CS5331: Web Security of things

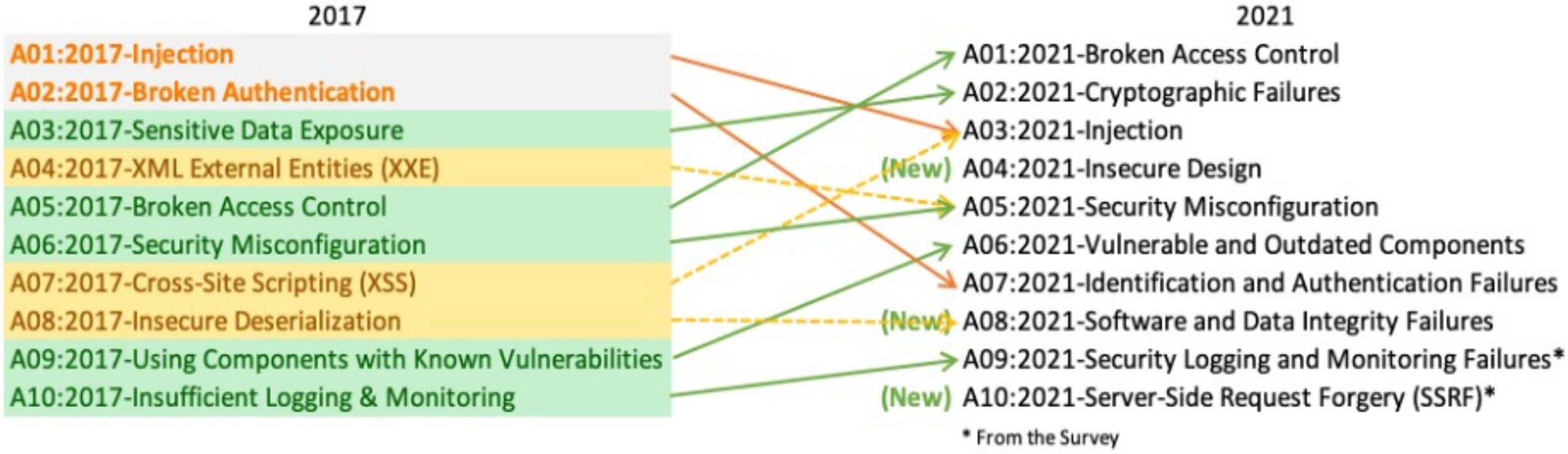
Lecture 1: Overview

The (In)Security of Web

What are the recent (web) security incidents in news?

OWASP Top 10

OWASP Top 10 - 2013	→	OWASP Top 10 - 2017
A1 – Injection	→	A1:2017-Injection

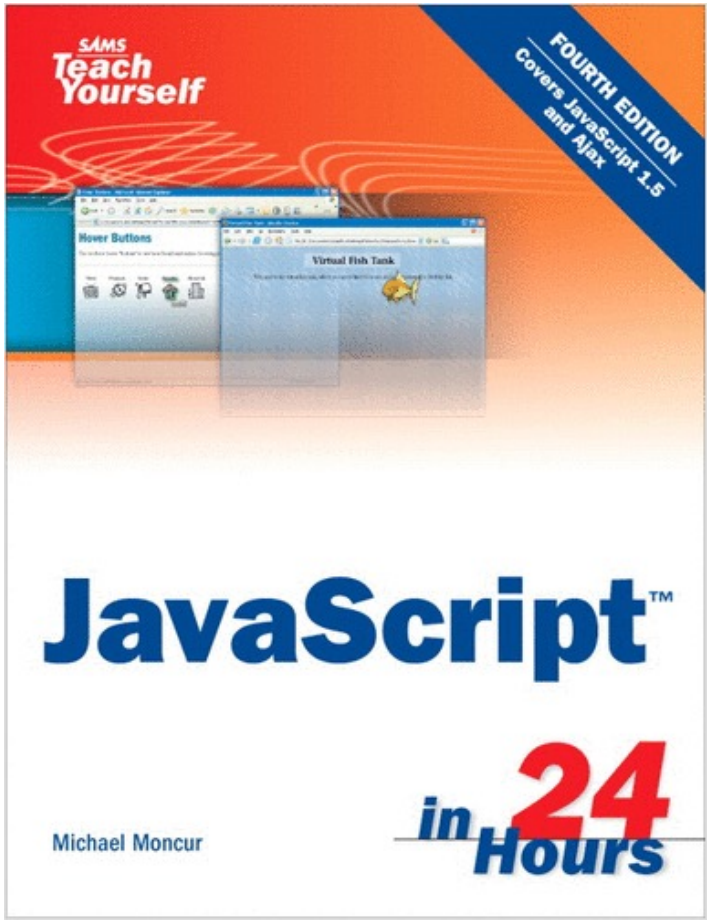


A10 – Unvalidated Redirects and Forwards	☒	A10:2017-Insufficient Logging&Monitoring [NEW,Comm.]

Why Does This Happen?

- Functionality: the primary concern during design and implementation.
  - Security is the secondary goal
  - Unawareness of security problems
- Unavoidable human mistakes
  - Awareness
  - Lazy programmer
- Complex modern computing systems

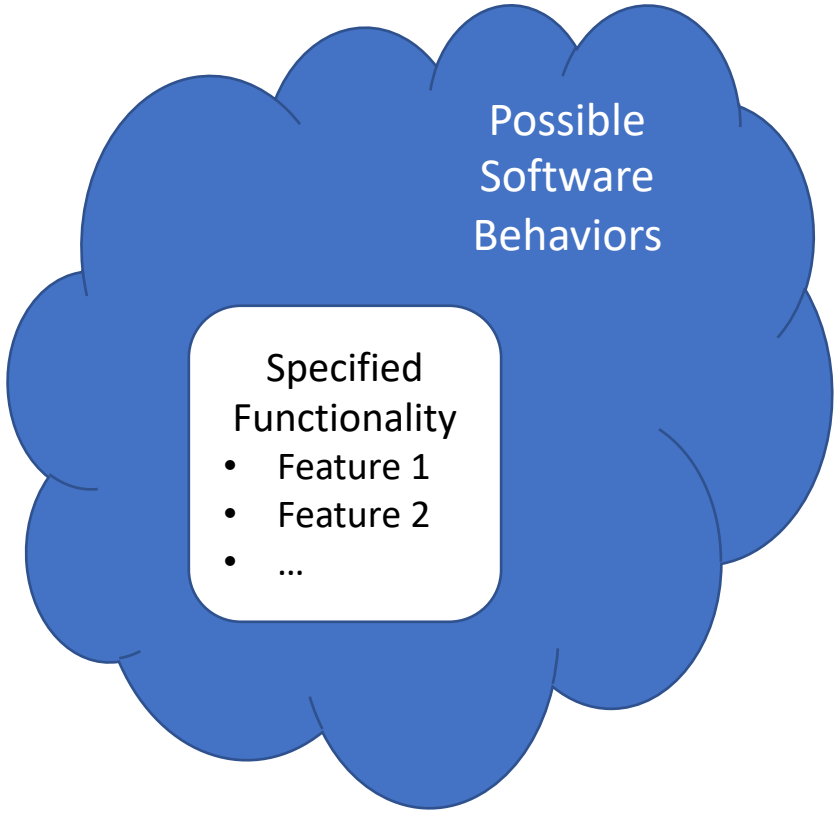
Impatient Programmers



- Maybe enough for learning basic functionality
- Never enough for to learn subtle implications of functionalities
- Result: programs can do more than you expect

Functionality, Security, and Trust

- Security is about "nothing else"
  - Specified functionality and **only** specified functionality
- Trust for functionality vs. Trust for security
  - E.g., trusting CPU for computation and for security enclave



Principle of Easiest Penetration

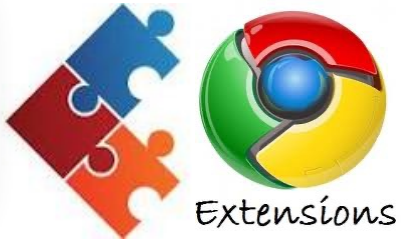
- Security is about every aspect of a computing system
  - Hardware, software, data, and people.
- Principle of easiest penetration:
  - Any system is most vulnerable at its *weakest point*.
  - Attackers don't follow any rules. Don't underestimate their creativity.


Master

The Web Platform

Browser

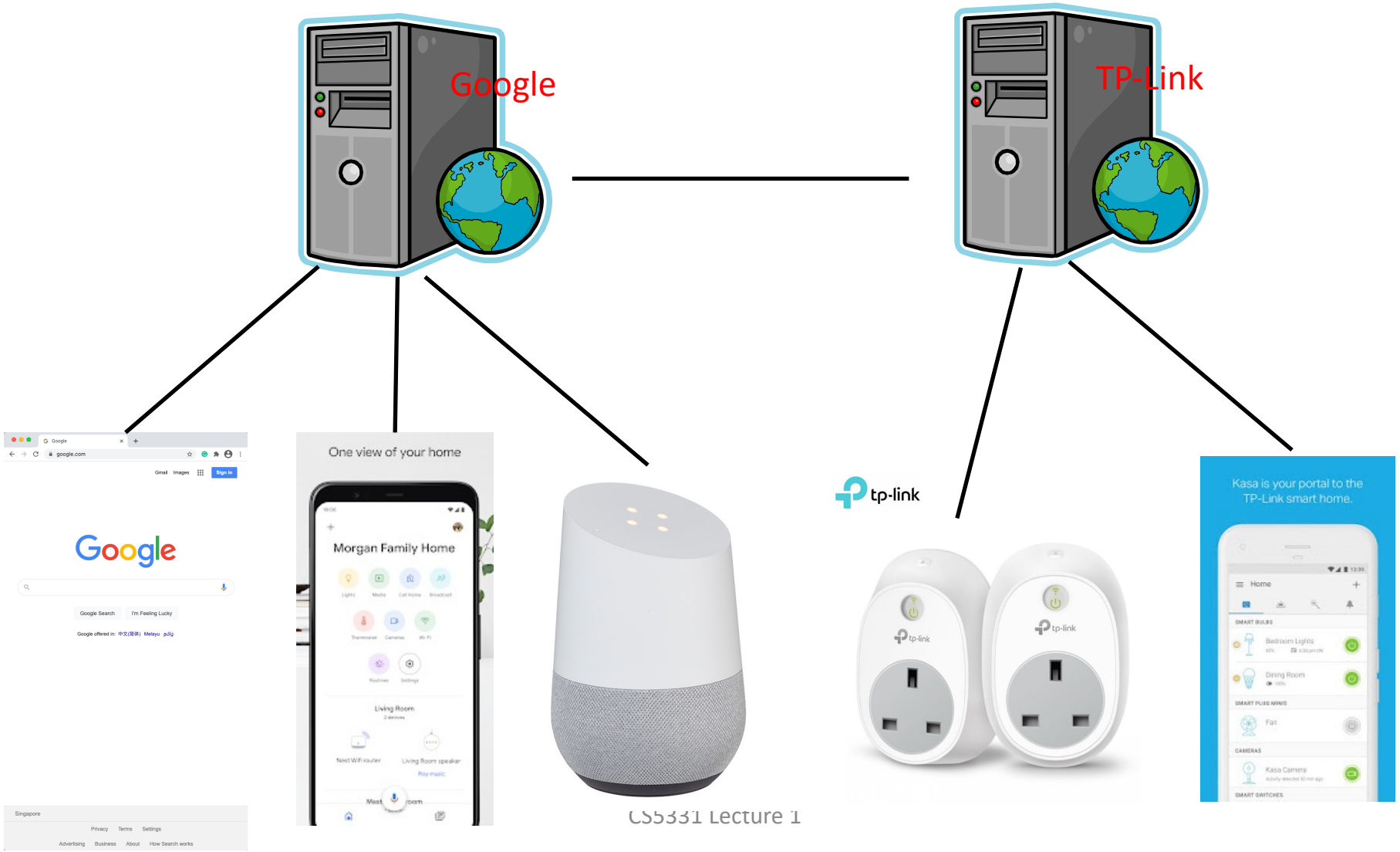
Extensions



Application Protocols

Web Frameworks

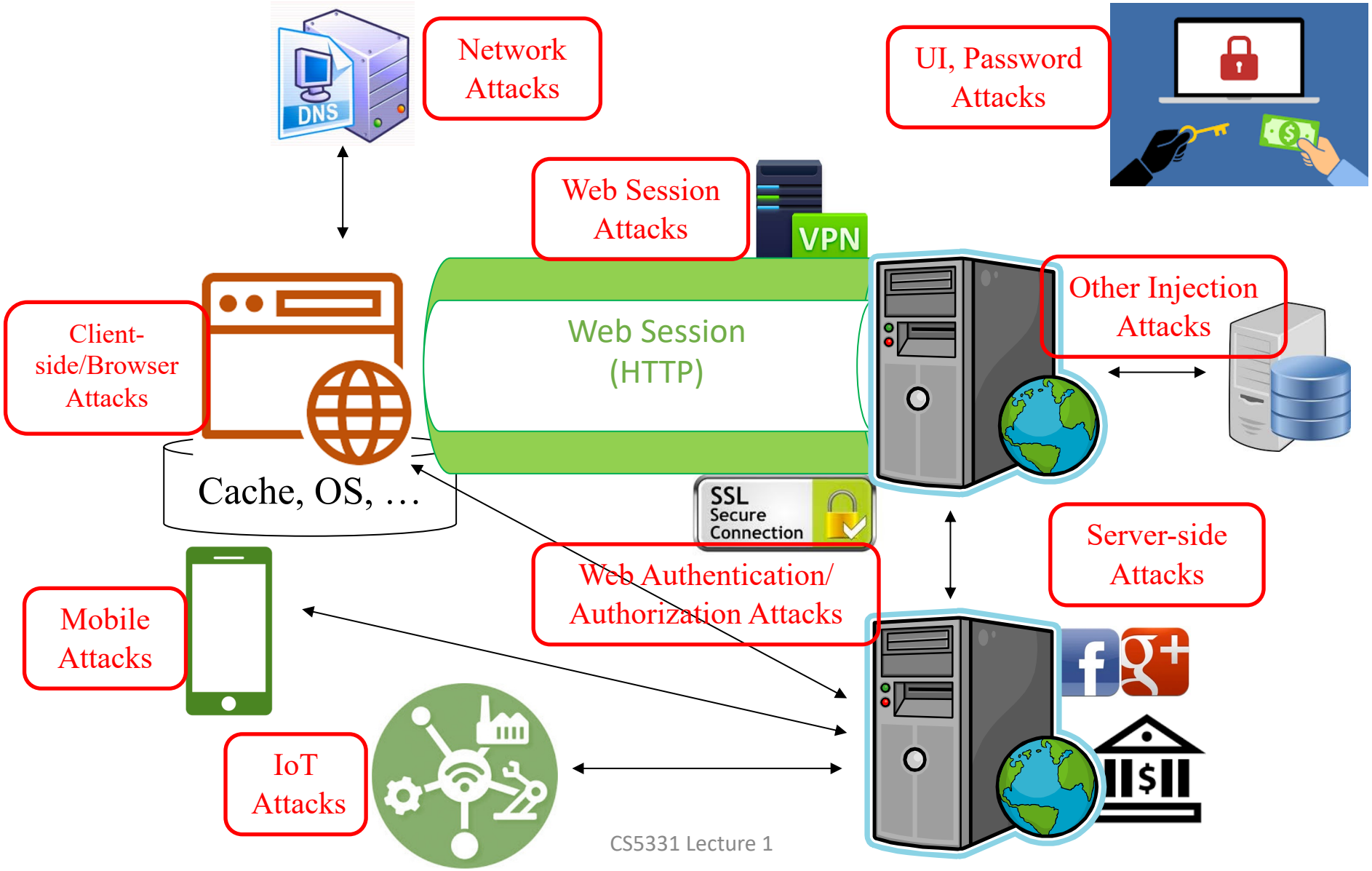
Why are they Web objects?



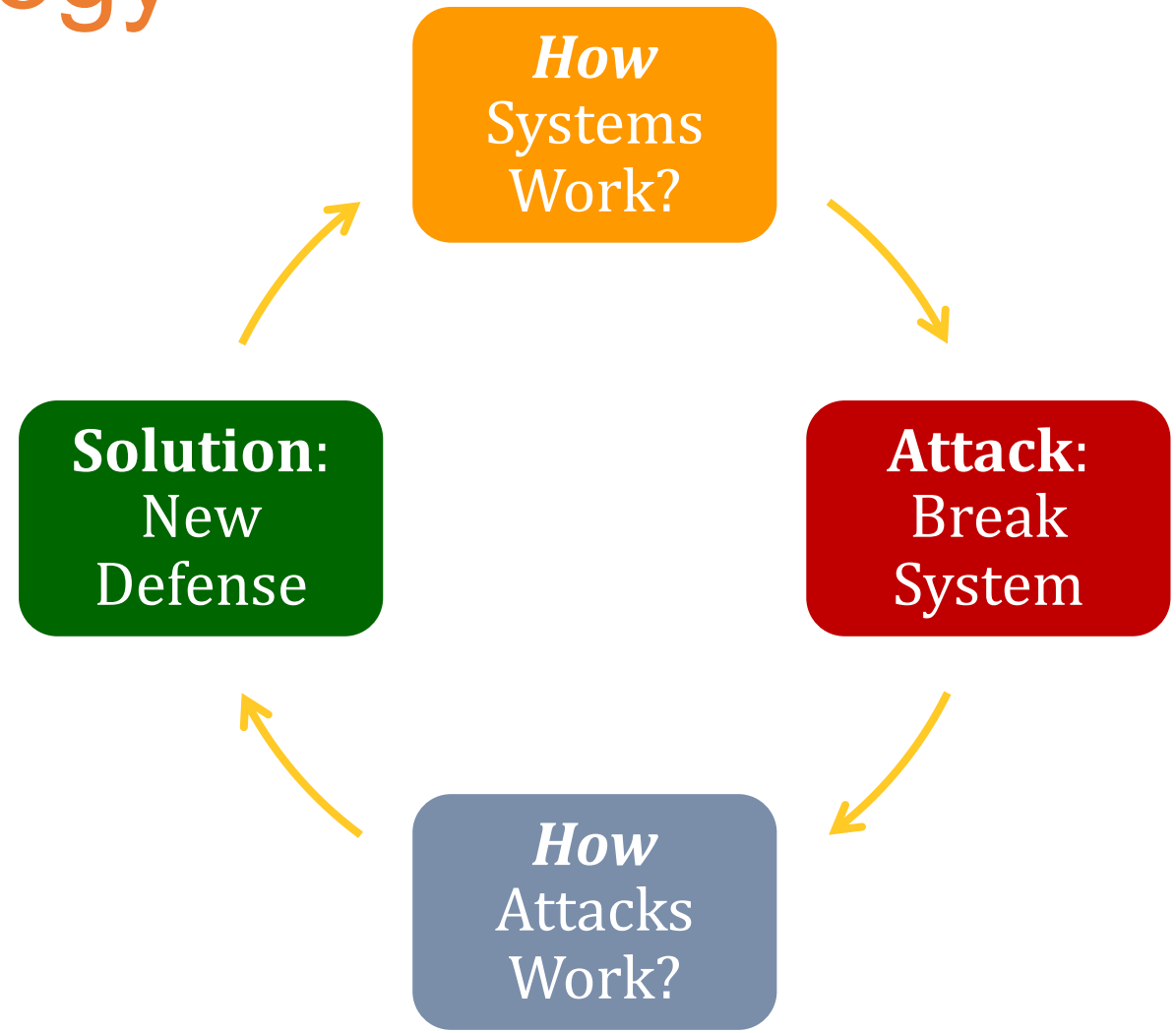
Web as a platform or infrastructure

- What is the trend of Web?
  - Chromebook
  - Web assembly (WASM)
  - Web 3.0?
  - Industry 4.0
  - Metaverse (Humanity 5.0)

Overview of Web Threats



Methodology



Ethics of Web Security

Learning to Attack

- If you know the enemy and know yourself, you need not fear the result of a hundred battles.

知己知彼，百战不殆。

Sun Tzu, Art of War

- To prevent attack, we need to learn how attack happens

Ethical Use of Security Information

- We discuss vulnerabilities and attacks
  - Most vulnerabilities have been fixed
  - Some attacks may still cause harm
  - Do *not* try these at home
- Purpose of this class
  - Learn to prevent malicious attacks
  - Use knowledge for good purposes

Administrative Matters

CA Components and Support

- Tests and quiz: 30%
- Individual assignments: 45%
- Final group project:
  - 25%
- Module resources on Canvas
- Class mailing list
  - cs5331ta@googlegroups.com
  - Consultation channel on Teams

Group-based Final Project

- Project Goal:
  - Apply our methodology: **Deeply** understand of a large system, understand attacks, and design solutions.
- Each group is expected to have two to three students
  - Please announce your group information to the TA mailing list

Project Proposal

- Due date: Mid February, 2023
- What to submit:
  - Problem description
  - Your solution and its novelty, list of reference
  - The platform and tools used in project
  - Project schedule
- You need to make sure your group is capable to handle the technical challenge independently

Progress Report

- Due date: Mid-March, 2023
- How is your progress compared to your proposal?
- Literature survey
- Initial approach description
- If you have difficult or question, raise them early

Final Report and Presentation

- Final report due before reading week
  - Following the typical format of technical report or research papers used in our class
- Final presentation: options for in-class presentation, or video-recording submission
  - 10 minutes for each group

Sample Project Topics

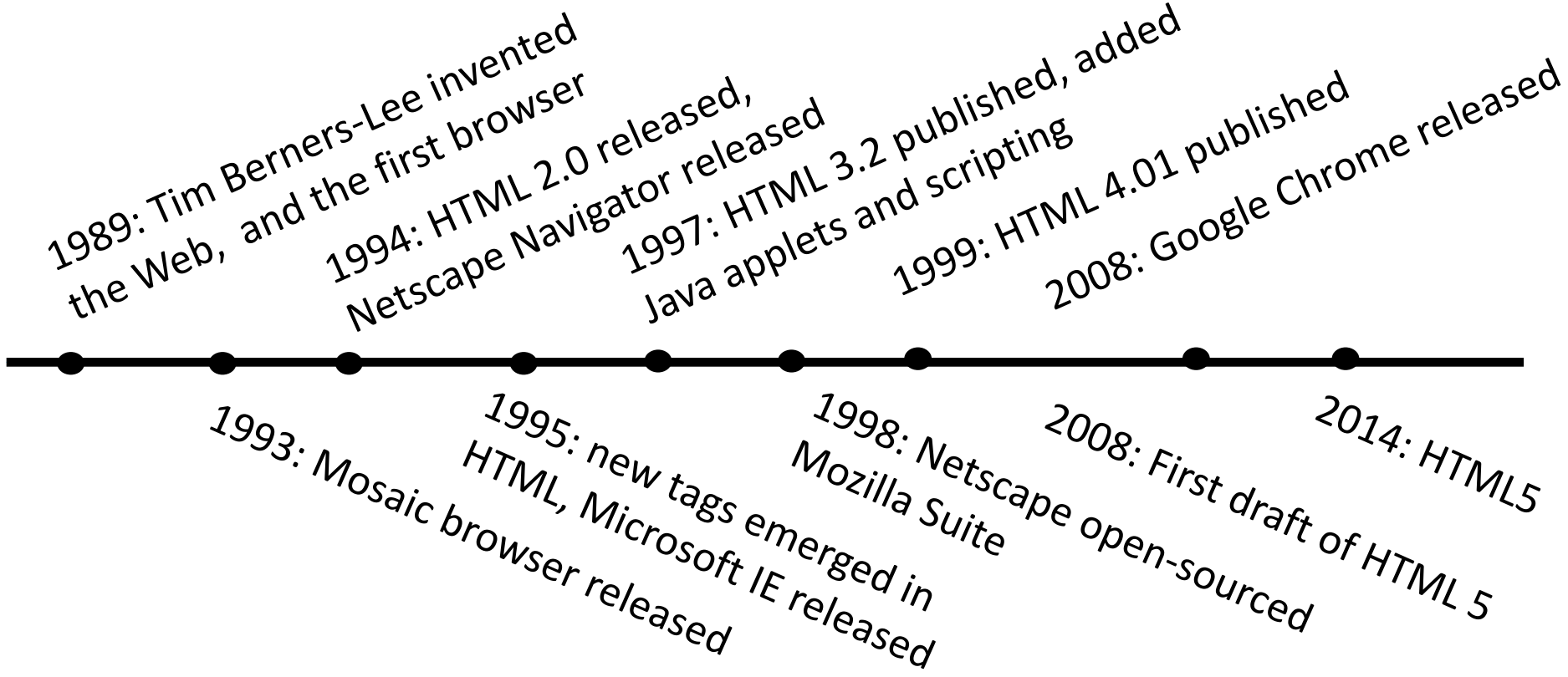
- Sanitization analysis to prevent XSS/SQLI attacks in web applications
- Transparent web-server plugin to give strong protection on sensitive web data
- Analysis of privacy leakage in browser extensions
- Detecting authentication vulnerability in IoT devices

Brief History of the Web

Assumption: You already know …

- Basic familiarity with:
  - HTTP, HTML, CSS, JS, URLs, Frames, DOM, Navigation, Cross-frame communication
- If not, CS 142 (Stanford) undergrad course
  - Browse through Week 1, 4, 5, 6, 7 material
- Or learn from https://www.w3schools.com
- Today: Just a recap…

Evolution of the Web



Architecture of Web

- HTTP protocol

```
http://www.example.com
GET / HTTP 1.1
Host: www.example.com
User-Agent: Mozilla/…
```

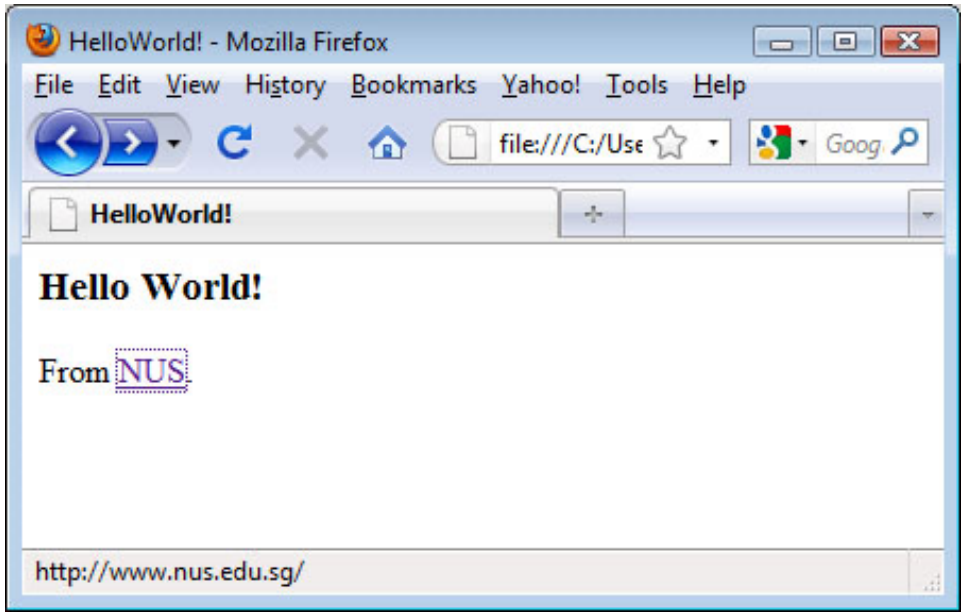
```
HTTP/1.1 200 OK
Date: Thu, 13 Oct2011
Server: Apache/1.3.41
Content-Type: text/html
... ...
```

A Static HTML Page

File Access

Request hello.html

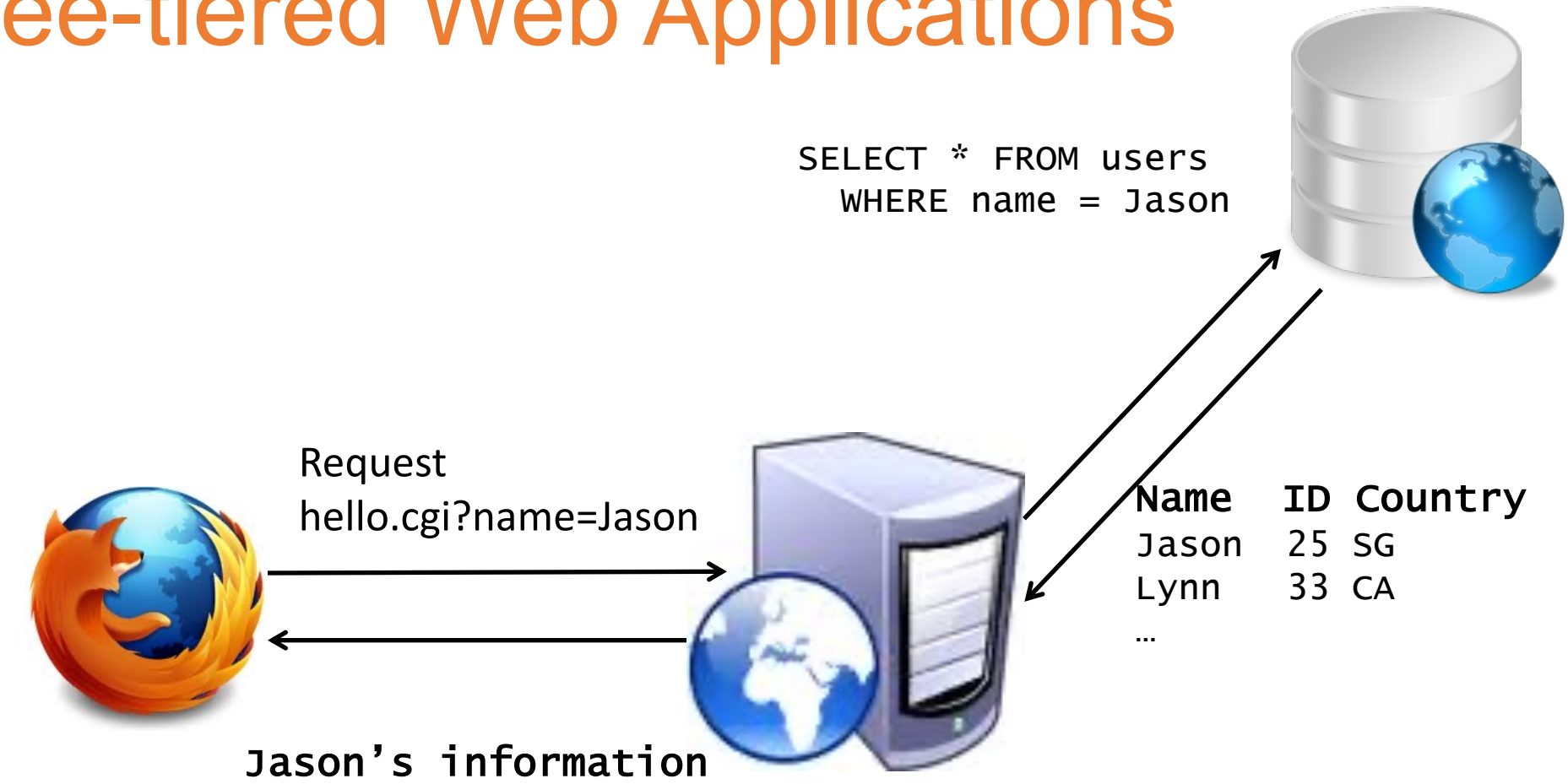
```
<HTML>
  <HEAD>
        <TITLE>HelloWorld!</TITLE>
  </HEAD>
  <BODY>
    <H3>Hello World!</H3>
    From <A HREF="http://www.nus.edu.sg">NUS</A>.
  </BODY>

</HTML>
```



Dynamic Web Pages

- Web servers dynamically generate different Web pages according to different requests and parameters
- CGI (Common Gateway Interface)
  - For Web servers to delegate the generation of Web pages to other programs, e.g. C, Perl, PHP, etc.
- Other alternatives:
  - FastCGI: more efficient process and IPC handlings
  - Apache modules: `mod_php`, `mod_perl`, …

Three-tiered Web Applications



PHP

- PHP:
  - a free server scripting language
  - a powerful tool for making dynamic and interactive Web pages
  - widely used
- Sample PHP page:

```
<!DOCTYPE html>
<html>
<body>

<?php
echo "My first PHP script!";
?>

</body>
</html>
```

Active HTML

- To reduce server load and increase responsiveness, new techniques execute code in browsers
  - JavaScript, Java Applet
- For example, using JavaScript to validate telephone number formats
  - Ensures telephone numbers in the format of XXX-XXX-XXXX.

Phone Number Format Validation

```
<HTML>
  <BODY>
  <script type="text/javascript">
  function FormValidate()
  {
     if(document.Form1.PhoneNumber.value.search(/\d{3}\-\d{3}\-\d{4}/)==-1){
        alert("Error: phone # format should be xxx-xxx-xxxx.");
        return false;
     }
  }
  </script>
  <form name="Form1" onsubmit="return FormValidate()">
  <input type="text" size="25" name="PhoneNumber" />
  <br /><br />
  <input type="submit" value="Submit" />
  </form>
  </BODY>

</HTML>
```

JavaScript

```
<!DOCTYPE html>
<html>

<body>

    <h1>My First JavaScript</h1>
    <p>Click the button to display the date.</p>
    <p id="demo"></p>

    <button type="button" onclick="myFunction()">Try it</button>

    <script>
        function myFunction() {
            document.getElementById("demo").innerHTML = Date();
        }
    </script>

    <script src="/scripts/foo.js"></script>

</body>

</html>
```

JavaScript

- JavaScript:
  - To program the behavior of web pages.
  - Make web pages interactive and responsive.
- Some usages:
  - Change HTML content:
    ```
    document.getElementById("demo").innerHTML = Date();
    ```
  - Change HTML attributes, e.g. change the `src` (source) attribute of an `<img>` tag.
  - Change HTML Styles (CSS):
    ```
    document.getElementById("demo").style.fontSize = "35px";
    ```
  - Hide HTML elements:
    ```
    document.getElementById("demo").style.display = "none";
    ```
  - Show HTML elements:
    ```
    document.getElementById("demo").style.display = "block";
    ```
  - Read cookies:
    ```
    var x = document.cookie;
    ```
  - Navigate to a new place:
    ```
    window.location = "http://www.mozilla.org";
    ```
  - Pop up an alert box: `window.alert("Hello world");`
  - … (many more)

JavaScript

- Ways of introducing JavaScript code:
  - Between <script> and </script> tags:
    - Inline between `<script>` and `</script>`:
      `<script>alert(1)</script>`
    - External file using `src` attribute:
      `<script src="myScript.js"></script>`
    - External reference/server using `src` attribute :
      `<script src="https://www.w3schools.com/js/myScript1.js">`
      `</script>`
      The script has privileges of the loading page, not the source server
  - HTML event handler:
    `<button onclick="alert(1)">Click me</button>`
  - URL (with `javascript:` pseudo protocol):
    `<iframe src="javascript:alert(1)">`
    (Try entering this in your browser's address bar: `javascript:alert(1);`)
  - Dynamically-evaluated CSS style (on older browsers):
    `<x style:x:expression(alert(1))>`

Reading JavaScript

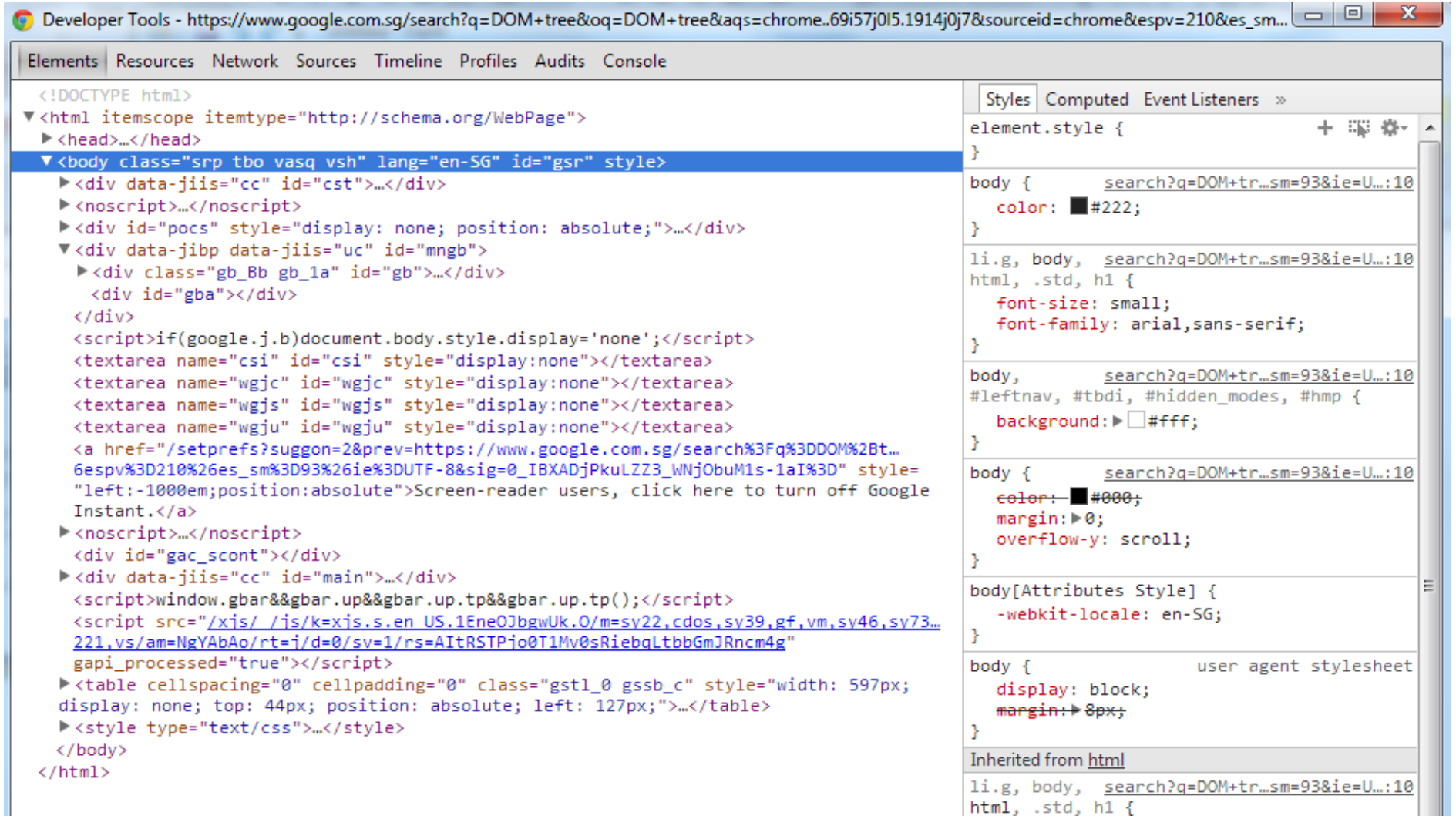
https://www.google.com.sg/xjs/_/js/k=xjs.s.en_US.1EneOJbgwUk.O/m=wta,aspn,sy90,sy92,sy82,ac

```
var _=_||{};(function(_){var window=this;
try{
(0,_.yi)("wta");var xE,yE,zE,AE,sIa,tIa,uIa,BE,vIa=function(a,b){a+="&ei="+window.google.kEI;b&&(a+
e.innerHTML:null)BE=d,(0,_.Mf)(xE,"width",d+"px"),xE&&(d=xE.querySelector("div.wtalbc"))&&(d.innerH
(window.document.body,"click",zIa),vIa("o",b)}},zIa=function(a){a=a.target||a.srcElement;
null===a||a==zE||(0,_.Ig)(a,"wtaal")||(0,_.Ig)(a,"wtali")||CE("cm")},wIa=function(){if(xE){(0,_.Mf)
a=xE.querySelector("a.wtaal");a&&AE&&((0,_.Vf)(a,"click",AE),AE=null)}zE=null}},CE=function(a,b){BI
(0,_.Wh)(a)/2:yE&&(c-=(0,_.Wh)(a)-3);var d=yE?-1:1,b=(0,_.Th)()?b+d*
c:b-d*c;a=(0,_.lf)(a)+(0,_.Vh)(a)+11;var c=0,d=(0,_.Th)()?window.document.querySelector(".wtalbar")
window.document.querySelector(".wtalbal"):window.document.querySelector(".wtalbar"),f=yE;(0,_.Th)()
(c=Math.max(0,6-b),(0,_.Ye)(d,"right",c+13+"px"),(0,_.Ye)(e,"right",c+"px"));xE&&((0,_.Ye)(xE,"left
a+"px"))}},BIa=function(){return xE&&"visible"==(0,_.Uh)(xE,"visibility",!0)?!0:!1},CIa=function(a)
class="wtalbc f"></div>');(0,_.Mf)(a,"id","wtalb");(0,_.Mf)(a,"display","none");xE=a;(0,_.Kf)(a)},F
{b=b||window.event;b.preventDefault&&b.preventDefault();b.returnValue=
```

http://jsbeautifier.org/

```
1  var _ = _ || {};
2  (function (_) {
3          var window = this;
4          try {
5              (0, _.yi)("wta");
6              var xE, yE, zE, AE, sIa, tIa, uIa, BE, vIa = function (a, b) {
7                      a += "&ei=" + window.google.kEI;
8                      b && (a += "&ved=" + b);
9                      window.google.log("wta", a)
10                 }, AIa = function (a, b, c, d) {
11                     wIa();
12                     if (a && xE) {
13                         var e;
14                         if (e = (e = a.parentNode.querySelector(".wtalbc")) ? e.innerHTML : null) BE = d,
15                     }
16                 }, zIa = function (a) {
17                     a = a.target || a.srcElement;
18                     null === a || a == zE || (0, _.Ig)(a, "wtaal") || (0, _.Ig)(a, "wtali") || CE("cm")
```

DOM


Developer Tools - https://www.google.com.sg/search?q=DOM+tree&oq=DOM+tree&aqs=chrome..69i57j0l5.1914j0j7&sourceid=chrome&espv=210&es_sm...
Elements Resources Network Sources Timeline Profiles Audits Console
<!DOCTYPE html>
<html itemscope itemtype="http://schema.org/WebPage">
<head>…</head>
<body class="srp tbo vasq vsh" lang="en-SG" id="gsr" style>
…
<noscript>…</noscript>
…
<div data-jibp data-jiis="uc" id="mngb">
…

<script>if(google.j.b)document.body.style.display='none';</script>
<textarea name="csi" id="csi" style="display:none"></textarea>
<textarea name="wgjc" id="wgjc" style="display:none"></textarea>
<textarea name="wgjs" id="wgjs" style="display:none"></textarea>
<textarea name="wgju" id="wgju" style="display:none"></textarea>
<a href="/setprefs?suggon=2&prev=https://www.google.com.sg/search%3Fq%3DDOM%2Bt…
6espv%3D210%26es_sm%3D93%26ie%3DUTF-8&sig=0_IBXADjPkuLZZ3_WNjObuM1s-1aI%3D" style=
"left:-1000em;position:absolute">Screen-reader users, click here to turn off Google
Instant.
<noscript>…</noscript>

…
<script>window.gbar&&gbar.up&&gbar.up.tp&&gbar.up.tp();</script>
<script src="/xjs/_/js/k=xjs.s.en_US.1EneOJbgwUk.O/m=sy22,cdos,sy39,gf,vm,sy46,sy73…
221,vs/am=NgYAbAo/rt=j/d=0/sv=1/rs=AItRSTPjo0T1Mv0sRiebqLtbbGmJRncm4g"
gapi_processed="true"></script>
<table cellspacing="0" cellpadding="0" class="gstl_0 gssb_c" style="width: 597px;
display: none; top: 44px; position: absolute; left: 127px;">…</table>
<style type="text/css">…</style>
</body>
</html>
Styles Computed Event Listeners »
element.style {
}
body { search?q=DOM+tr…sm=93&ie=U…:10
color: #222;
}
li.g, body, search?q=DOM+tr…sm=93&ie=U…:10
html, .std, h1 {
font-size: small;
font-family: arial,sans-serif;
}
body, search?q=DOM+tr…sm=93&ie=U…:10
#leftnav, #tbdi, #hidden_modes, #hmp {
background: #fff;
}
body { search?q=DOM+tr…sm=93&ie=U…:10
color: #000;
margin: 0;
overflow-y: scroll;
}
body[Attributes Style] {
-webkit-locale: en-SG;
}
body { user agent stylesheet
display: block;
margin: 8px;
}
Inherited from html
li.g, body, search?q=DOM+tr…sm=93&ie=U…:10
html, .std, h1 {

DOM Manipulation

```
<!DOCTYPE html>
<html>

<body>

    <h1>My First JavaScript</h1>
    <p>Click the button to display the date.</p>
    <p id="demo"></p>

    <button type="button" onclick="myFunction()">Try it</button>

    <script>
        function myFunction() {
            document.getElementById("demo").innerHTML = Date();
        }
    </script>

    <script src="/scripts/foo.js"></script>

</body>

</html>
```

Inspecting DOM using Browser's Developer Tool

- DOM tree view displays DOM structure of the current page
- Each DOM node is a page element, e.g. a header node, paragraph node
- You can live-edit the content and structure of your pages, but *not* the source files https://developers.google.com/web/tools/chrome-devtools/inspect-styles/edit-dom

JavaScript Object Notation (JSON)

- JSON:
  - A lightweight data-interchange format (compared to XML) .
  - Commonly used for information exchange between the browser and server.
  - Language independent.
  - Easy for humans to read and write, easy for machines to parse and generate.
- Two structures:
  - An unordered collection of name/value pairs (i.e. **object**, dictionary, or associative array): {*name1=value1*, *name2=value2*, … }.
  - An ordered list of values (i.e. **array**, list, vector, or sequence): [*value1*, *value2*, …].

JavaScript Object Notation (JSON)

- JSON and JavaScript:
  - JSON is a string representation of a JavaScript object (hence the name).
  - A sample JavaScript object:

```
var myObj = { "name":"AhBeng", "age":30, "city":"Singapore" };
```

  - Corresponding JSON string representation:

```
{ "name":"AhBeng", "age":30, "city":"Singapore" }
```

- Conversions:
  - A JavaScript object to JSON text:
    `var myJSON = JSON.stringify(myObj);`
  - A JSON text to a JavaScript object:
    `var myObj = JSON.parse(myJSON);`
- PHP functions to handle JSON:
  - `json_encode()`: a PHP object into JSON
  - `json_decode()`: JSON into a PHP object
- Reference: https://www.w3schools.com/js/js_json_intro.asp

Frames / Windows

- Each window is a frame
  - A frame hosts a web origin
- iFrames: inline frame
  - Can host a different site, allowing a mashup
    - May be hidden (0px width-ht), no borders, transparent
- Why do we use frames?
  - To delegate screen area to content from another origin
  - Parent iframe can still work even if a child frame is broken
  - Browser provides isolation based on frames
  - A frame can access data belonging to its own origin (“principal”) only: will be discussed more later

Frames / Windows

```
<!DOCTYPE html>
<html>
<body>

<iframe src="http://www.w3schools.com">
  <p>Your browser does not support iframes.</p>
</iframe>

<script>
document.write (frames[0].parent.location.href);
</script>

</body>
</html>
```

http://www.w3schools.com/tags/tr

Frame Navigation

- Can be “navigated” by
  - User typing in the URL bar, user clicks links
  - Using scripts

```
<iframe src="http://www.w3schools.com">
  <p>Your browser does not support iframes.</p>
</iframe>

<script>
frames[0].location = "http://www.comp.nus.edu.sg/~prateeks/teaching/sp14/cs5331-sp14.html";
</script>
```

- Note: many modern browsers now implement “descendant iFrame” navigation policy

Summary

- Learning principles through practice
  - Seeing is believing
- Practical skills
  - Experience with web technology
  - Solutions for your own concerns
- Learn and solve cutting-edge research problems in web security
- Loaded with programming and system-level tasks (to get your hand dirty)