Web Security

CS 161: Computer Security

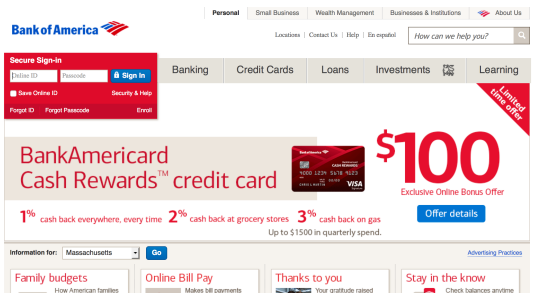
Prof. Raluca Ada Popa

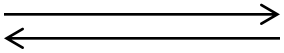
February 3, 2015

What is the Web?

A platform for deploying applications and sharing information, *portably* and *securely*

client browser





web server

Bank of America

HTTP

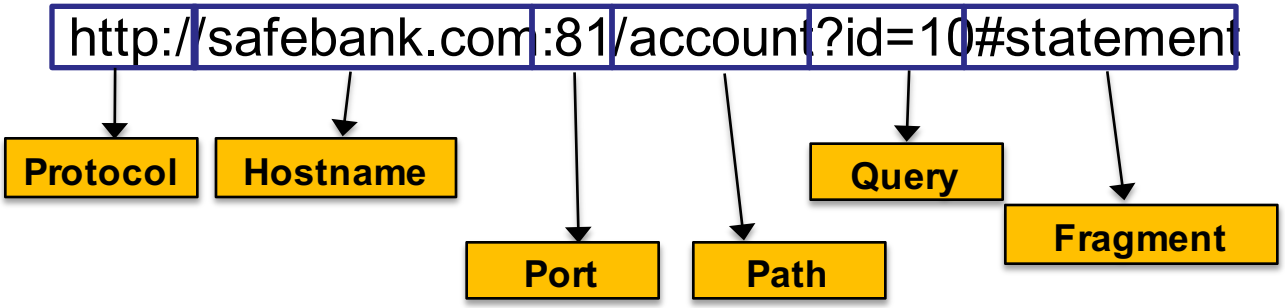
(Hypertext Transfer Protocol)

A common data communication protocol on the web

URLs

Global identifiers of network-retrievable resources

Example:



HTTP

CLIENT BROWSER

WEB SERVER

HTTP REQUEST:

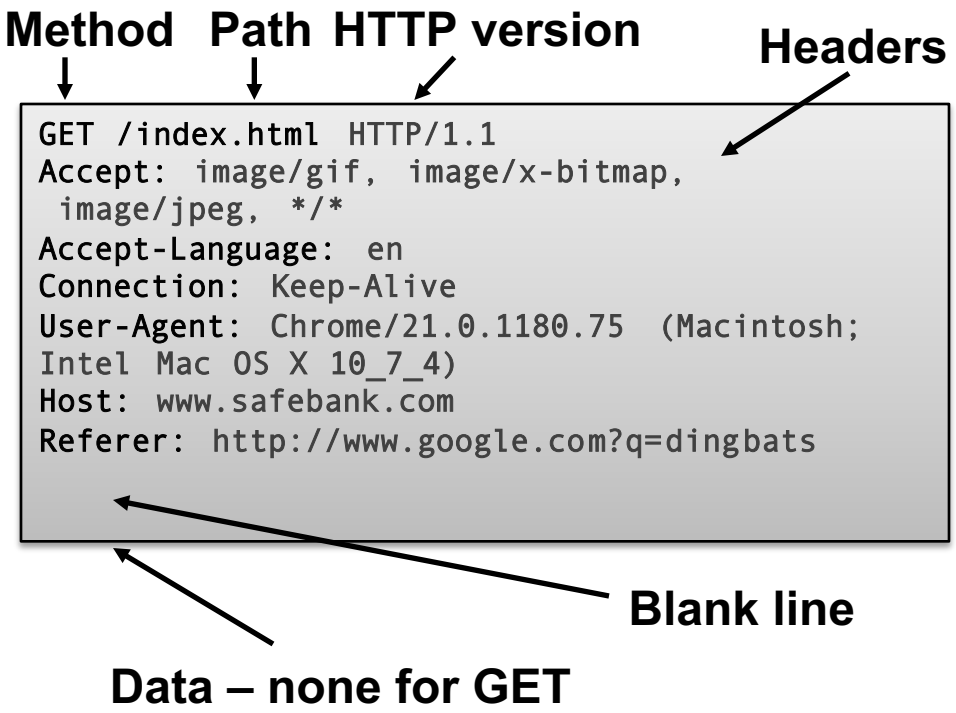
```
GET /account.html HTTP/1.1
Host: www.safebank.com
```

HTTP RESPONSE:

```
HTTP/1.0 200 OK
<HTML> . . . </HTML>
```

HTTP Request

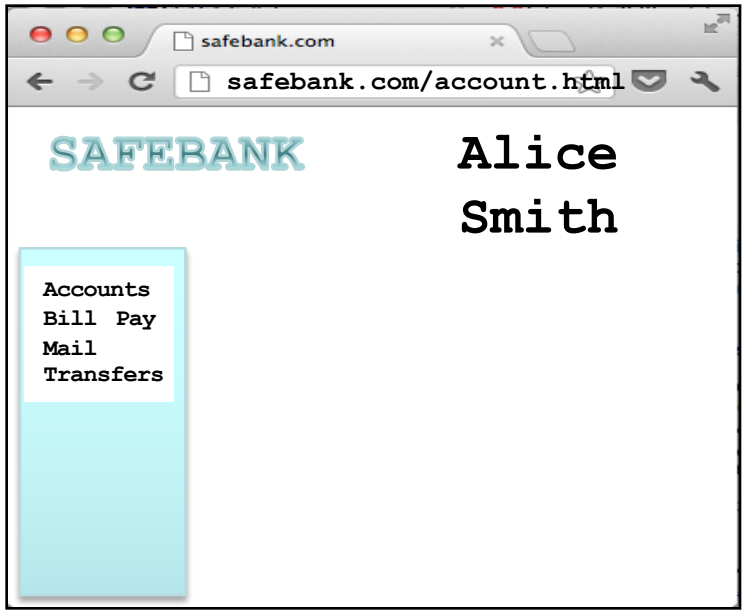
GET: no side effect
POST: possible side effect



HTTP

CLIENT BROWSER

WEB SERVER



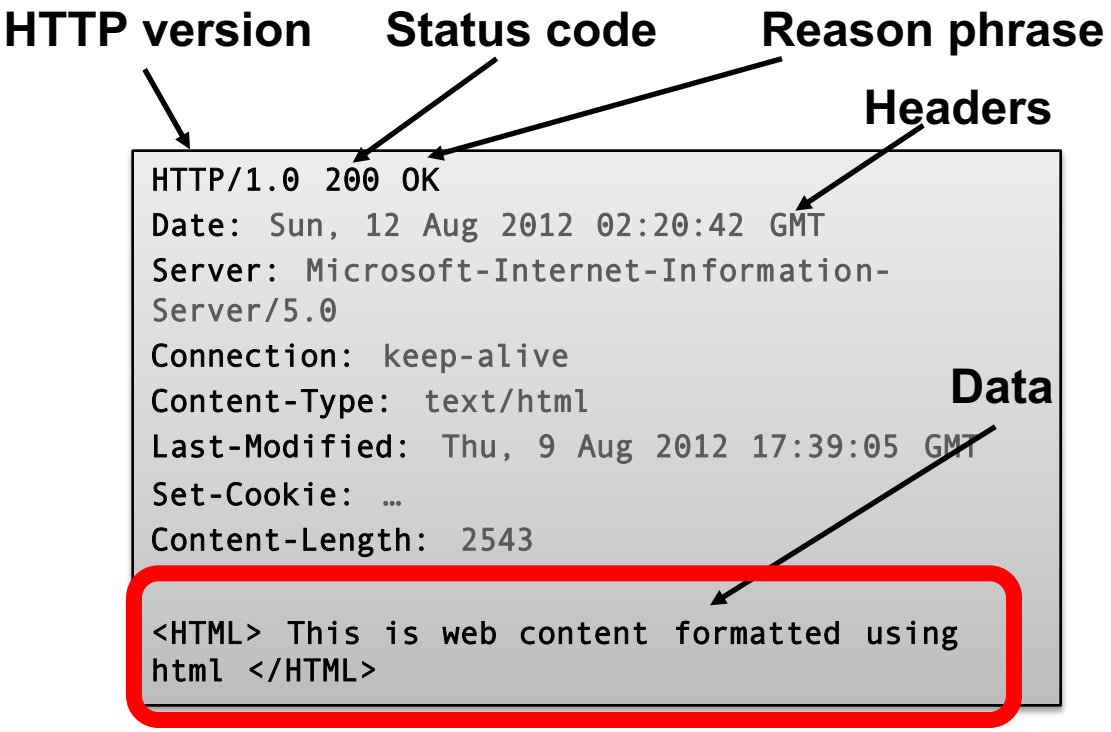
HTTP REQUEST:

```
GET /account.html HTTP/1.1
Host: www.safebank.com
```

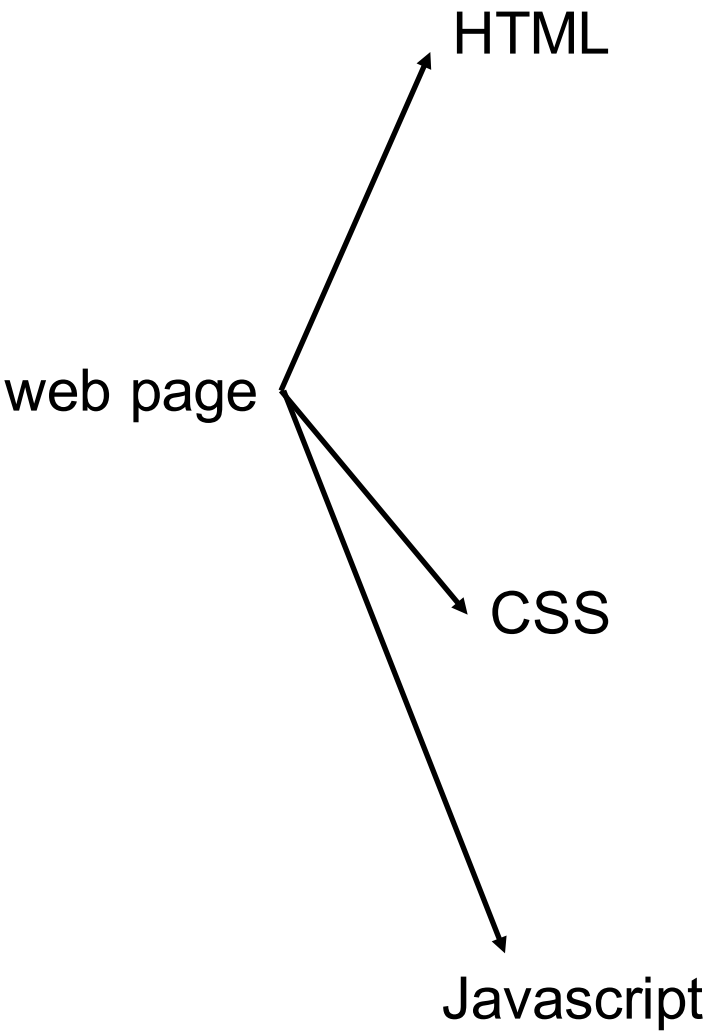
HTTP RESPONSE:

```
HTTP/1.0 200 OK
<HTML> . . . </HTML>
```

HTTP Response



Web page



HTML

A language to create structured documents
One can embed images, objects, or create interactive forms

index.html

```
<html>
    <body>
        <div>
            foo
            <a href="http://google.com">Go  to Google!</a>
        </div>
        <form>
            <input type="text"  />
            <input type="radio"  />
            <input type="checkbox"  />
        </form>
    </body>
</html>
```

CSS

Style sheet language used for describing the presentation of a document

index.css

```
p.serif {
font-family: "Times New Roman", Times, serif;
}
p.sansserif {
font-family: Arial, Helvetica, sans-serif;
}
```

Javascript

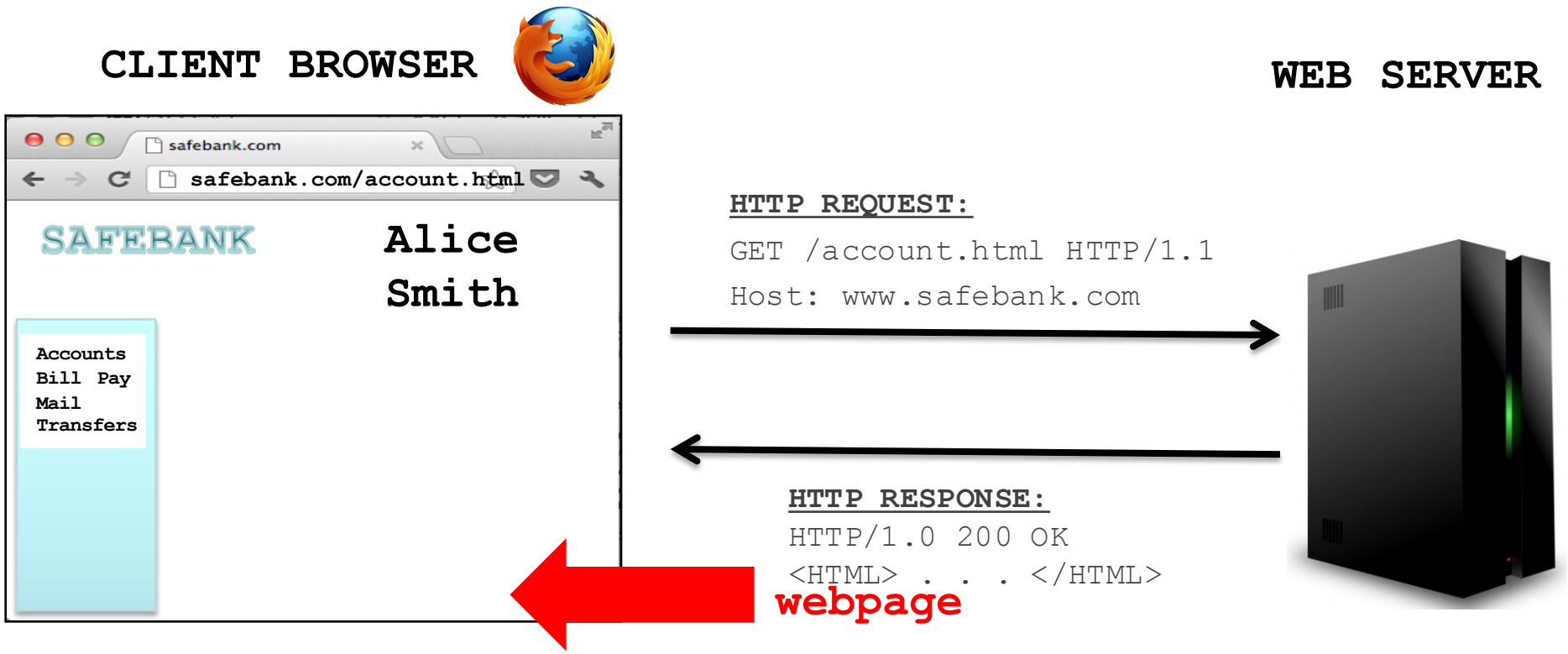
Programming language used to manipulate web pages

Supported by all web browsers

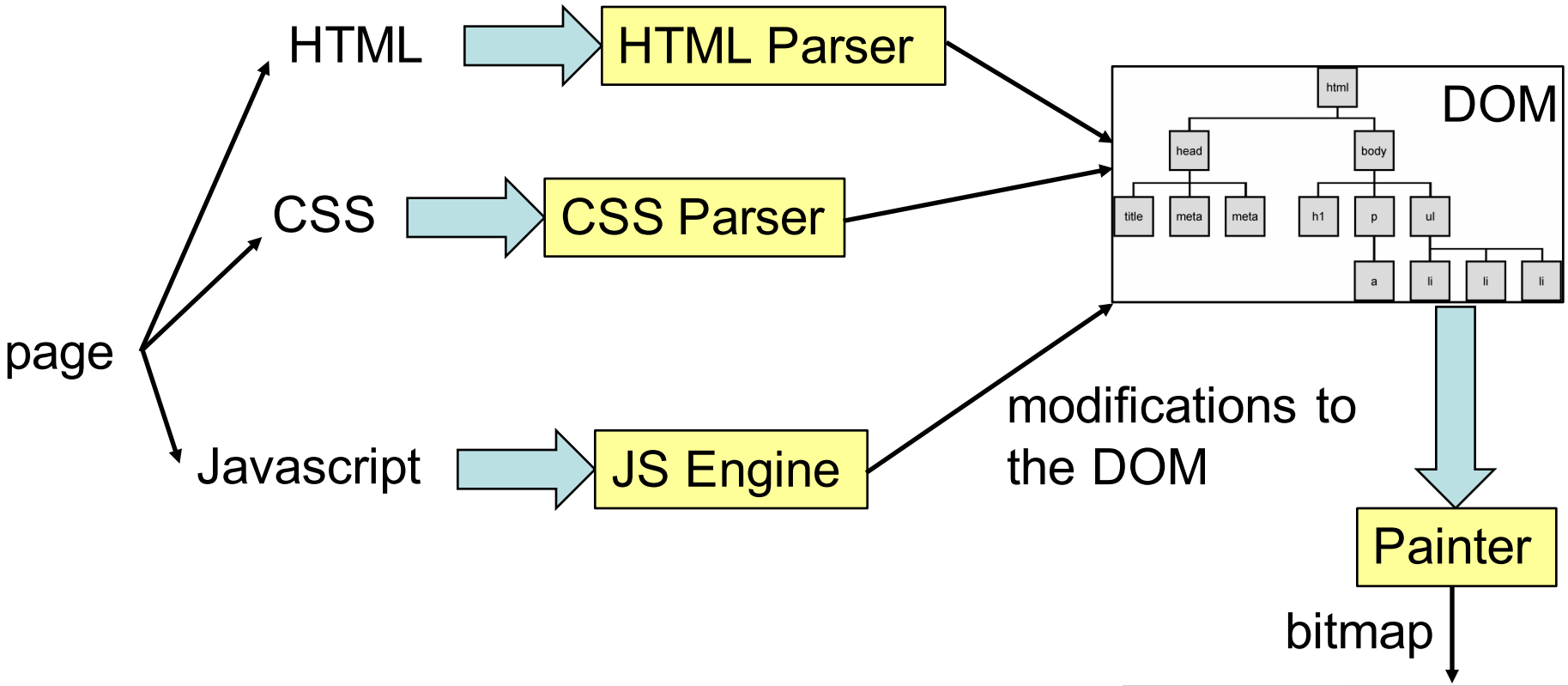
```
<script>
function myFunction() {
document.getElementById("demo").innerHTML = ”Text changed.";
}
</script>
```

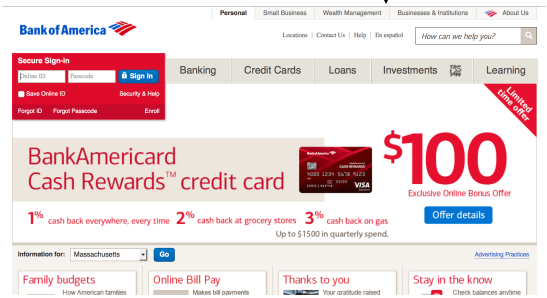
Very powerful!

HTTP


CLIENT BROWSER
WEB SERVER
safebank.com
safebank.com/account.html
SAFEBANK
Alice
Smith
Accounts
Bill Pay
Mail
Transfers
HTTP REQUEST:
GET /account.html HTTP/1.1
Host: www.safebank.com
HTTP RESPONSE:
HTTP/1.0 200 OK
<HTML> . . . </HTML>
webpage

Page rendering



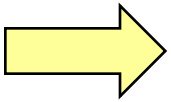


DOM (Document Object Model)

a cross-platform model for representing and interacting with objects in HTML

HTML

```
<html>
    <body>
        <div>
            foo
        </div>
        <form>
            <input type="text" />
            <input type="radio" />
            <input type="checkbox" />
        </form>
    </body>
</html>
```



DOM Tree

```
|-> Document
    |-> Element (<html>)
        |-> Element (<body>)
            |-> Element (<div>)
                |-> text node
            |-> Form
                |-> Text-box
                |-> Radio Button
                |-> Check Box
                |-> Button
```

Web & HTTP 101

CLIENT BROWSER

WEB SERVER

HTTP REQUEST:

```
GET /account.html HTTP/1.1
Host: www.safebank.com
```

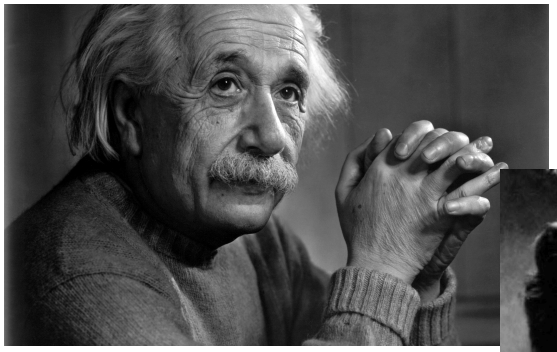
HTTP RESPONSE:

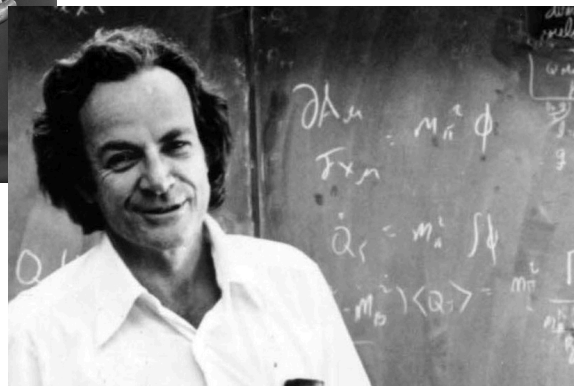
```
HTTP/1.0 200 OK
<HTML> . . . </HTML>
```

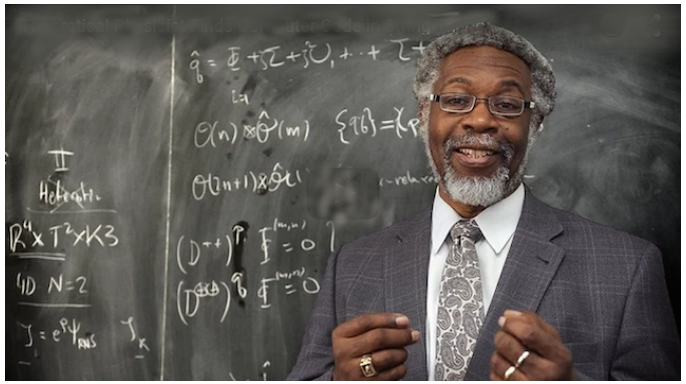
Web security

A historical perspective

- The web is an example of “bolt-on security”
- Originally, the web was invented to allow physicists to share their research papers
  - Only textual web pages + links to other pages; no security model to speak of







The web became complex and adversarial quickly

- Then we added embedded images
  - Crucial decision: a page can embed images loaded from another web server
- Then, Javascript, dynamic HTML, AJAX, CSS, frames, audio, video, …
- Today, a web site is a distributed application
- Attackers have various motivations

Web security is a challenge!

Desirable security goals

- **Integrity:** malicious web sites should not be able to tamper with integrity of my computer or my information on other web sites
- **Confidentiality:** malicious web sites should not be able to learn confidential information from my computer or other web sites
- **Privacy:** malicious web sites should not be able to spy on me or my activities online

Security on the web

- Risk #1: we don't want a malicious site to be able to trash my files/programs on my computer
  - Browsing to `awesomevids.com` (or `evil.com`) should not infect my computer with malware, read or write files on my computer, etc.

Security on the web

- Risk #1: we don't want a malicious site to be able to trash my files/programs on my computer
  - Browsing to `awesomevids.com` (or `evil.com`) should not infect my computer with malware, read or write files on my computer, etc.
- Defense: Javascript is sandboxed;
  try to avoid security bugs in browser code;
  privilege separation; automatic updates; etc.

Security on the web

- Risk #2: we don't want a malicious site to be able to spy on or tamper with my information or interactions with other websites
  - Browsing to `evil.com` should not let `evil.com` spy on my emails in Gmail or buy stuff with my Amazon account

Security on the web

- Risk #2: we don't want a malicious site to be able to spy on or tamper with my information or interactions with other websites
  - Browsing to `evil.com` should not let `evil.com` spy on my emails in Gmail or buy stuff with my Amazon account
- Defense: the same-origin policy
  - A security policy grafted on after-the-fact, and enforced by web browsers

Security on the web

- Risk #3: we want data stored on a web server to be protected from unauthorized access

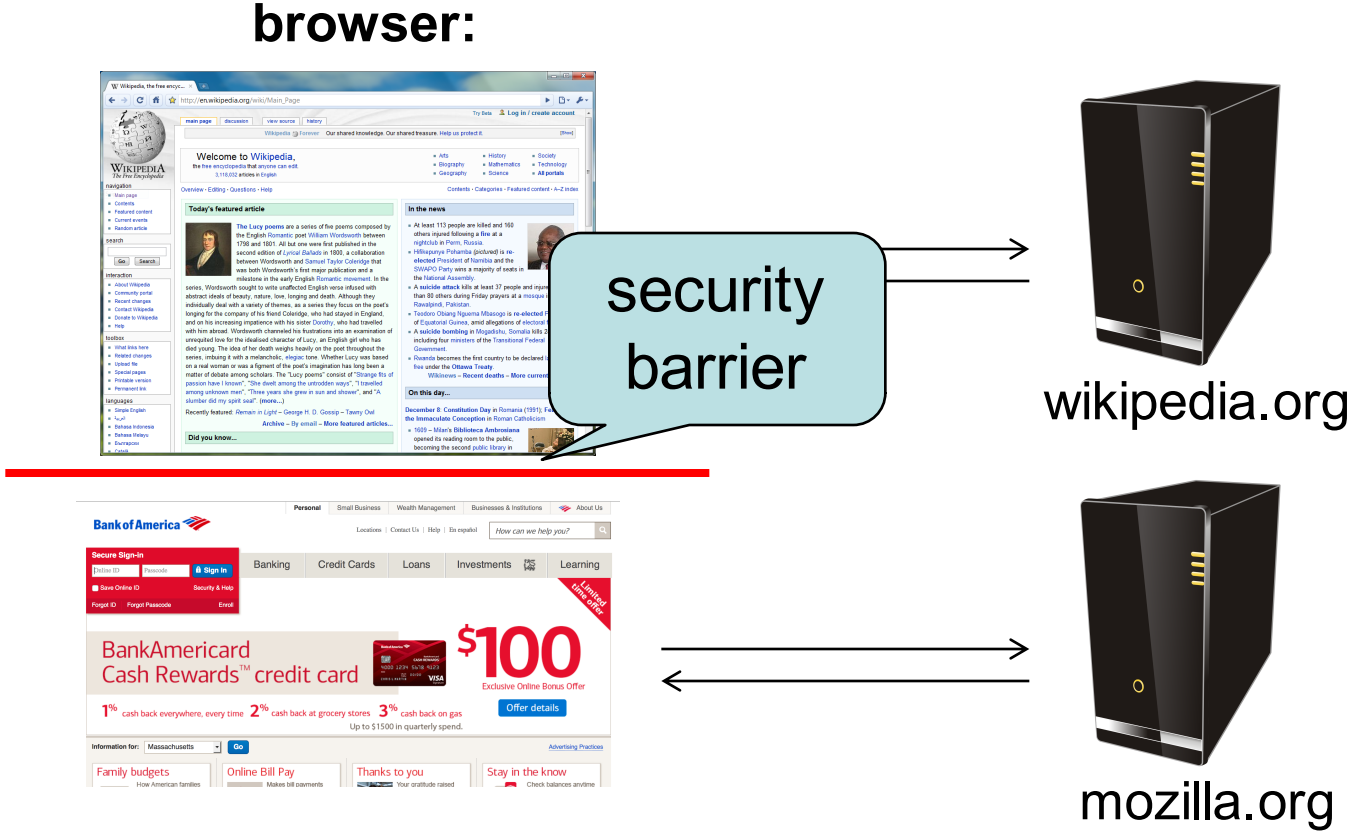
Security on the web

- Risk #3: we want data stored on a web server to be protected from unauthorized access
- Defense: server-side security

Same-origin policy

Same-origin policy

- Each site in the browser is isolated from all others



Same-origin policy

- Multiple pages from the same site are not isolated

browser:

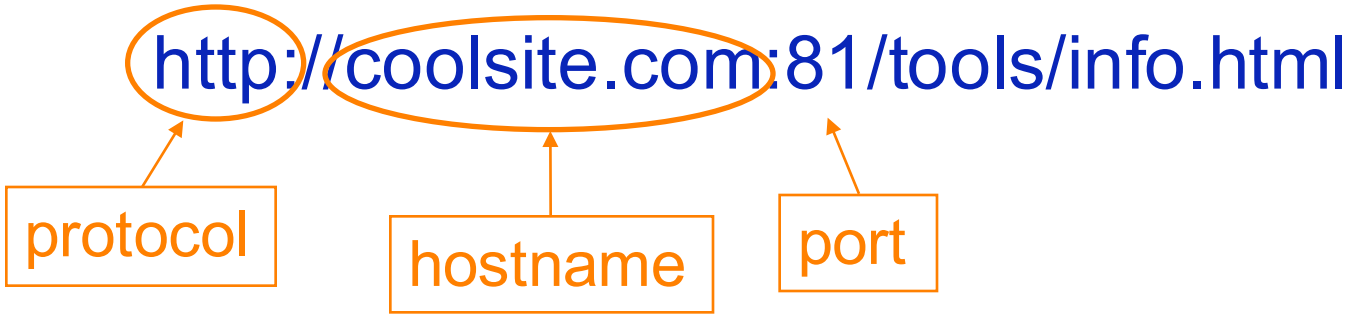
No security barrier

wikipedia.org

wikipedia.org

Origin

- Granularity of protection for same origin policy
- Origin = protocol + hostname + port



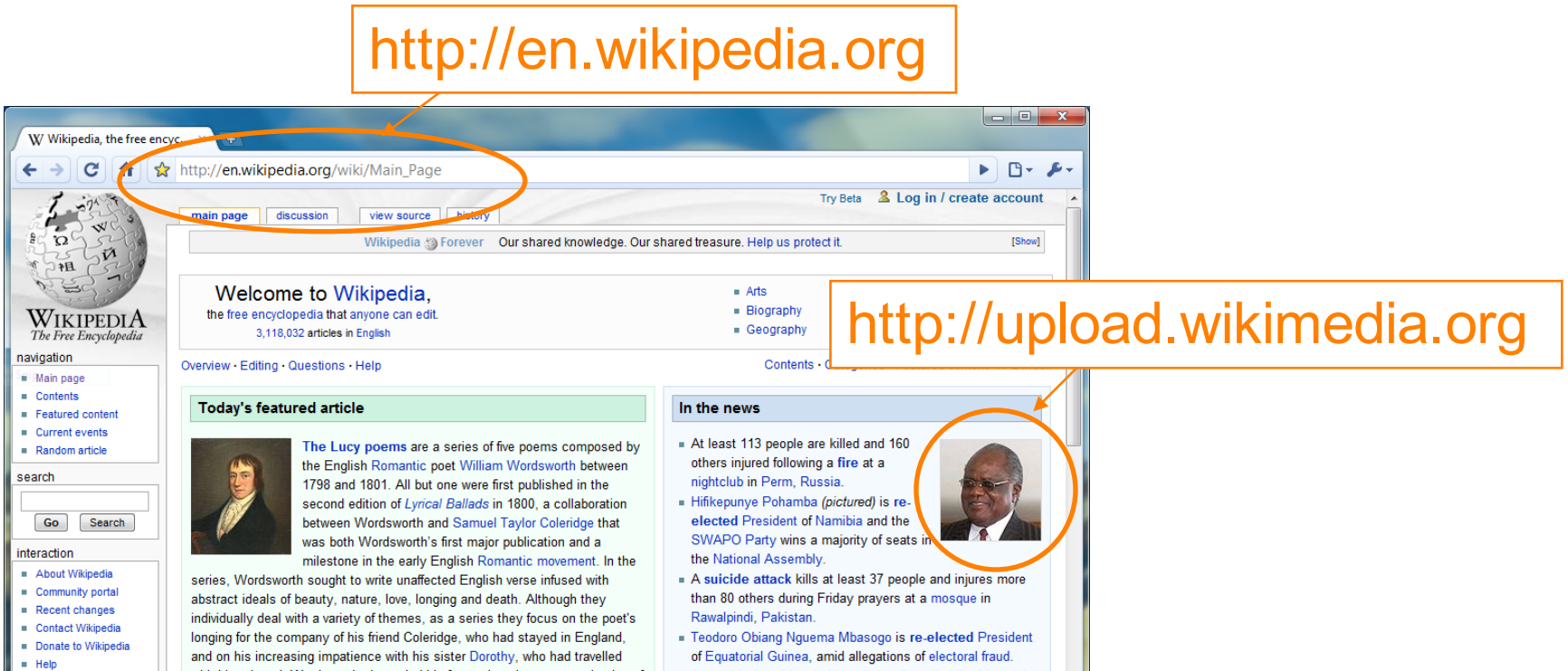
Same-origin policy

One origin should not be able to access the resources of another origin

Javascript on one page cannot read or modify pages from different origins

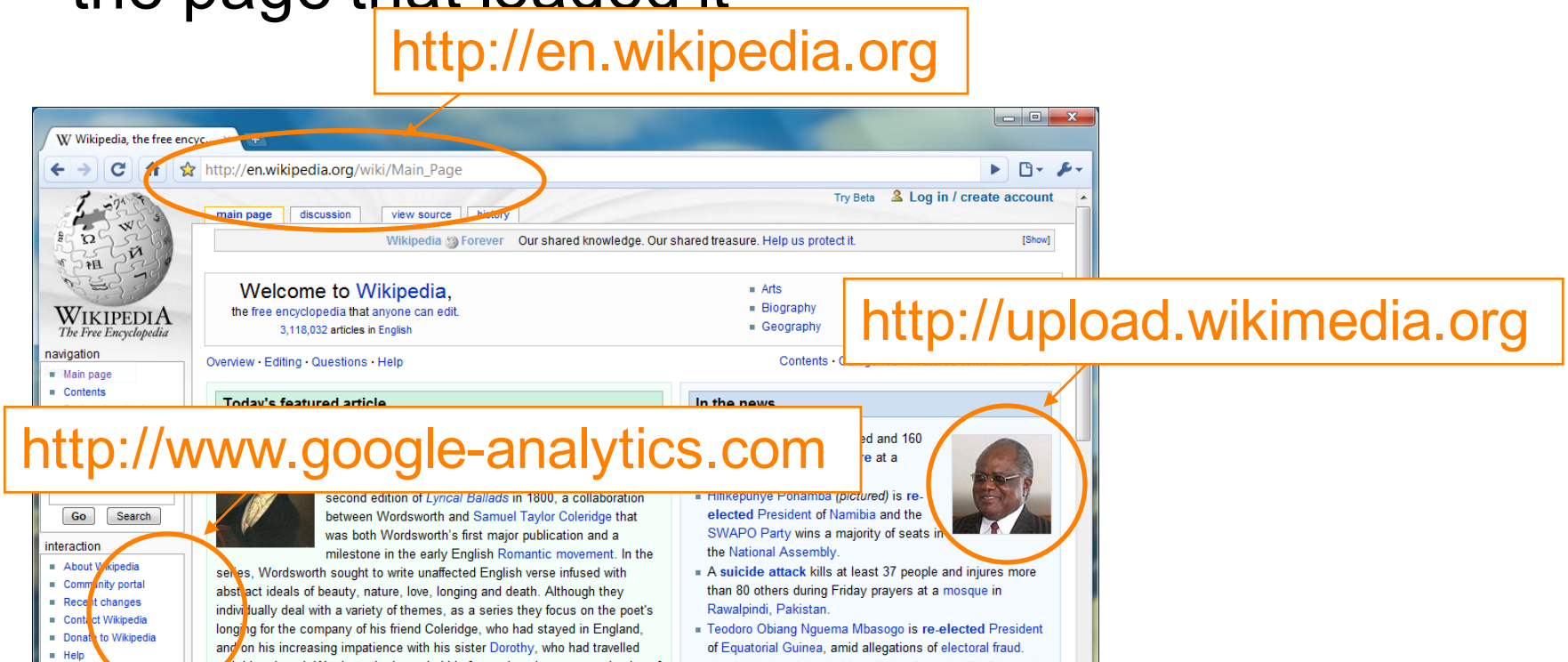
Same-origin policy

- The origin of a page is derived from the URL it was loaded from



Same-origin policy

- The origin of a page is derived from the URL it was loaded from
- Special case: Javascript runs with the origin of the page that loaded it



Exercises

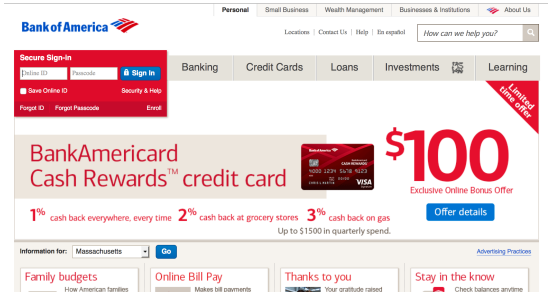
Originating document	Accessed document
http://wikipedia.org/**a**/	http://wikipedia.org/**b**/
http://wikipedia.org/	http://**www.**wikipedia.org/
http://wikipedia.org/	**https**://wikipedia.org/
http://wikipedia.org**:81**/	http://wikipedia.org**:82**/
http://wikipedia.org**:81**/	http://wikipedia.org/

except !!!

Cross-origin communication

- Allowed through a narrow API: **postMessage**
- Receiving origin decides if to accept the message based on origin

```
postMessage
("run this
script",
script)
```



Check origin, and request!

Chromodo
Private Internet Browser

Fast and versatile Internet Browser based on Chromium, with highest levels of speed, security and privacy!

Issue 704: Comodo: Comodo "Chromodo" Browser disables same origin policy, Effectively turning off web security.

13 people starred this issue and may be notified of changes.

tus: Fixed
ner: tav...@google.com
sed: Yesterday
project-...@google.com
dor-Comodo
duct-Chromodo
rity-critical

Project Member Reported by tav...@google.com, Jan 21, 2016

When you install Comodo Internet Security, by default a new browser called Chromodo is installed and set as the default browser. Additionally, all shortcuts are replaced with Chromodo links and all settings, cookies, etc are imported from Chrome. They also hijack DNS settings, among other shady practices.

https://www.comodo.com/home/browsers-toolbars/chromodo-private-internet-browser.php

Chromodo is described as "highest levels of speed, security and privacy", but actually disables all web security. Let me repeat that, they ***disable the same origin policy***.... ?!?..

Coming up:

attacks on web servers!