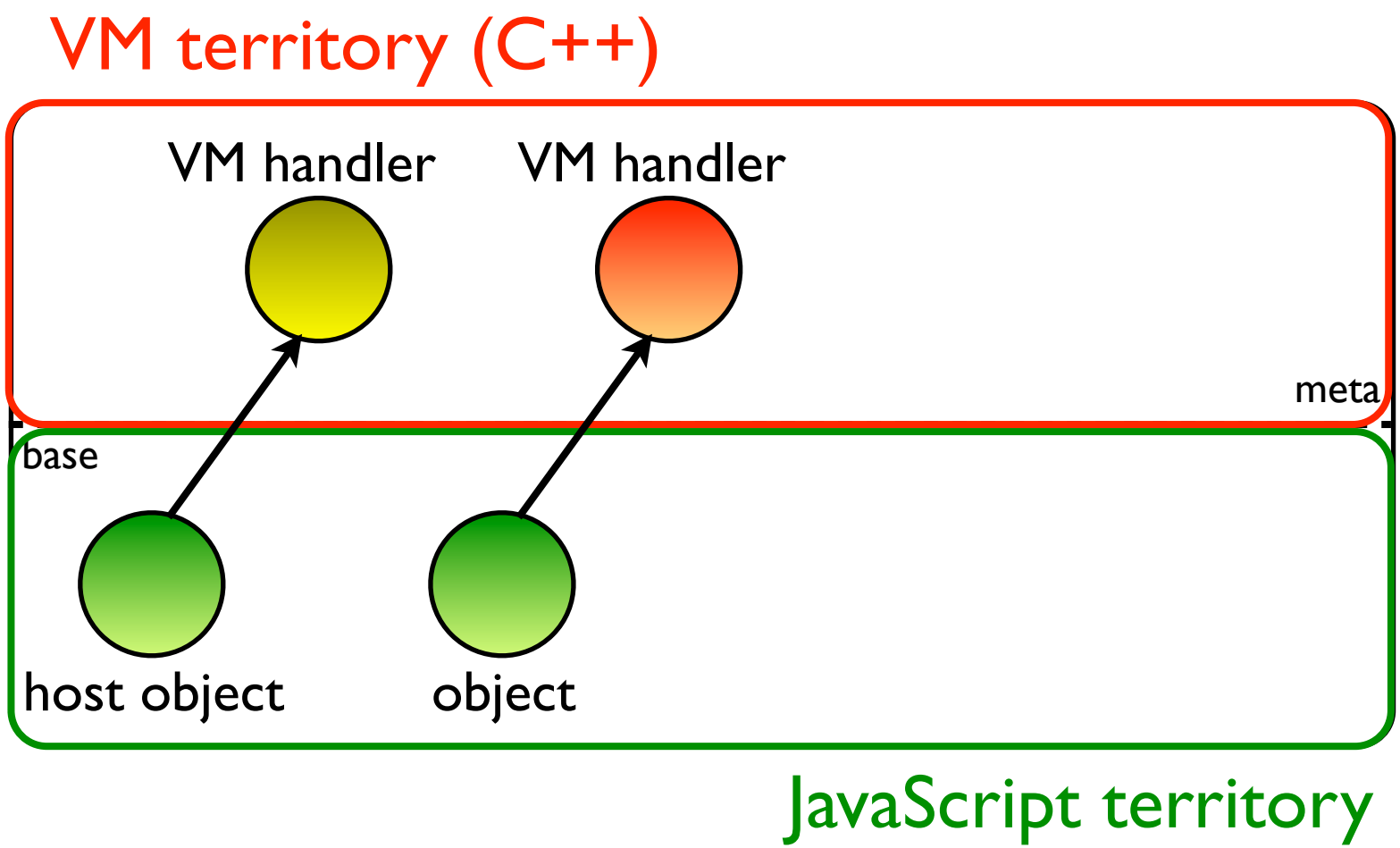
Selective Interception



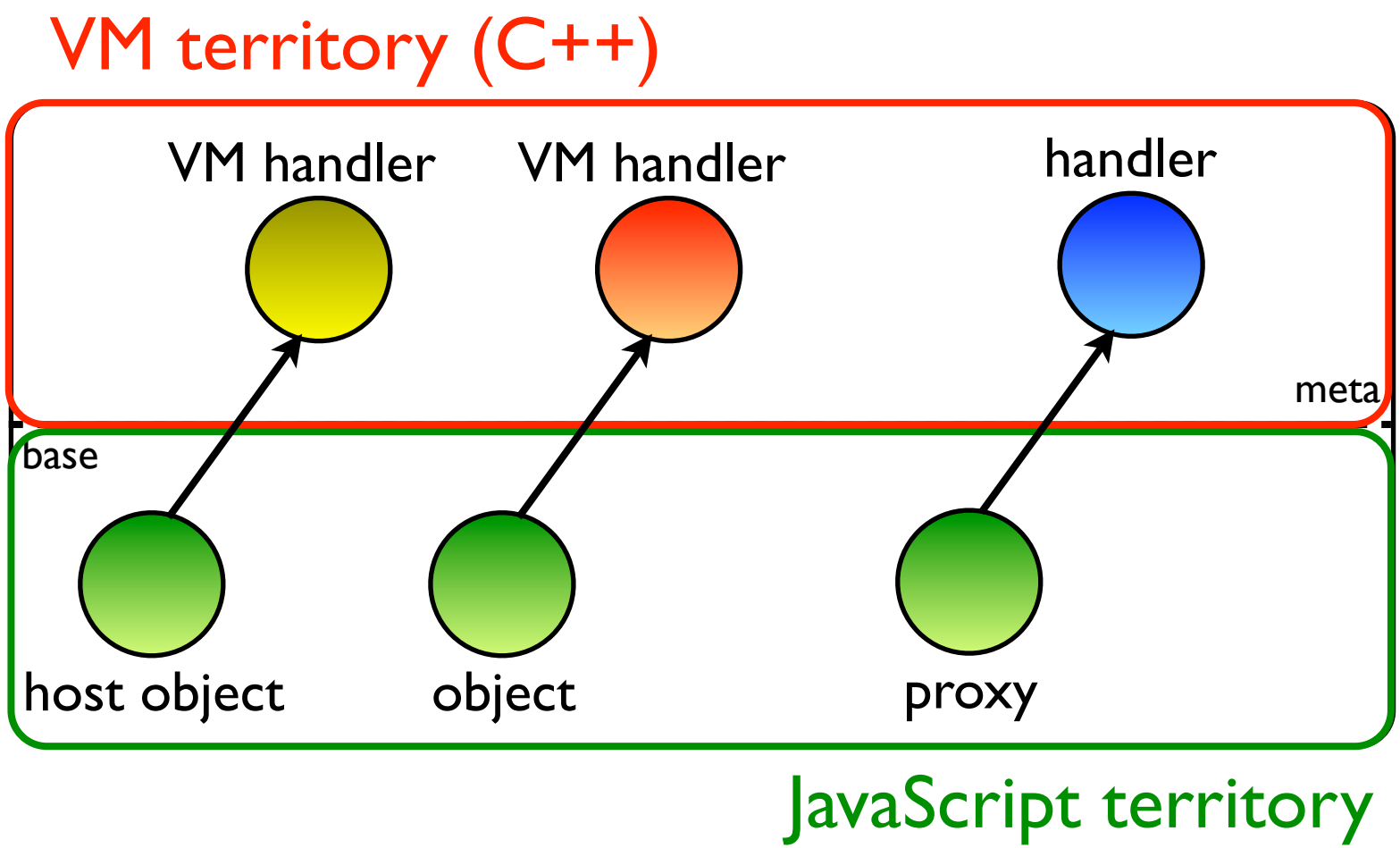
Selective Interception



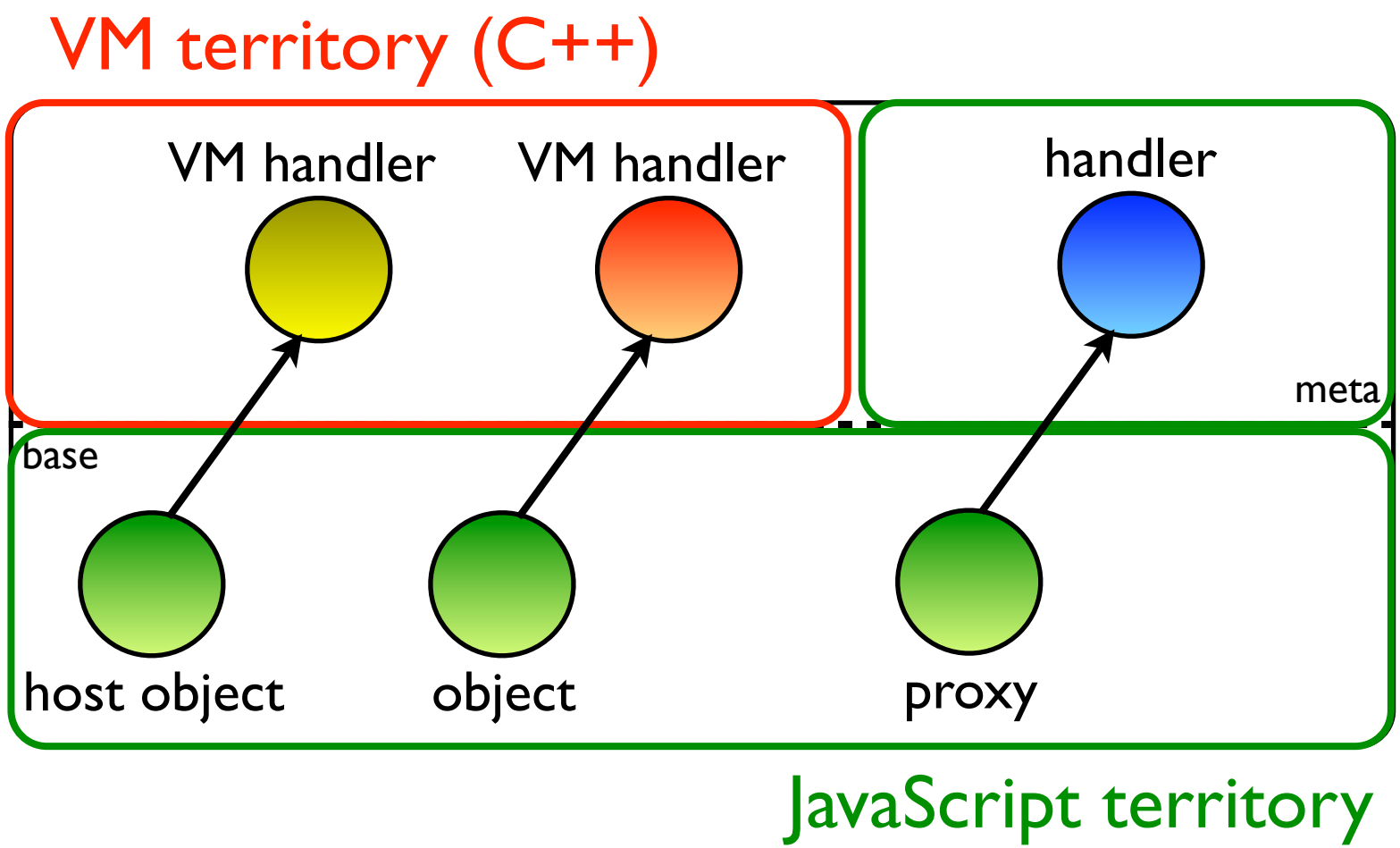
Selective Interception



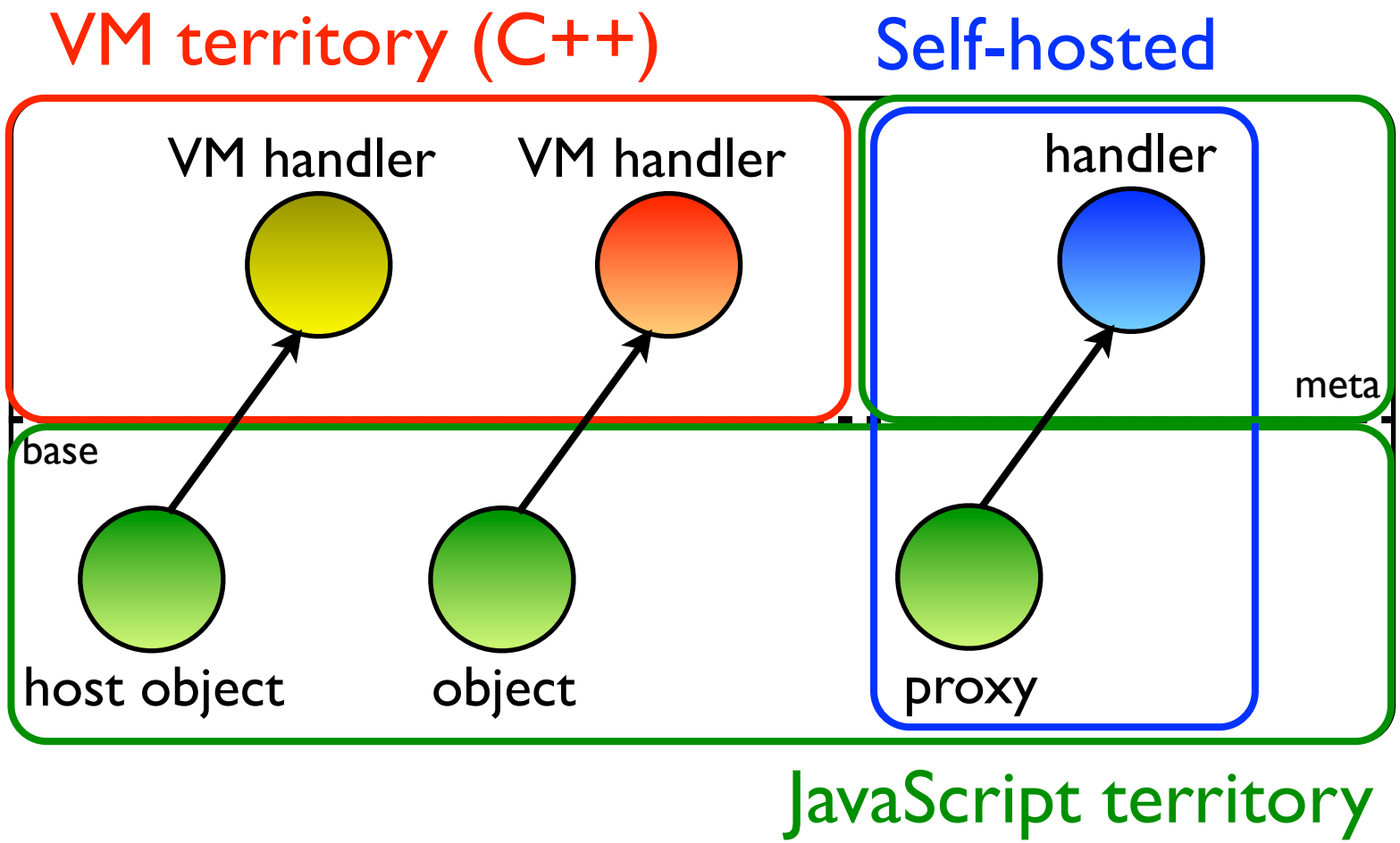
Selective Interception



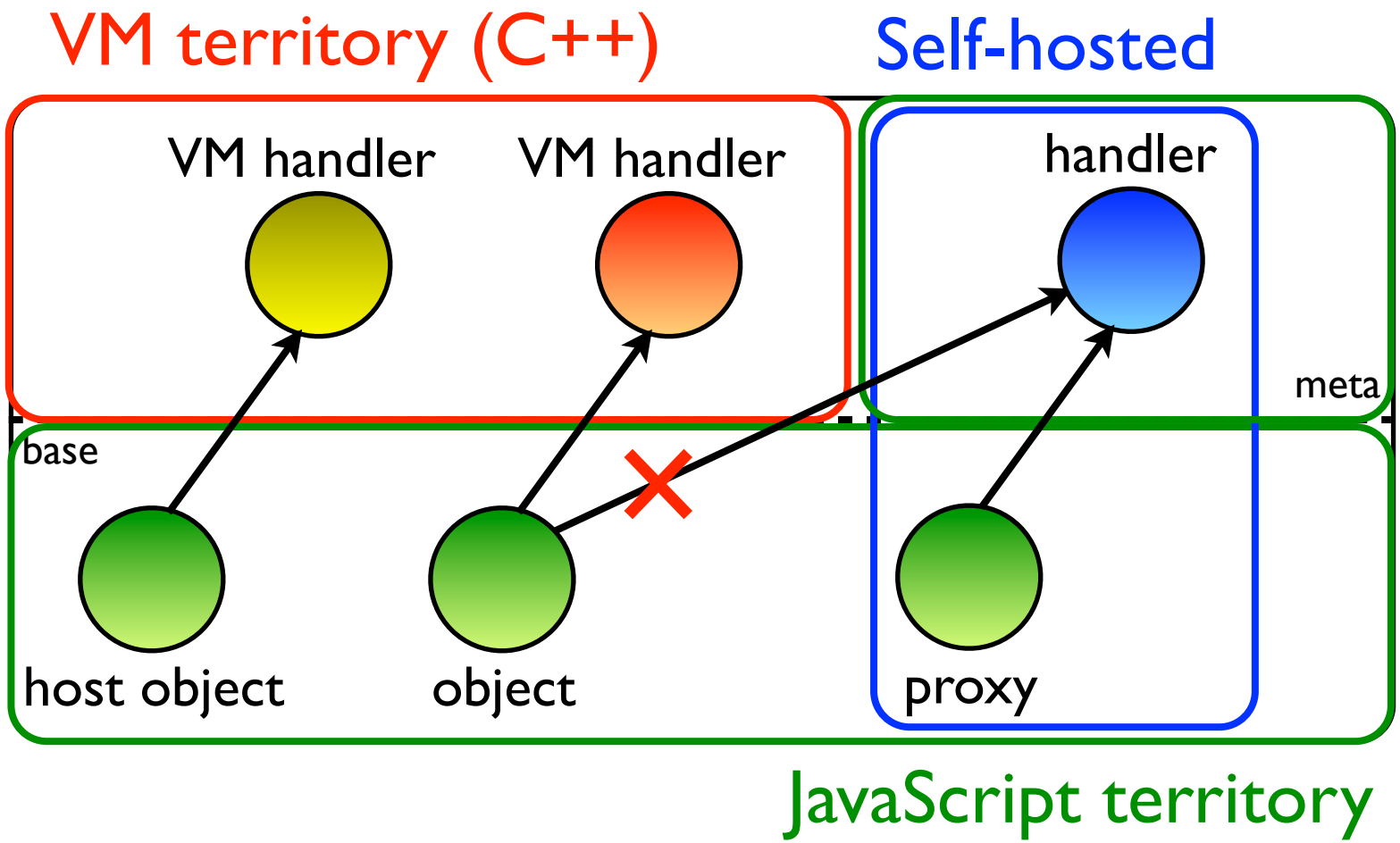
Selective Interception



Selective Interception



Selective Interception

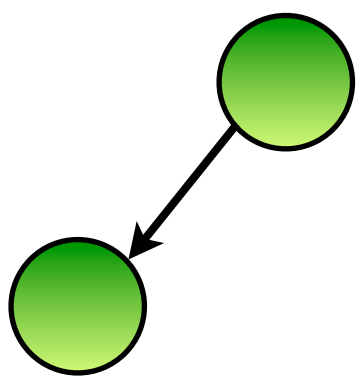


Example: Membranes

- Firefox security wrappers (anti-XSS, XUL, etc.)
- Google Caja: capability-secure subset
- Object-capability model: an object is powerless unless given a reference to other objects
- References can be made revocable through a membrane

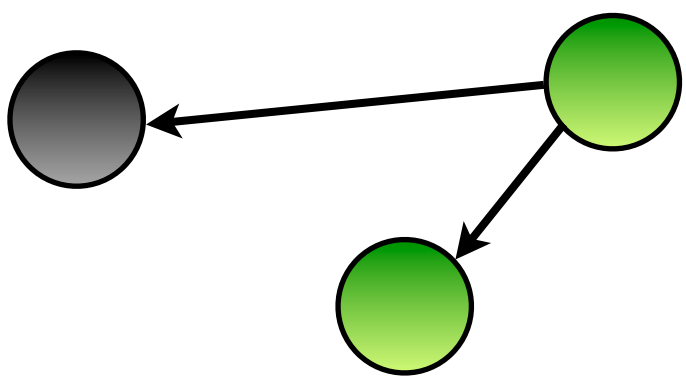
Example: Membranes

- Firefox security wrappers (anti-XSS, XUL, etc.)
- Google Caja: capability-secure subset
- Object-capability model: an object is powerless unless given a reference to other objects
- References can be made revocable through a membrane



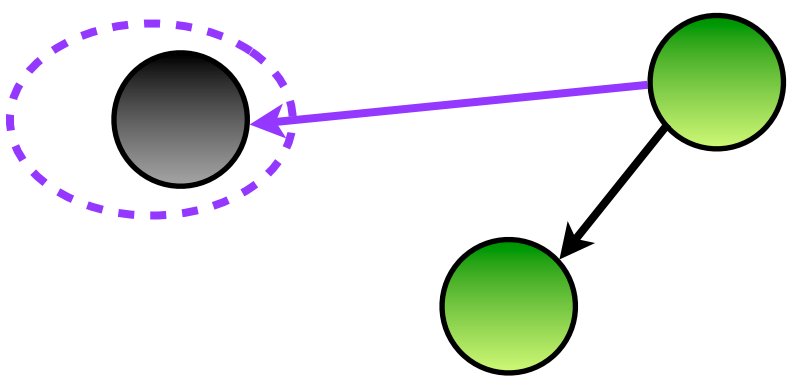
Example: Membranes

- Firefox security wrappers (anti-XSS, XUL, etc.)
- Google Caja: capability-secure subset
- Object-capability model: an object is powerless unless given a reference to other objects
- References can be made revocable through a membrane



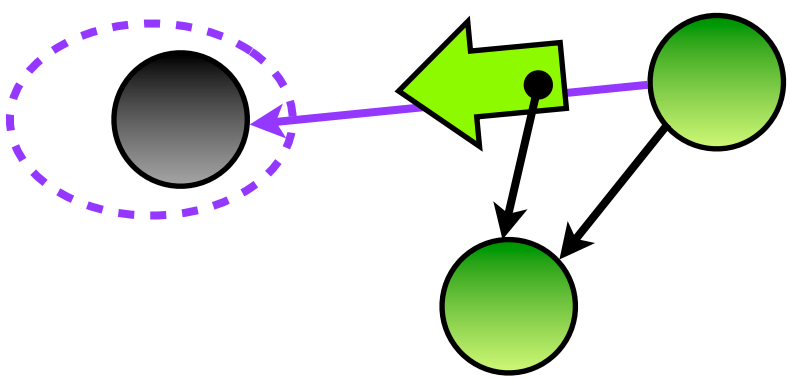
Example: Membranes

- Firefox security wrappers (anti-XSS, XUL, etc.)
- Google Caja: capability-secure subset
- Object-capability model: an object is powerless unless given a reference to other objects
- References can be made revocable through a membrane



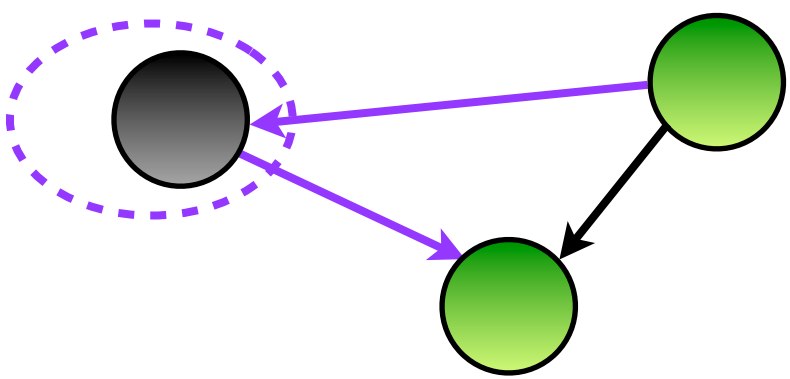
Example: Membranes

- Firefox security wrappers (anti-XSS, XUL, etc.)
- Google Caja: capability-secure subset
- Object-capability model: an object is powerless unless given a reference to other objects
- References can be made revocable through a membrane



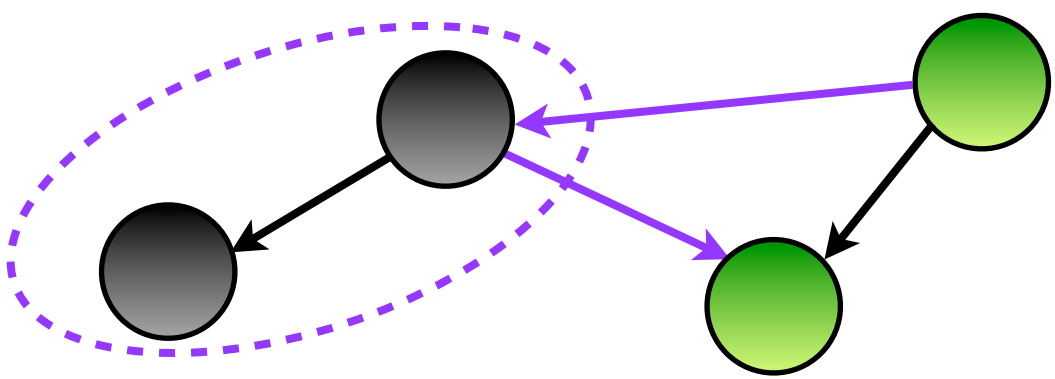
Example: Membranes

- Firefox security wrappers (anti-XSS, XUL, etc.)
- Google Caja: capability-secure subset
- Object-capability model: an object is powerless unless given a reference to other objects
- References can be made revocable through a membrane



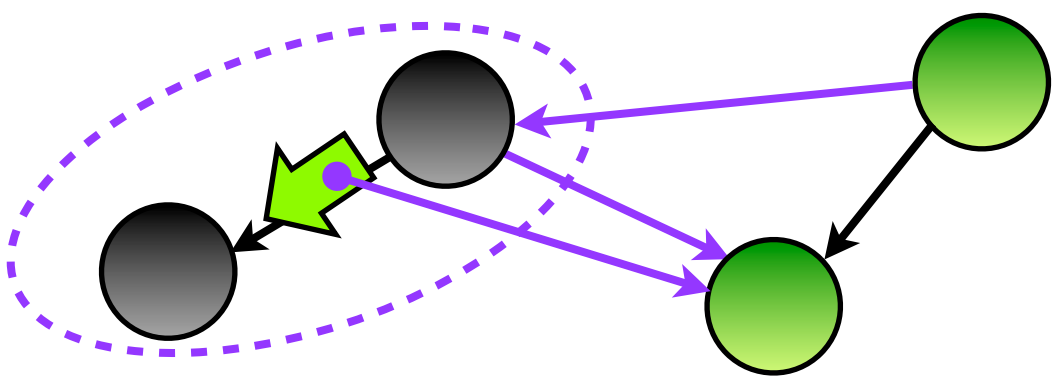
Example: Membranes

- Firefox security wrappers (anti-XSS, XUL, etc.)
- Google Caja: capability-secure subset
- Object-capability model: an object is powerless unless given a reference to other objects
- References can be made revocable through a membrane



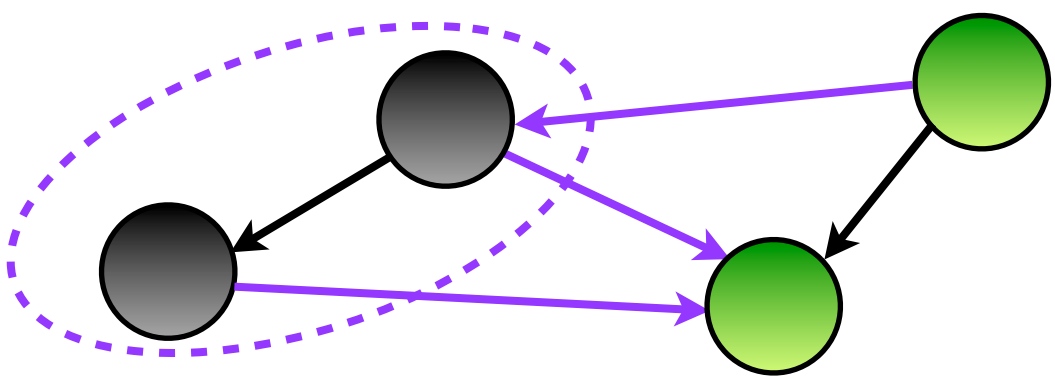
Example: Membranes

- Firefox security wrappers (anti-XSS, XUL, etc.)
- Google Caja: capability-secure subset
- Object-capability model: an object is powerless unless given a reference to other objects
- References can be made revocable through a membrane



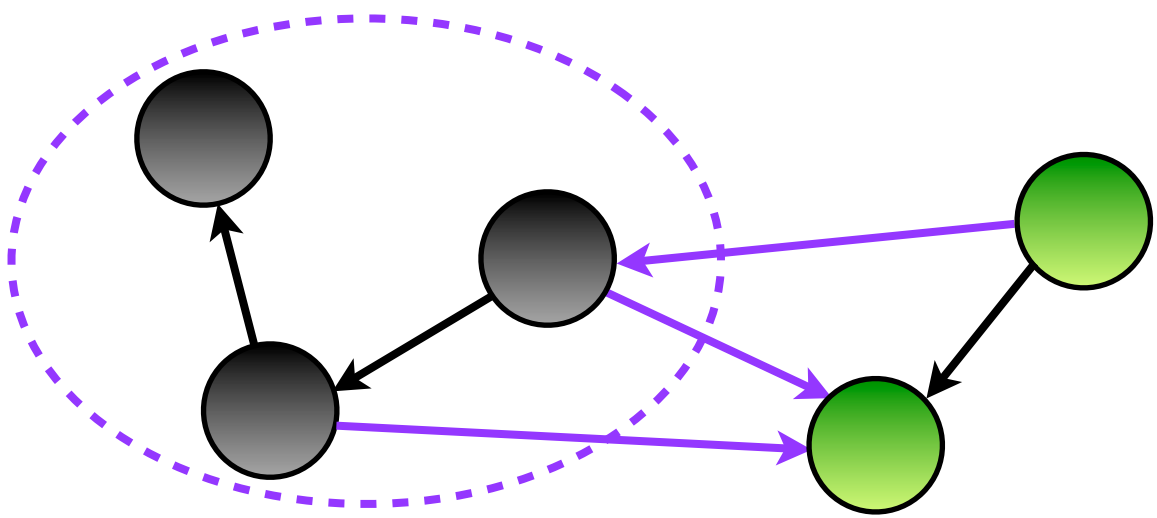
Example: Membranes

- Firefox security wrappers (anti-XSS, XUL, etc.)
- Google Caja: capability-secure subset
- Object-capability model: an object is powerless unless given a reference to other objects
- References can be made revocable through a membrane



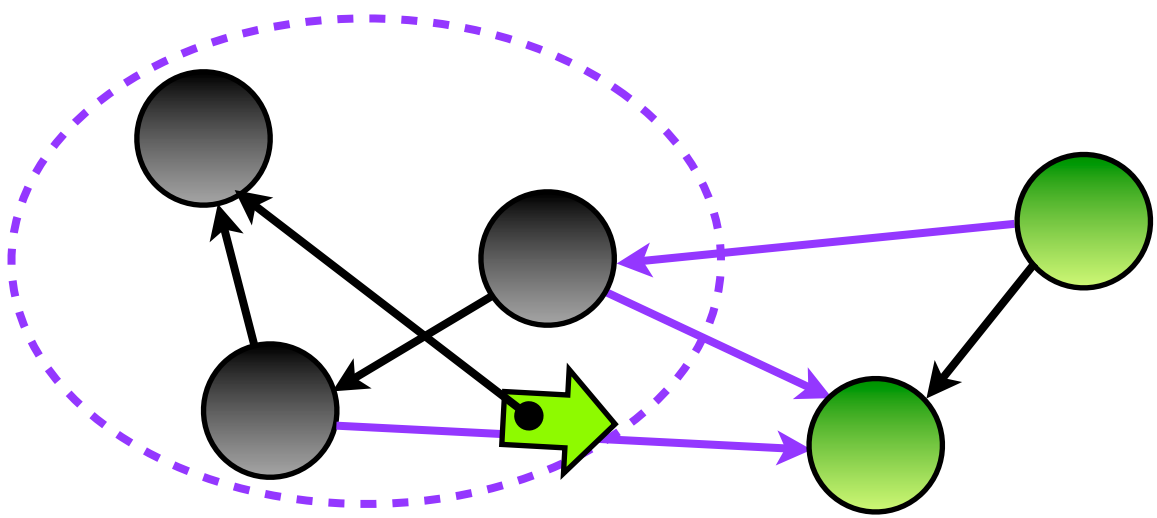
Example: Membranes

- Firefox security wrappers (anti-XSS, XUL, etc.)
- Google Caja: capability-secure subset
- Object-capability model: an object is powerless unless given a reference to other objects
- References can be made revocable through a membrane



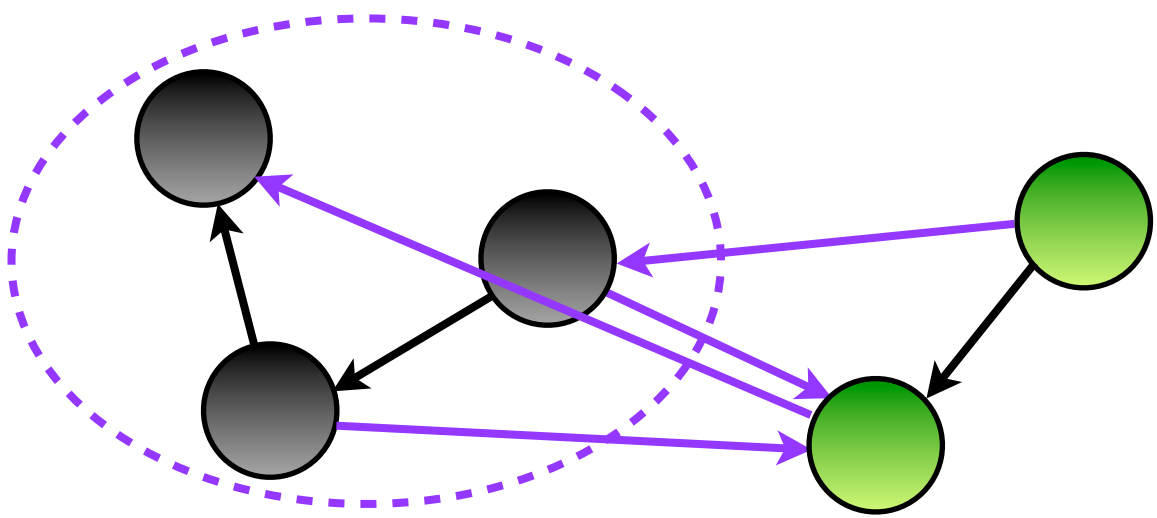
Example: Membranes

- Firefox security wrappers (anti-XSS, XUL, etc.)
- Google Caja: capability-secure subset
- Object-capability model: an object is powerless unless given a reference to other objects
- References can be made revocable through a membrane



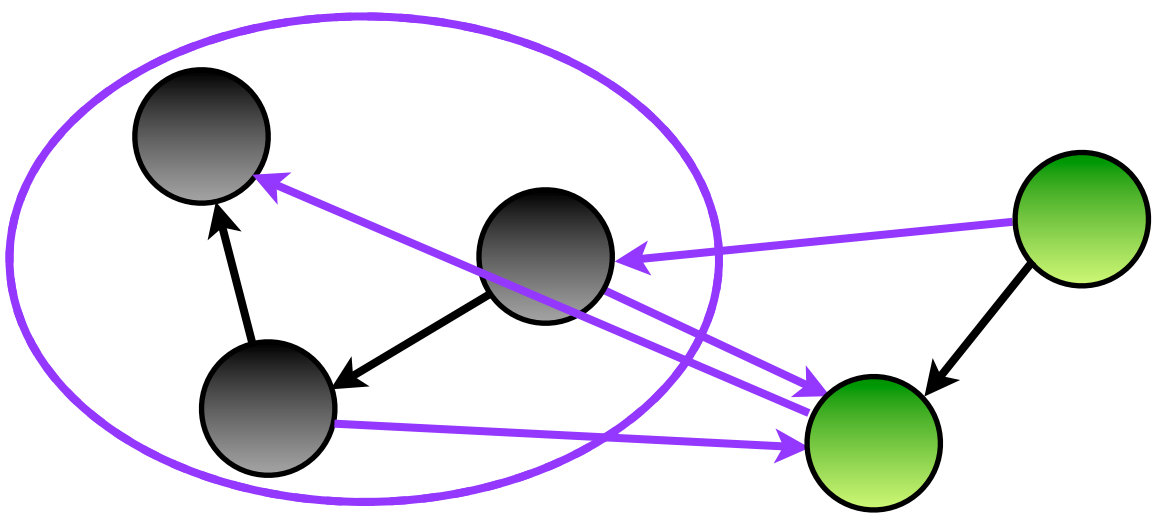
Example: Membranes

- Firefox security wrappers (anti-XSS, XUL, etc.)
- Google Caja: capability-secure subset
- Object-capability model: an object is powerless unless given a reference to other objects
- References can be made revocable through a membrane



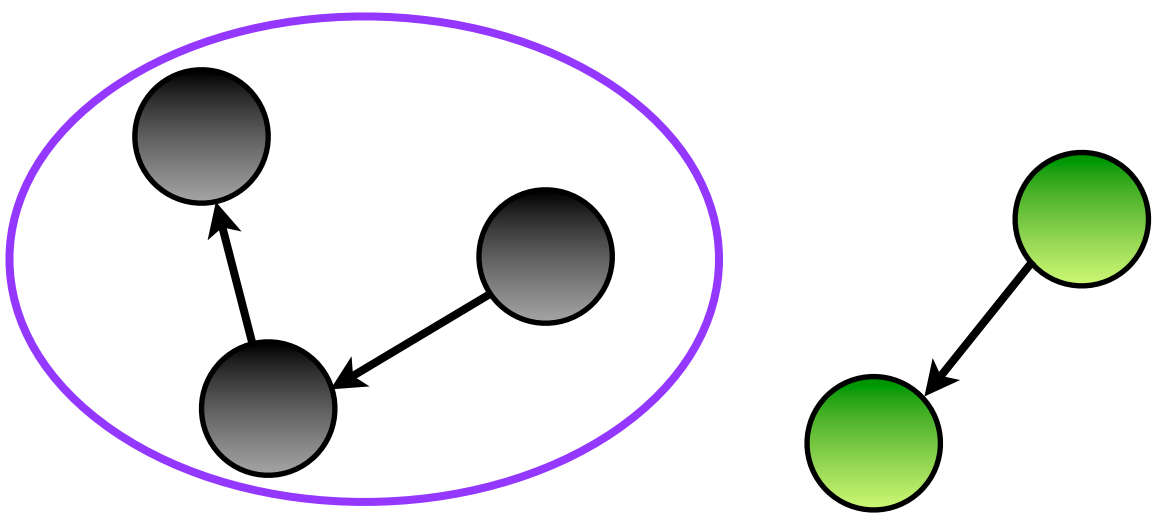
Example: Membranes

- Firefox security wrappers (anti-XSS, XUL, etc.)
- Google Caja: capability-secure subset
- Object-capability model: an object is powerless unless given a reference to other objects
- References can be made revocable through a membrane

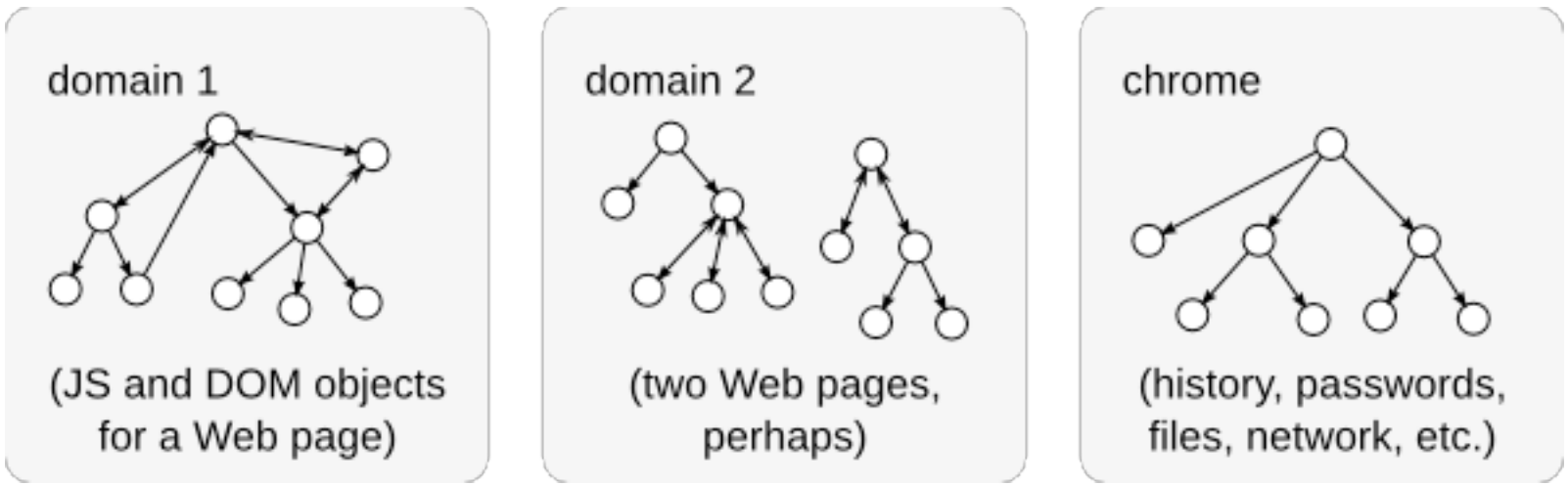


Example: Membranes

- Firefox security wrappers (anti-XSS, XUL, etc.)
- Google Caja: capability-secure subset
- Object-capability model: an object is powerless unless given a reference to other objects
- References can be made revocable through a membrane

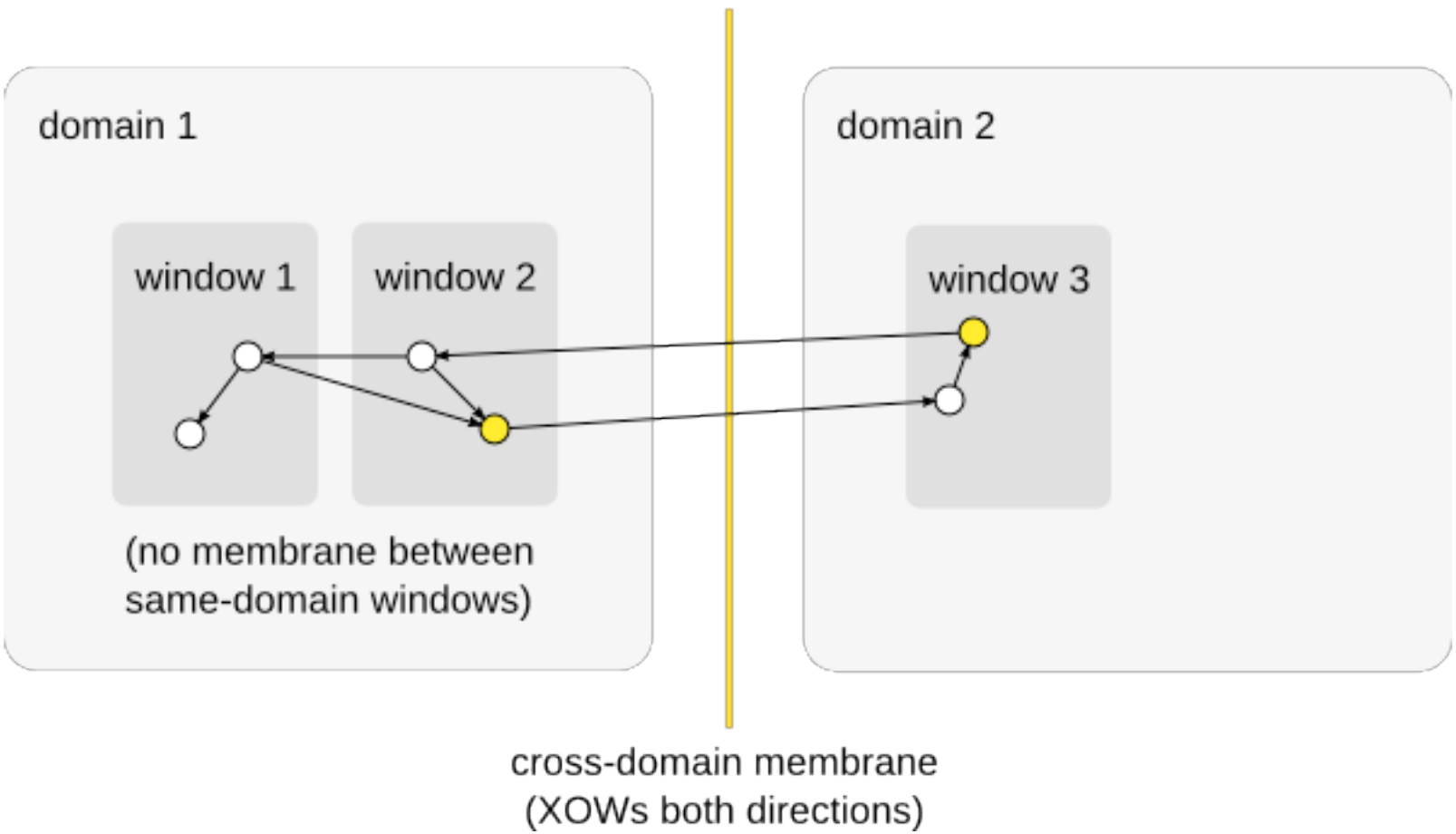


Compartments (Isolated Heaps)



Cross-Compartment Mediation

- Cross-Origin Wrapper



Conclusions + Question

- Same-Origin Policy continues to evolve
- Better isolation, OCap membranes reduce attack surface
- Can we break the web cooperatively?
  - Label cross-site `<script src="url2">` in document from `url1` with subordinate origin computed from both URLs?