Computer Security

12a. More Web Security

Paul Krzyzanowski

Rutgers University

Spring 2017

HTML image tags

```
<img src="http://pk.org/images/balloons.jpg" height="300" width="400"/>
```

- Images are static content with no authority
- Any problems with images?

HTML image tags

```
<img src="http://evil.com/images/balloons.jpg?extra_information"
height="300" width="400"/>
```

- URL may pass arguments
  - Communicate with other sites
- Hide resulting image
  -

Common way for a sender to force HTML-formatted email to provide read notifications

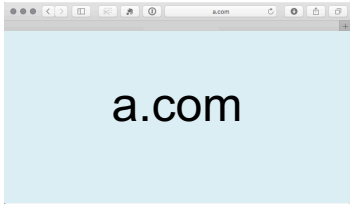
- Social engineering: add logos to fool a user

Frames and iFrames

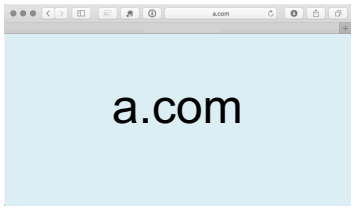
- Browser window may contain frames from different sources
  - Frame = rigid division as part of frameset
  - iFrame = floating inline frame
- Why use them?
  - Delegate screen area to content from another source
  - Browser provides isolation based on frames
  - Parent can continue to function even if frame is broken

Web security policy goals

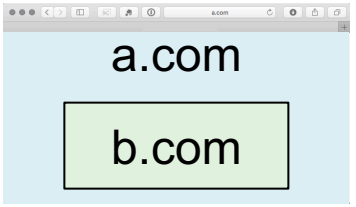
- Safe to visit an evil web site



- Safe to visit two pages at one time
  - Address bar distinguishes them



- Allow safe delegation
  - Frame inside a frame
  - Each frame = origin of the content within it



Same-origin Policy

- Web application security model: **same-origin policy**
- A browser permits scripts in one page to access data in a second page ***only if*** both pages have the same origin
- Origin = { URI scheme, hostname, port number }
- Same origin
  - http://www.poopybrain.com/419/test.html
  - http://www.poopybrain.com/index.html
- Different origin
  - https://www.poopybrain.com/index.html – different URI scheme (https)
  - http://www.poopybrain.com:8080/index.html – different port
  - http://poopybrain.com/index.html – different host

Ideas behind the same-origin policy

- Each origin has client-side resources
  - Cookies: simple way to implement state
    - Browser sends cookies associated with the origin
  - JavaScript namespace: functions & variables
  - DOM storage: key-value storage per origin
  - DOM tree: JavaScript version of the HTML structure
- Each frame gets the origin of its URL
- JavaScript code executes with the authority of its frame's origin
  - If cnn.com loads JavaScript from jQuery.com, the script runs with the authority of cnn.com
- Passive content (CSS files, images) has *no* authority
  - It doesn't (and shouldn't) contain executable code

Mixed content: http & https

- HTTPS page may contain http content:
  `<script src="http://www.mysite.com/script.js"> </script>`
  - Active network attacker can now hijack the session

- Safer approach
  `<script src="//www.mysite.com/script.js"> </script>`
  - Served over the same protocol as the embedding page (frame)

- Some browsers warn you of mixed content
  - Some warning may be unclear to the user

Extended Validation Certificates

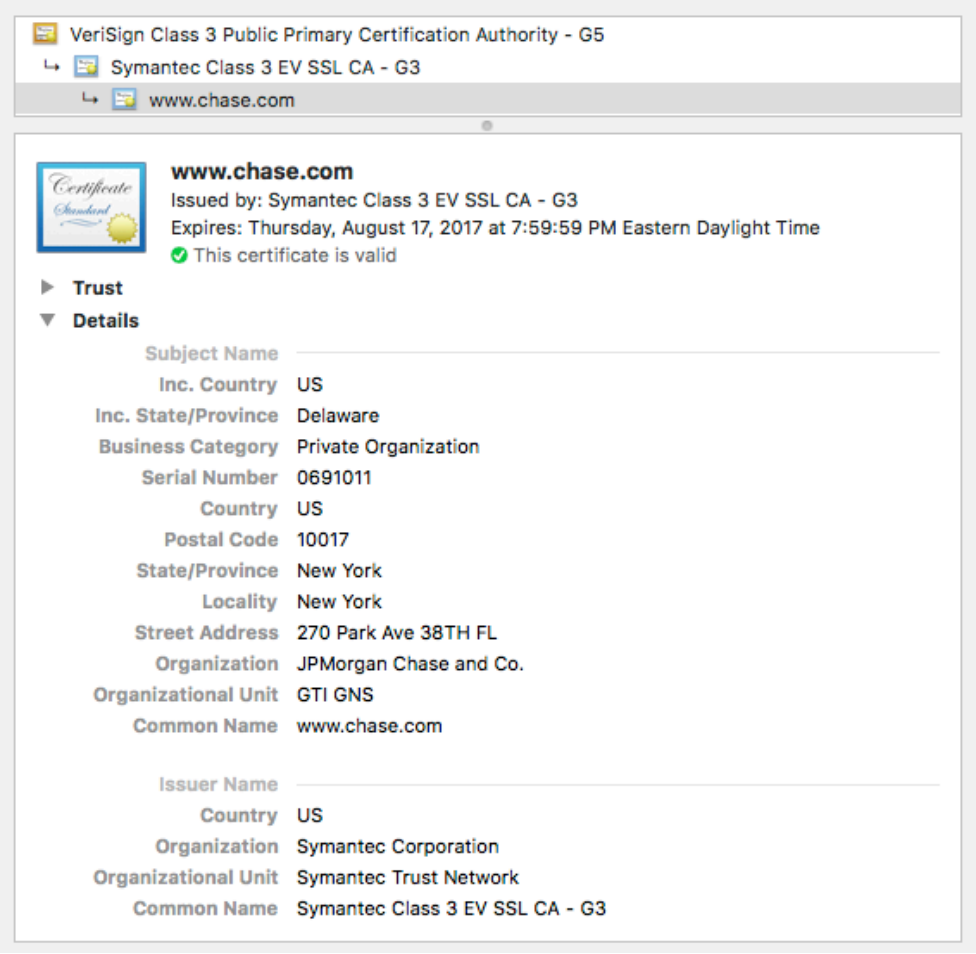
For SSL/TLS authentication to be meaningful, the server's X.509 certificate must belong to the party the user believes it belongs to

- Domain validated certificates
  - Only require proof of domain control
  - Do not prove that a legal entity has a relationship with the domain
- Extended validation (EV) certificates
  - Belong to the legal entity controlling the domain (or software)
  - Certificate Authority must validate the entity's identity
    - More stringent validation: check company incorporation, domain registration, position of applicant, etc.

Extended Validation Certificates

EV certificate will contain

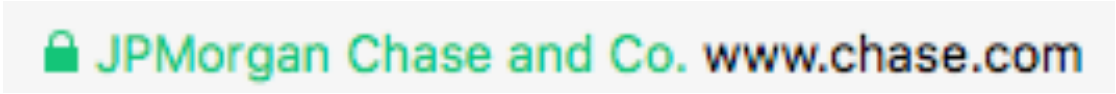
- Government-registered serial number
- Physical address
- + the usual stuff: name, location, issuer, …



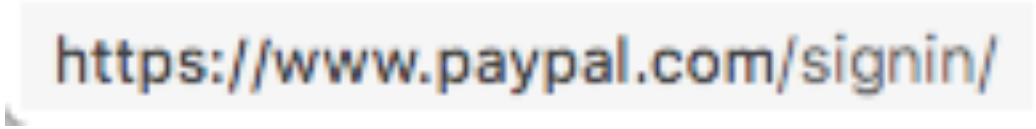
Extended Validation Certificates

- Browsers would show a lock icon for *any* SSL/TLS connection

- This led to a false sense of security
  - Fraud sites would use TLS to let users think they are legitimate
- Modern browsers
  - Identify & validate EV certificates
  - Present a security indicator that identifies the certificate owner



Status Bar



Trivial to spoof with JavaScript & the onclick attribute

```
<a href="http://www.paypal.com/signin"
        onclick="this.href = 'http://www.evil.com/';">
        PayPal</a>
```

The end