

Web Security Overview

Cloud Computing and SaaS



Announcements

- Checkpoint 2 due Friday 10/23
 - Authorization Microservice
 - Dockerfile!
 - Will be using this to run your code in AG
 - Minor fixes, changes pushed to skeleton (make sure to pull!)
 - Latest update to skeleton at ~6:15pm
- AWS Educate
 - Do not create any services! We can't guarantee extra credits

Last Time

- Codebase Walkthrough
 - Different microservices
 - Authorization, Posts, etc.
- Microservices

Today

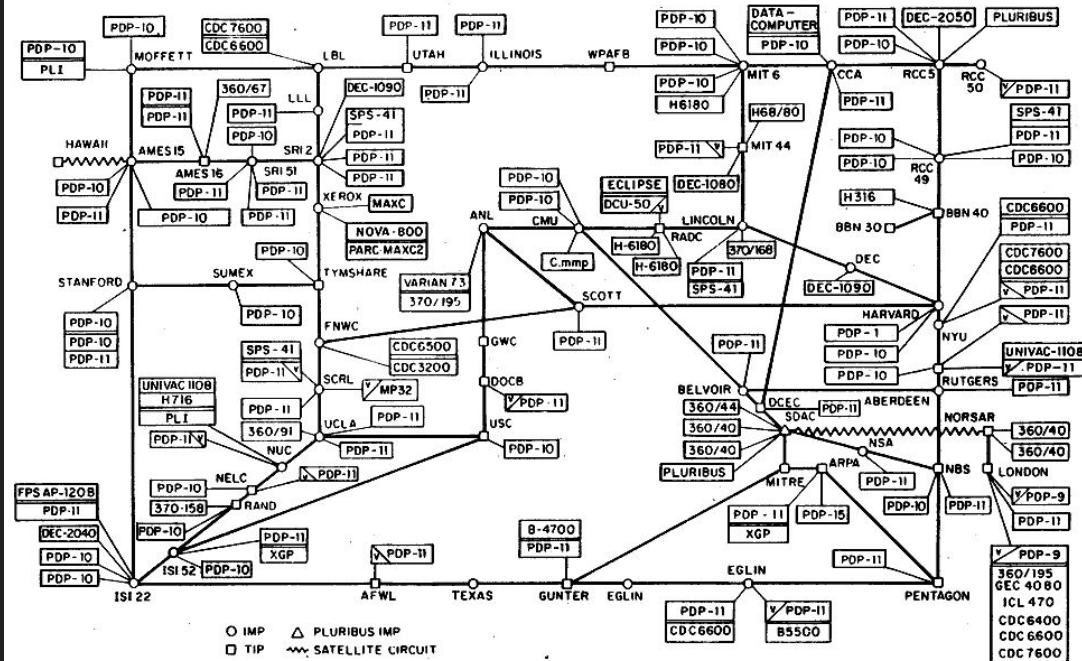
- Introduction to Web Security
 - Beginning of the internet
 - Why it's actually a complete mess
 - Web Pages
 - HTML, CSS, Javascript(!)
 - Cookies
 - Frames
 - Same Origin Policy

Beginning of the Internet

- Problem: Need computers to communicate
- Solution: Create the concept of a “network”
- First workable prototype: ARPANET, funded by US DoD
 - Advanced Research Projects Agency Network
 - Allowed multiple computers to communicate on a single network
 - First message (“LOGIN”) sent from UCLA to Stanford, ended up crashing the network
 - Stanford received only the first two letters

ARPANET Logical Map

ARPANET LOGICAL MAP, MARCH 1977



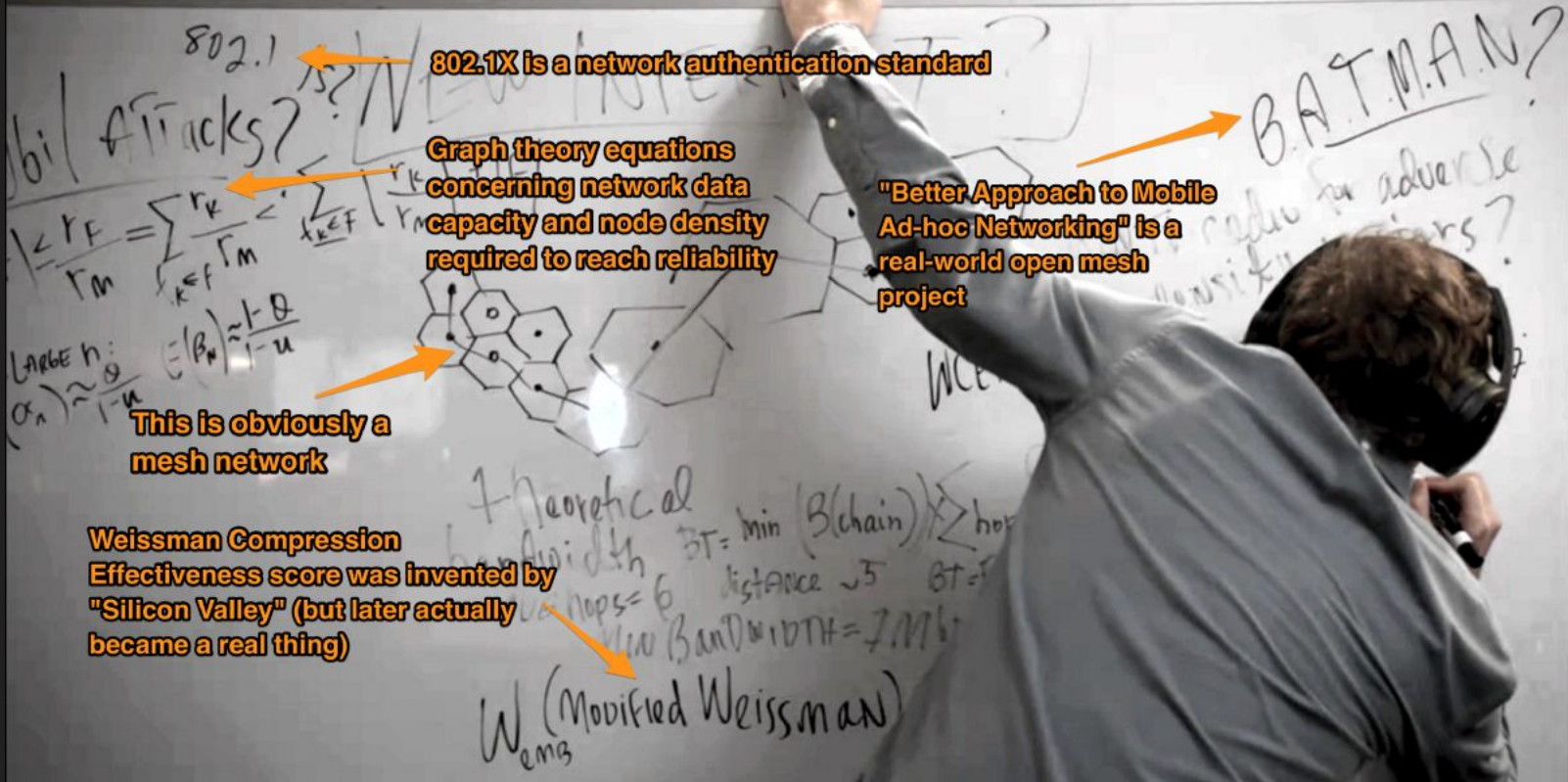
(PLEASE NOTE THAT WHILE THIS MAP SHOWS THE MOST POPULATION OF THE NETWORK ACCORDING TO THE BEST INFORMATION OBTAINABLE, NO CLAIM CAN BE MADE FOR ITS ACCURACY)

NAMES SHOWN ARE IMP NAMES, NOT (NECESSARILY) HOST NAMES


Why Internet Security is a Mess

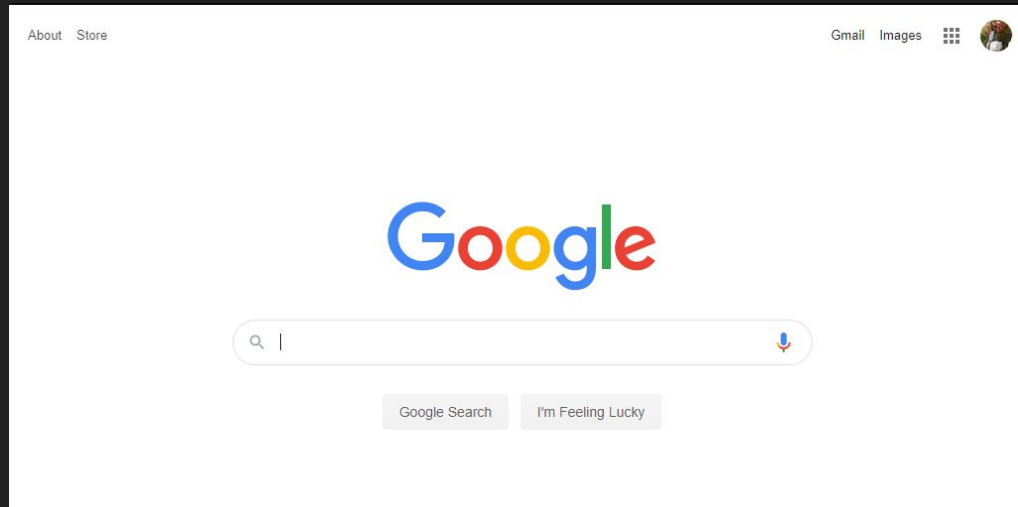
- Most important principle of security: build security in from the start
- Problem: Internet was meant to only share research papers
 - Not many security concerns here
 - Very few people had access anyway
- Everything since then has been a hack/patch*
 - Maybe it's time for a new internet?

Why Internet Security is a Mess



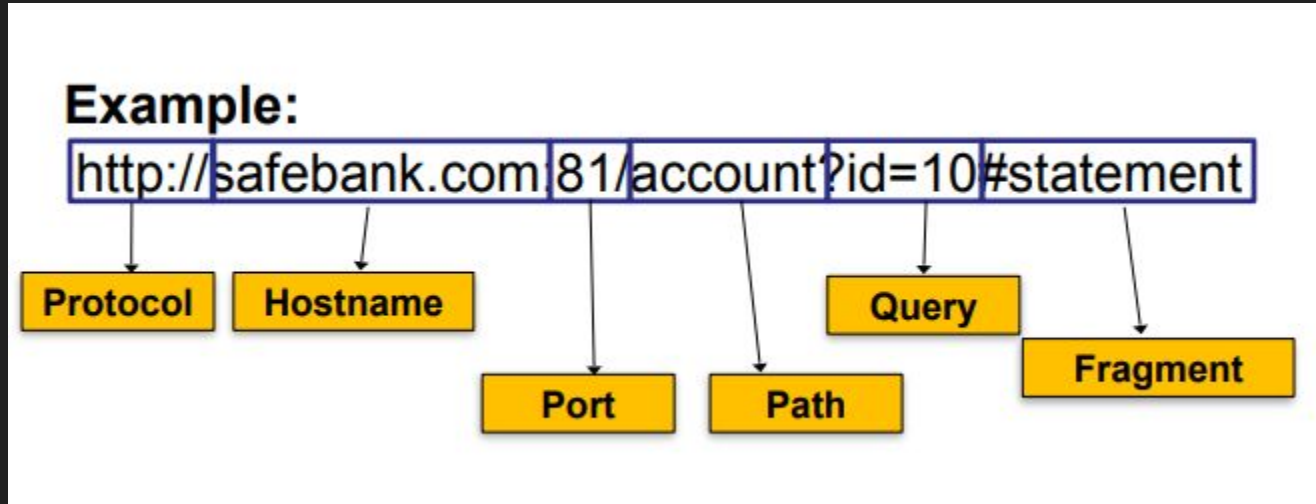
Web Pages

 <https://www.google.com>



Web Page URLs

- URL: Uniform Resource Locator



Recall: HTTP Requests



Web Page Components

- HTML
 - Hypertext Markup Language
 - “LaTeX for web pages”
- CSS
 - Cascading Style Sheets
 - Formats HTML content, Aesthetics
- Javascript
 - Scripting language that can read and modify a webpage and its elements

Web Pages: HTML

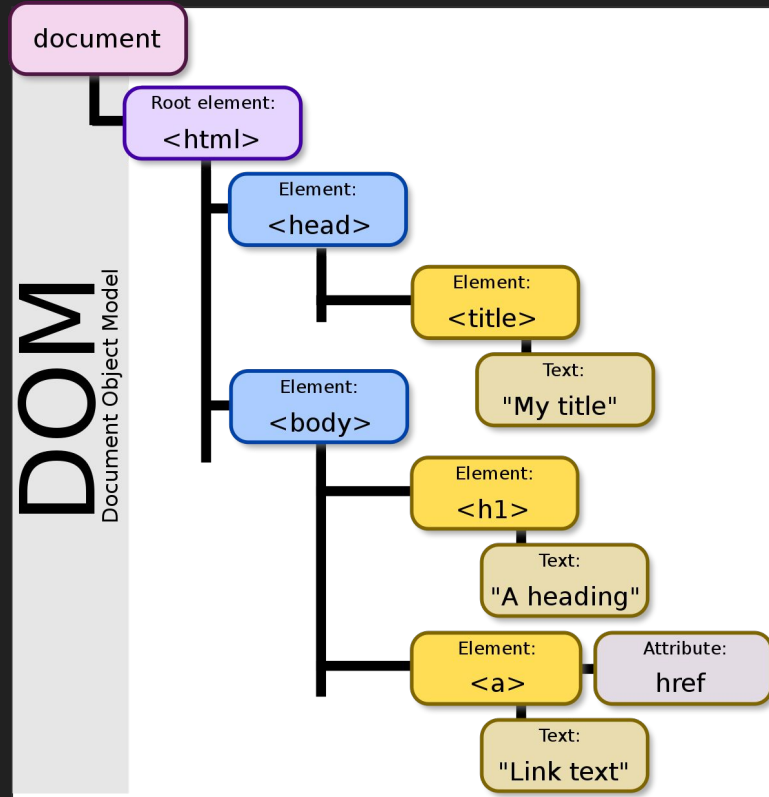
```
<!DOCTYPE html>
<html>

  <head>
    <title>My First Webpage</title>
  </head>

  <body>
    <h1>
      My First Webpage
    </h1>
    <p>This is a paragraph...</p>
  </body>

</html>
```

Document Object Model (DOM)



Javascript

- Concerns:
 - Can change images, hide/unhide elements, change cursor
- Problem: Javascript can modify pretty much anything
- More Problems:
 - Cookies, can read and modify values
 - Javascript can make HTTP requests
 - What if they send your access_token to themselves and then delete it?

Javascript, Cookies

- Cookies
 - Reading
 - `var x = document.cookie;`
 - Writing
 - `document.cookie = "username=DeCal;
expires=Tue, 20 Oct 2020 12:00:00 UTC;
path=/";`

Javascript - Quick Demo

Frames

- Enable embedding web pages within web pages

The image shows a side-by-side comparison of code and its rendered output. On the left, the Brackets code editor displays the HTML code for an index.html file. The code includes a DOCTYPE declaration, HTML lang attribute, charset and viewport meta tags, a link to a style.css file, and a title 'Form'. The body contains a 'wrap' class with several h2 elements: 'Embeds', 'iFrame', 'Audio', and 'Video'. The 'iFrame' section is highlighted with a red box, showing the following code:

```
15 <h2>iFrame</h2>
16 <iframe src="https://gmail.com"
17 frameborder="2"></iframe>
```

On the right, a browser window displays the rendered page. The page has a purple background and contains the following sections: 'Embeds', 'iFrame', 'Audio', and 'Video'. The 'iFrame' section contains a screenshot of a Gmail page with a 'Google' logo, 'sign in' link, and 'Create an account' button. A red arrow points to the iFrame, and a red box highlights the text: 'Also notice our iFrame has a border of 2px around it'.

Frames

- Problem: What if Javascript in one frame accesses the information of another?
- Solution: Browsers enforce frame isolation

Same-Origin Policy

- Big Idea: One origin shouldn't be able to read or modify the resources of another origin
- Different web pages are isolated from each other
- Different frames in a web page are isolated from each other
- What defines origin?
 - Protocol, Hostname, Port
 - String matching, nothing more

Same-Origin Policy Practice

- 1

- <http://calcloud.org/lec1>
- <http://calcloud.org/lec2>

- 2

- <http://calcloud.org:80/lec1>
- <http://api.calcloud.org:80/lec2>

- 3

- <https://calcloud.org:80/lec1>
- <http://api.calcloud.org:81/lec2>

- 4

- <https://calcloud.org:80/lec1>
- <https://calcloud.org:80/lec2>