Lecture 6 – Intro to Web

University of Illinois

ECE 422/CS 461

Announcements

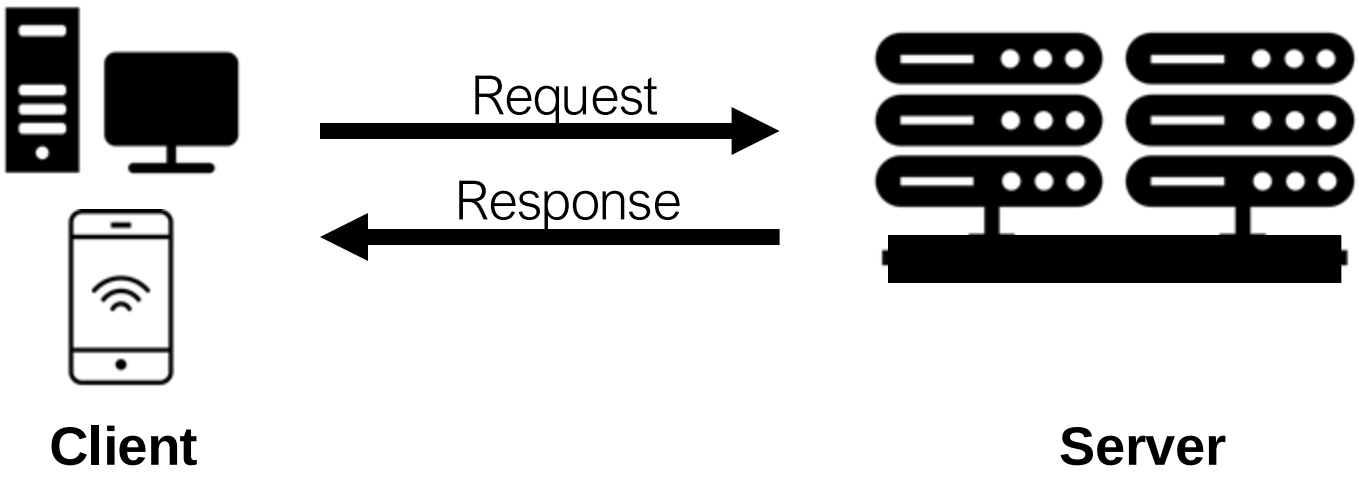
- AppSec CP2 due Thursday 6pm
- Quiz 3 due Friday
  - Last question is covered today. If you have already completed Quiz 3 and lost a point, contact instructor

Goals

- By the end of this lecture you should:
  - Know the basics of web
    - HTTP, HTML, JavaScript, and cookies
  - Understand the threat model of web
  - Understand and apply the same-origin policy

Web's Client-Server Model

- **Web:** application layer on top of TCP/UDP



HTTP and URL

- **H**yper**T**ext **T**ransfer **P**rotocol: Protocol for request and response between client and server

- **U**niform **R**esource **L**ocator

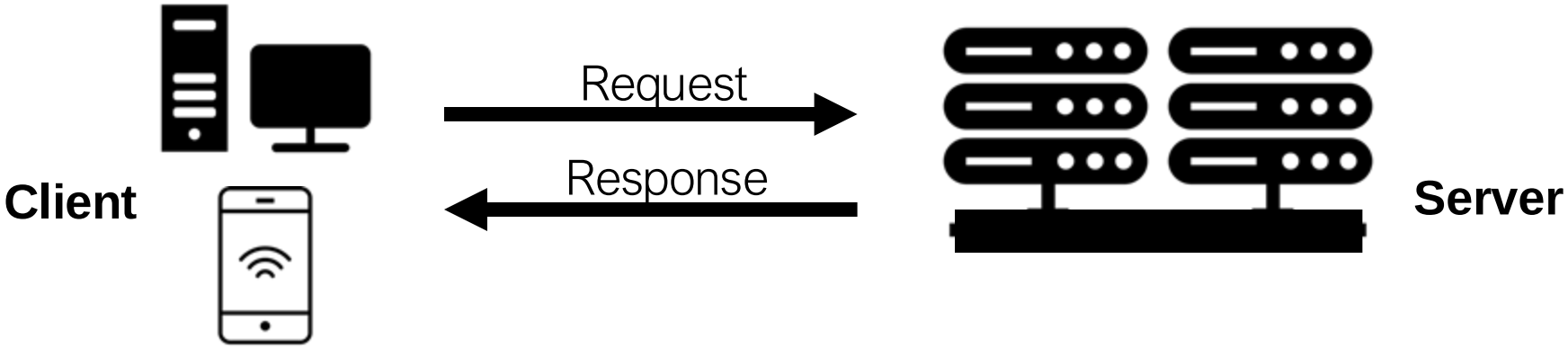
http://www.example.com/home/somepage

↓ Browser generates HTTP request

```
GET /home/somepage HTTP/1.1
Host: www.example.com
Accept: text/html
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows 98)
……
```

HyperText Transfer Protocol (HTTP)

```
GET /home/somepage HTTP/1.1
Host: www.example.com
Accept: text/html
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows 98)
……
```




```
HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8
Expires: Tue, 22 Sep 2020 14:33:51 GMT
......

Page content goes here
```

HTTP Requests

```
GET /home/somepage HTTP/1.1
Host: www.example.com
Accept: text/html
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows 98)
……
```

- Method (e.g., GET)
- Path (/…)
- Protocol version (e.g., HTTP)
- Host
- Various metadata, also called headers

HTTP Methods

```
GET /home/somepage HTTP/1.1
Host: www.example.com
Accept: text/html
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows 98)
……
```

- GET: request data
  - GET /videos/32410 (download a video)
- POST: submit data
  - POST /login … (submit username & pw)
- PUT, DELETE, …

HTTP Responses

- Contain protocol version, status code, various metadata, and the requested resource

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8
Expires: Tue, 22 Sep 2020 14:33:51 GMT
......

Page content goes here
```

HTTP Status Code

- 1xx: informational
- 2xx: success (200 OK)
- 3xx: redirections
  - 301 Moved Permanently
- 4xx: client errors
  - 403 Forbidden, 404 Not Found
- 5xx: server errors
  - 500 Internal Server Error

Hyper**T**ext **M**arkup **L**anguage (HTML)

- Most frequently requested resource are webpages written in HTML

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8
Expires: Tue, 22 Sep 2020 14:33:51 GMT
......

<!doctype html>
<html>
<head>
   <title>Example Page</title>
</head>
<body>
...
```

History of Web/HTTP/HTML

- Invented by Tim Berners Lee starting 1989

History of Web/HTTP/HTML

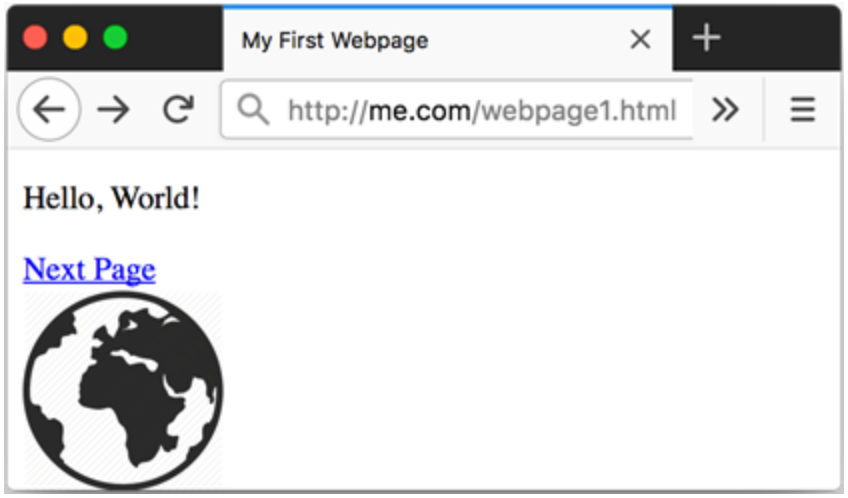
- Initially to share info between physicists
  - Only supported text pages and links
- … then embedded images & contents from multiple servers loaded on a single client
- … then dynamic elements using JavaScript
- … then hardware access (e.g., files, cameras)

- No initial security considerations -- bolted on!

Hyper**T**ext **M**arkup **L**anguage (HTML)

- Specify web page content and layout

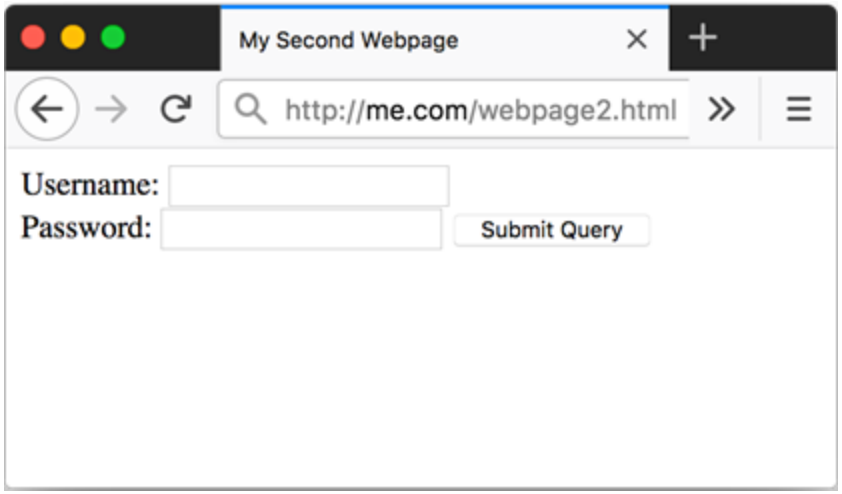
```html
<!DOCTYPE html>
<html>
    <head>
        <title>My First Webpage</title>
    </head>
    <body>
        <p>Hello, World!</p>
        <a href="/webpage2.html">Next Page</a>
        <br>
        <img src="http://wiki.com/Earth_globe.png">
    </body>
</html>
```



Hyper**T**ext **M**arkup **L**anguage (HTML)

- Specify web page content and layout

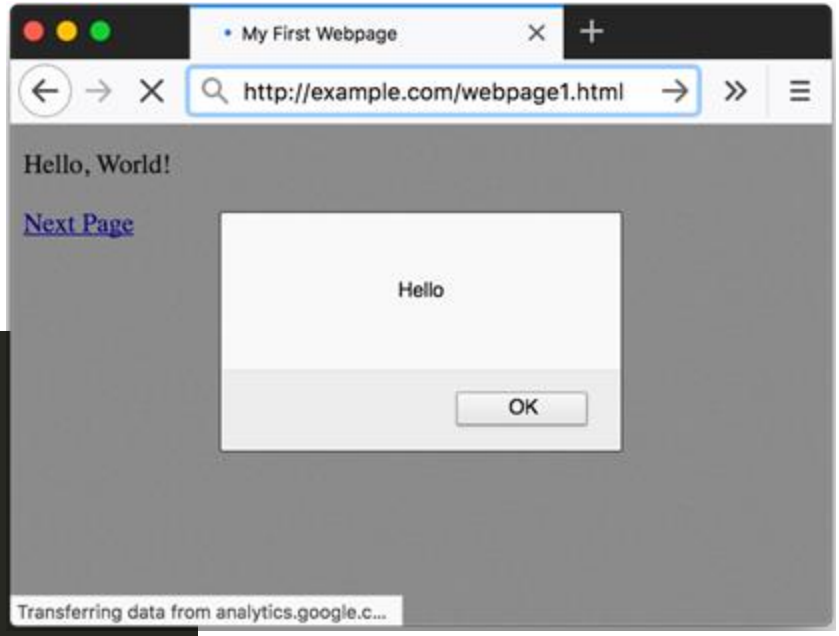
```html
<!DOCTYPE html>
<html>
    <head>
        <title>My Second Webpage</title>
    </head>
    <body>
        <form action="/login" method="POST">
            Username:
            <input type="text" name="username">
            <br>
            Password:
            <input type="password" name="password">
            <input type="submit" name="submit">
        </form>
    </body>
</html>
```



Dynamic HTML using JavaScript

- Usually included with <script> … </script> HTML tags

```
<!DOCTYPE html>
<html>
    <head>
        <title>My First Webpage</title>
    </head>
    <body>
        <p>Hello, World!</p>
        <a href="/webpage2.html">Next Page</a>
    </body>
    <script type="text/javascript">
            alert("Hello");
    </script>
    <script type="text/javascript" src="http://analytics.google.com/">
    </script>
</html>
```



JavaScript

- Can be included in a few ways:
  - Most commonly, <script> … </script> tag
  - Import from file (from same or another website)
    - <script src="http://analytics.google.com/api/js">
  - Event handlers
    - <img src="pic.gif" onMouseOver="alert('leave the picture alone!')">

JavaScript

- Powerful programming language that can
  - Access and alter page contents
    - Not only its own page, but also another page it opens in a new tab/window (with restrictions)
  - Track events (mouse click, motion, keystrokes)
  - Access hardware (filesystem, camera, location, …)
  - Read and set cookies
  - Make HTTP requests
  - Open new page: window.open(“http://illinois.edu”);

Cookies

- A way for websites to store states on clients
  - Browser maintains all cookies it receives (cookie-jar)
  - Browser **automatically** attaches all cookies **in scope** in subsequent HTTP requests to the website

Cookies

- A way for websites to store states on clients
  - Browser maintains all cookies it receives (cookie-jar)
  - Browser **automatically** attaches all cookies **in scope** in subsequent HTTP requests to the website
  - Many uses: login info, user profile, preferences, shopping cart, analytics, …

Cookies

H1Y3fHC7I69v9W3oC2	Guest

GET /

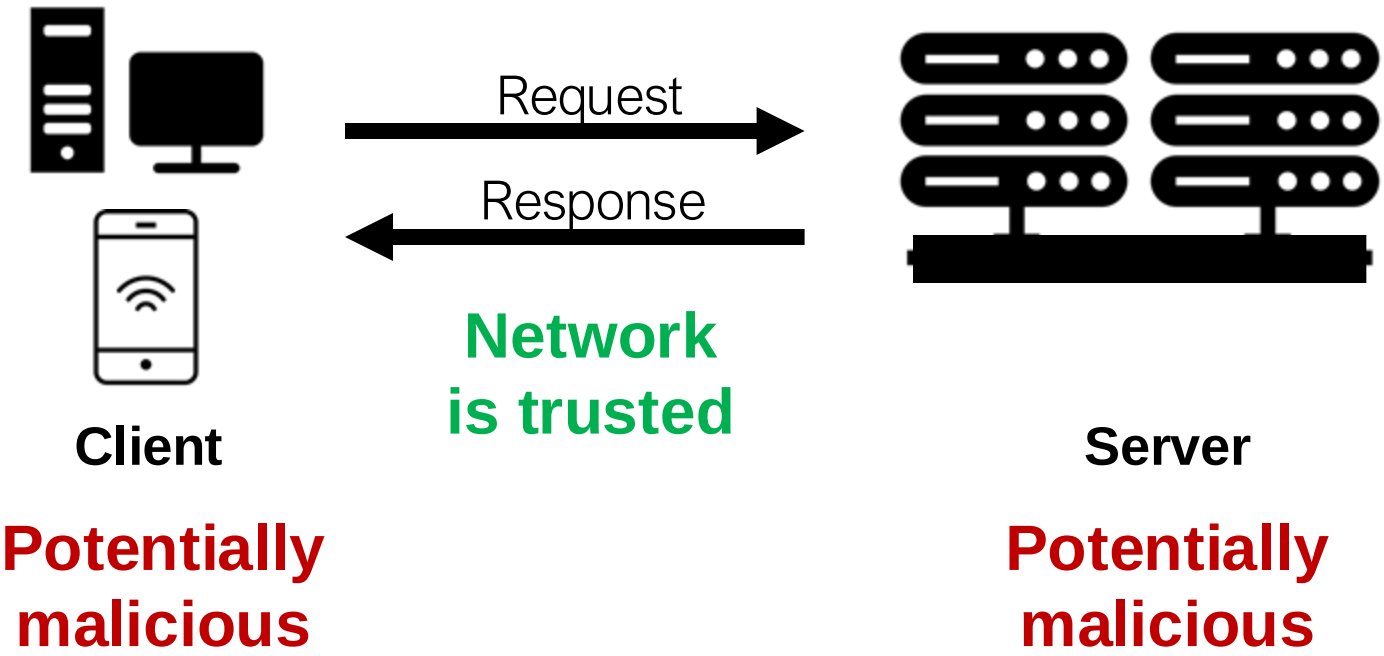
HTTP response:
Set-cookie: sessionID =
H1Y3fHC7I69v9W3oC2

GET /item123
Cookie: sessionID = H1Y3fHC7I69v9W3oC2

POST /cart
Cookie: sessionID = H1Y3fHC7I69v9W3oC2

Web Threat Model

- Trusted network, malicious client *or* server



Web Threat Model

- Server may be malicious
  - Install malware, show fake info, steal user data, track user action on another (benign) server, …
- Client may be malicious
  - Take control of server, steal data from server, attack other clients of the server, …

Web Threat Model

- Trusted network
  - Trustworthiness of the network is a topic for later

- Some non-goals (of web security):
  - A network attacker who eavesdrops or modifies communication between client and server
  - Distributed denial of service (network attack)
  - A malicious or phishing website that scams clients

A Motivating Question

- If you are logged into your bank in one tab, is it safe to visit another (potentially malicious) website at the same time?

JavaScript Example Scenario?

- JavaScript of foo.com can
  - Open a new tab in the browser and open bar.com in the new tab (by making a HTTP GET request)
  - Modify contents of bar.com shown to user?
  - Track how the user interacts with bar.com?
  - Read cookie set by bar.com?
  - Make HTTP requests to bar.com on user's behalf?

Same-Origin Policy (SOP)

- The previous scenario is *very* problematic
- Browser implements strict isolation mechanisms
  - JavaScript of a website can only access webpages belonging to the same website, i.e., **same origin**

Uniform Resource Locator (URL)

- Query and fragment are optional
- Port may be omitted, each scheme has a default
  - E.g., default port for http is 80, for https is 443

http://www.example.com/home/somewebpage

Scheme Host Path

https://courses.engr.illinois.edu:80/cs461/fa2024?user=admin#grading

Scheme Host Port Path Query Fragment

Same-Origin Policy (SOP)

- A **web origin** = (scheme, host, port)
  - Simple string match
  - Case insensitive

*Internet Explorer (IE) defines origin as (scheme, host)

https://courses.engr.illinois.edu:80/cs461/fa2024?user=admin#grading

Scheme **Host** **Port** **Path** **Query** **Fragment**

SOP Exercises

https://example.com/index.html?redirect=www.example.com#h2

URL	Origin	Same?
http://example.com?q=search		
https://www.example.com/index.html		
https://example.com:80#fb.com		
https://example.COM:443/login/check.php?%20		
HttPs://fb.com/example.com#@illinois.edu		

Summary

- Web basics: HTTP, HTML, JavaScript, cookies
- Threat model: either client or server can be malicious (network is trusted)
- Same-origin policy is used by browsers to isolate websites; origin = (scheme, host, port)