Web Application Security

Lecture:02

Web Technologies

Miss Maryam Malik
Lecturer
Department of cyber security
Air university Islamabad

Hypertext Transfer Protocol (HTTP)

- Connectionless protocol
- Client sends an HTTP request to aWeb server
- Gets an HTTP response
- No session formed, nothing remembered--no "state"

HTTP Requests

- Verb: GET (also called "method") URL: /css?family=Roboto:400,700
- Portion after ? is the query string containing
- Parameters
- Version: HTTP/1.1

```
GET /css?family=Roboto:400,700 HTTP/1.1
Host: fonts.googleapis.com
Connection: keep-alive
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_4) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/52.0.2743.116 Safari/537.36
X-Chrome-UMA-Enabled: 1
X-Client-Data: CKi1yQEIhrbJAQimtskBCMG2yQEIhZnKAQjxnMoB
Accept: text/css,*/*;q=0.1
Referer: http://aol-travel-priceline-widget.s3.amazonaws.com/home.html
Accept-Encoding: gzip, deflate, sdch
Accept-Language: en-US,en;q=0.8
```

HTTP Requests

- Referer: URL the request originated from
- User-Agent: browser being used
- Host: Hostname of the server
- Essential when multiple hosts run on the same IP Requi

```
GET /css?family=Roboto:400,700 HTTP/1.1
Host: fonts.googleapis.com
Connection: keep-alive
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_4) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/52.0.2743.116 Safari/537.36
X-Chrome-UMA-Enabled: 1
X-Client-Data: CKi1yQEIhrbJAQimtskBCMG2yQEIhZnKAQjxnMoB
Accept: text/css,*/*;q=0.1
Referer: http://aol-travel-priceline-widget.s3.amazonaws.com/home.html
Accept-Encoding: gzip, deflate, sdch
Accept-Language: en-US,en;q=0.8
```

HTTP Requests

- Cookie: additional parameters the server has issued to the client

```
GET / HTTP/1.1
Host: www.aol.com
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_4) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/52.0.2743.116 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Encoding: gzip, deflate, sdch
Accept-Language: en-US,en;q=0.8
Cookie: seg_version=2; grvinsights=35c06b5ba037ab3bad4ae2c04b408301;
_ga=GA1.2.1579599759.1469159982; AMPMV-45=ada77aa4d0a0e4f4dc3d26e8078fd88d7fb87e3b;
AMCV_6B25357E519160E40A490D44%40AdobeOrg=1256414278%7CMCMID
%7C73953456523279724286843161717900149091%7CMCAID%7CNONE
%7CMCAAMLH-1470844977%7C7%7CMCAAMB-1470844977%7Chmk_Lq6TPIBMW925SPhw3Q;
AMPMV-40=deae0e17945f78867f4024b45753e2ba2503e72d; s_pers=%20s_getnr%3D1470843782236-
Repeat%7C1533915782236%3B%20s_nrgvo%3DRepeat%7C1533915782238%3B;
UNAUTHID=1.2bee72f03b3711e696cae7771dcb269e.4175;
T_UNAUTHID=1.62bb7d1f797f4899a7132cc65f86f11c.4795
```

HTTP Response

- First line
  - HTTP version
  - Status code (200 in this case)
- Textual "reason phrase" describing the response
  - Ignored by br

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Content-Type: application/javascript;charset=UTF-8
Date: Sun, 21 Aug 2016 18:04:42 GMT
Server: Apache-Coyote/1.1
Transfer-Encoding: chunked

16d0
jQuery112002265769519066880 3_1471802683680({"payload":{"articles":[{"title":"Trump
campaign hints at softening immigration stance","clickUrl":"http://grvrdr.aol.com/302/
redirect?
grcc3=jVJNjxshDP0tc8iRifky0Fu0TaVeW_VKZT4mO20yE2XI5u_Xk2yk9rSFh8FgPczDL8cx_96_1alFFaVx0o
```

HTTP Response

- Server: banner of server software
- Not always accurate
- Set-Cookie used to set cookie values

```
HTTP/1.1 200 OK
Date: Sun, 21 Aug 2016 18:04:41 GMT
Server: Apache/2.4.6 (CentOS)
X-Frame-Options: SAMEORIGIN
X-Content-Type-Options: nosniff
request-id: bb7e0c5a-67c9-11e6-965f-00163e638b0f
Cache-Control: no-cache, no-store, private, max-age=0
Set-Cookie: AMPMV-40=0958af3c2c1695981fb105fa0556e9cf268f1cbb; expires=Tue, 01-Nov-2016 04:00:00 GMT; Max-Age=6170119; path=/
Set-Cookie: AMPMV-37=8f222490f08726dd8ea65f0b8f1d663994a02b74; expires=Tue, 20-Sep-2016 18:04:41 GMT; Max-Age=2592000; path=/article/
Set-Cookie: AMPMV-38=bdeafba36ea25ee9930e88d3a46b231473c1681b; expires=Thu, 01-Sep-2016 04:00:00 GMT; Max-Age=899719; path=/
Set-Cookie: AMPMV-48=022d64f20a7e9313b46c93bd67e903d2aae4e40e; expires=Tue, 20-Sep-2016 18:04:41 GMT; Max-Age=2592000; path=/
Content-Encoding: gzip
X-AOL-HN: amp-blogside-ent-s07
X-XSS-Protection: 1; mode=block
Vary: Accept-Encoding
Keep-Alive: timeout=5, max=499953
Connection: Keep-Alive
Transfer-Encoding: chunked
Content-Type: text/html;charset=UTF-8
```

HTTP Response

- Pragma: tells browser not to store response in its cache
- Expires: set to a date in the past to ensure that the content is freshly loaded

```
HTTP/1.1 200 OK
Cache-Control: no-cache,no-store,must-revalidate,max-age=0,proxy-revalidate,no-transform,private
Content-Encoding: gzip
Content-Type: application/javascript; charset=UTF-8
Date: Sun, 21 Aug 2016 18:04:42 GMT
DCS: usw2-prod-dcs-2.edge-usw2.demdex.com master-3.11.0.20160802.151351 5ms
Expires: Thu, 01 Jan 2009 00:00:00 GMT
P3P: policyref="/w3c/p3p.xml", CP="NOI NID CURa ADMa DEVa PSAa PSDa OUR SAMa BUS PUR COM NAV INT"
Pragma: no-cache
Vary: Accept-Encoding, User-Agent
transfer-encoding: chunked
Connection: keep-alive

bc
```

HTTP Response

- Message Body after header contains data of type specified in Content-Type header

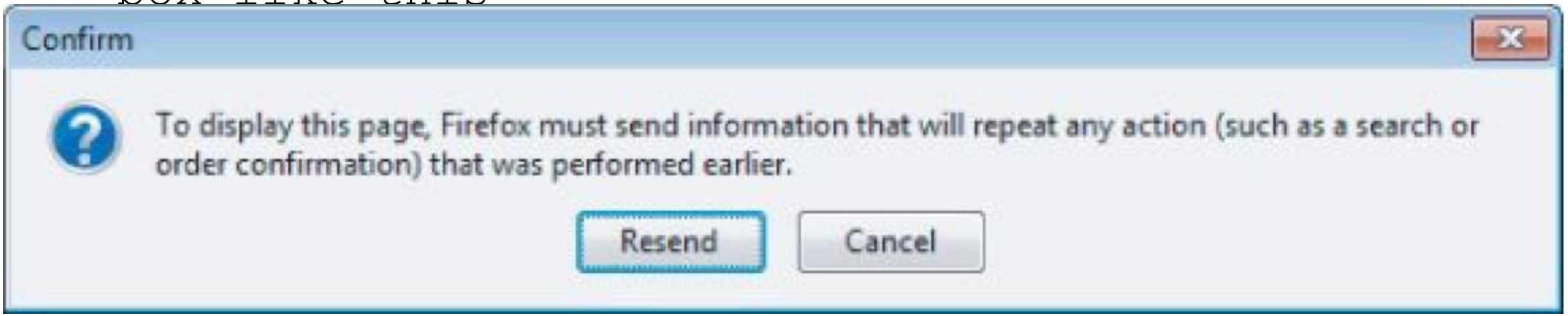
```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Content-Type: application/javascript;charset=UTF-8
Date: Sun, 21 Aug 2016 18:04:42 GMT
Server: Apache-Coyote/1.1
Transfer-Encoding: chunked

16d0
jQuery112002265769519066880 3_1471802683680({"payload":{"articles":[{"title":"Trump
campaign hints at softening immigration stance","clickUrl":"http://grvrdr.aol.com/302/
redirect?
grcc3=jVJNjxshDP0tc8iRifky0Fu0TaVeW_VKZT4mO20yE2XI5u_Xk2yk9rSFh8FgPczDL8cx_96_1alFFaVx0o
```

HTTP Methods: GET

- GET retrieves resources
  - Can send parameters in the URL query string
  - Users can bookmark the whole URL
  - Whole URL may appear in server logs and in Referer headers
  - Also on the browser's screen
- Don't put sensitive information in the query string

HTTP Methods: POST

- POST requests perform actions, like buying something
- Clicking the browser's Backbutton displays a box like this



Other HTTP Methods

- HEAD returns only the header, not the body

Can be used to check if a resource is available before getting it

- OPTIONS shows allowed methods
- PUT uploads to server (usually disabled)

URL (Uniform Resource Locator)

- If protocol is absent, it defaults to HTTP
- If port is absent, it uses the default port for the protocol
- 80 for HTTP, 443 for HTTPS, etc.

```
protocol://hostname[:port]/[path/]file[?param=value]
```

HTTP Headers

General Headers

- `Connection` tells the other end of the communication whether it should close the TCP connection after the HTTP transmission has completed or keep it open for further messages.
- `Content-Encoding` specifies what kind of encoding is being used for the content contained in the message body, such as `gzip`, which is used by some applications to compress responses for faster transmission.
- `Content-Length` specifies the length of the message body, in bytes (except in the case of responses to `HEAD` requests, when it indicates the length of the body in the response to the corresponding `GET` request).
- `Content-Type` specifies the type of content contained in the message body, such as `text/html` for HTML documents.
- `Transfer-Encoding` specifies any encoding that was performed on the message body to facilitate its transfer over HTTP. It is normally used to specify chunked encoding when this is employed.

Request Headers

- `Accept` tells the server what kinds of content the client is willing to accept, such as image types, office document formats, and so on.
- `Accept-Encoding` tells the server what kinds of content encoding the client is willing to accept.
- `Authorization` submits credentials to the server for one of the built-in HTTP authentication types.
- `Cookie` submits cookies to the server that the server previously issued.
- `Host` specifies the hostname that appeared in the full URL being requested.
- `If-Modified-Since` specifies when the browser last received the requested resource. If the resource has not changed since that time, the server may instruct the client to use its cached copy, using a response with status code 304.
- `If-None-Match` specifies an *entity tag*, which is an identifier denoting the contents of the message body. The browser submits the entity tag that the server issued with the requested resource when it was last received. The server can use the entity tag to determine whether the browser may use its cached copy of the resource.
- `Origin` is used in cross-domain Ajax requests to indicate the domain from which the request originated (see Chapter 13).
- `Referer` specifies the URL from which the current request originated.
- `User-Agent` provides information about the browser or other client software that generated the request.

Response Headers

- `Access-Control-Allow-Origin` indicates whether the resource can be retrieved via cross-domain Ajax requests (see Chapter 13).
- `Cache-Control` passes caching directives to the browser (for example, `no-cache`).
- `ETag` specifies an entity tag. Clients can submit this identifier in future requests for the same resource in the `If-None-Match` header to notify the server which version of the resource the browser currently holds in its cache.
- `Expires` tells the browser for how long the contents of the message body are valid. The browser may use the cached copy of this resource until this time.
- `Location` is used in redirection responses (those that have a status code starting with 3) to specify the target of the redirect.
- `Pragma` passes caching directives to the browser (for example, `no-cache`).
- `Server` provides information about the web server software being used.
- `Set-Cookie` issues cookies to the browser that it will submit back to the server in subsequent requests.
- `WWW-Authenticate` is used in responses that have a 401 status code to provide details on the type(s) of authentication that the server supports.
- `X-Frame-Options` indicates whether and how the current response may be loaded within a browser frame (see Chapter 13).

Cookies

- Cookies are resubmitted in each request to the same domain
- Unlike other request parameters, such as the que

A server issues a cookie using the `Set-Cookie` response header, as you have seen:

```
Set-Cookie: tracking=tI8rk7joMx44S2Uu85nSWc
```

The user's browser then automatically adds the following header to subsequent requests back to the same server:

```
Cookie: tracking=tI8rk7joMx44S2Uu85nSWc
```

Set-Cookie Header

- Optional attributes
- expires - date when the cookie stops being valid
- If absent, cookie is used only in the current browser session
- domain - specified domain for which cookie is valid
- Must be the same or a parent of the domain from which the cookie is received
- "Same-Origin Policy"

Status Codes Groups

- **1xx** – Informational.
- **2xx** – The request was successful.
- **3xx** – The client is redirected to a different resource.
- **4xx** – The request contains an error of some kind.
- **5xx** – The server encountered an error fulfilling the request.

Important Status Codes

- 200 OK - request succeeded, response body contains result
- 301Moved Permanently - redirects the browser, client should use new URL in the future
- 302Found - redirects browser temporarily. Client should revert to original URL in subsequent requests

Important Status Codes

- 304 Not Modified - browser should use cached copy of resource
- 400 Bad Request - invalid HTTP request
- 401 Unauthorized - Server requires HTTP authentication.
- WWW-Authenticate header specifies the type(s) of authentication supported

Important Status Codes

- 403 Forbidden - no one is allowed to access resource, regardless of authentication
- 404 Not Found - requested resource does not exist
- 500 Internal Server Error - unhanded exception in an app, such as a PHP error

HTTPS

- HTTP over SSL (Secure Sockets Layer)
- Actually now TLS (Transport Layer Security)
  All versions of SSL are deprecated
- Protects data with encryption
- Protects data in motion, but not at rest or in use

HTTPS and Man-in-the- Middle (MITM) Attacks

- HTTPS connections use public-key cryptography and end-to-end encryption
- Only the endpoints can decrypt traffic
- Companies wishing to restrict HTTPS traffic have two choices
- Perform complete MITM with fake certificates, or real root certificates from trusted CA's
- Allow encrypted traffic to trusted domains without being able to inspect it

HTTP Proxies

- Browser sends requests to proxy server
- Proxy fetches resource and sends it to browser
- Proxies may provide caching, authentication, and access control

HTTPS and Proxies

- Browser sends an HTTP request to the proxy using the CONNECT method and destination hostname and port number
- If proxy allows the request, it returns 200 status and keeps the TCP connection open
- Thereafter acts as a pure TCP-level relay to the destination web server

HTTP Authentication

- Basic: sends username and password in Base64-encoding
- NTLM: Uses Windows NTLM protocol (MD4 hashing)
- Digest: Challenge-response using MD5 hashing
- These are generally used in intranets, not on the Internet
- All are very weak cryptographically, and should be protected with HTTPS

Web Functionality

Server-Side Functionality

- Static content - HTML pages and images that are the same for all users
- Dynamic content - response created in the fly, can be customized for each user
- Created by scripts on the server
- Customized based on parameters in the request

HTTP Parameters

- May be sent in these ways:
  - **In the URL query string**
  - **In the file path of REST-style URLs**
  - **In HTTP cookies**
  - **In the body of requests using the `POST` method**

Technologies Used in Server-Side Development

- Scripting languages such as PHP, VBScript, and Perl
- Web application platforms such as ASP.NET and Java
- Web servers such as Apache, IIS, and Netscape Enterprise
- Databases such as MS-SQL, Oracle, and MySQL
- Other back-end components such as filesystems, SOAP-based web services, and directory services

The Java Platform

- Standard for large-scale enterprise applications
- Lends itself to multitiered and load-balanced architectures
- Well-suited to modular development and code reuse
- Runs on Windows, Linux, and Solaris

Java Platform Terms

- **Enterprise Java Bean (EJB)**
- Heavyweight software component to encapsulate business logic, such as transactional integrity
- **Plain Old Java Object (POJO)**
- User-defined, lightweight object, distinct from a special object such as an EJB
- **Java Servlet**
- Object on an application server that receives HTTP requests from client and returns HTTP responses

Common Components

- Third-party or open-source components that are often used alongside custom-built code

  - **Authentication** — JAAS, ACEGI
  - **Presentation layer** — SiteMesh, Tapestry
  - **Database object relational mapping** — Hibernate
  - **Logging** — Log4J

ASP.NET

- Microsoft's web application framework
- Competitor to Java platform
- Uses .NET Framework, which provides a virtual machine (the Common Language Runtime) and a set of powerful APIs (Application Program Interfaces)
- Applications can be written in any .NET language, such as C# or VB.NET

PHP

- Originally "Personal Home Page", now "PHP Hypertext Processor"
- Often used on LAMP servers
- Linux, Apache, MySQL, and PHP
- Free and easy to use, but many security problems
- Both in PHP itself and in custom code using it

Ruby on Rails

- Allows rapid development of applications
- Can autogenerate much of the code if developer follows the Rails coding style and naming conventions
- Has vulnerabilities like PHP

SQL (Structured Query Language)

- Used to access data in relational databases, such as Oracle, MS-SQL, and MySQL
- Data stored in tables, each containing rows and columns
- SQL queries are used to read, add, update, or delete data
- SQL injection vulnerabilities are very severe

XML (eXtensible Markup Language)

- A specification to encode data in machine-readable form
- Markup uses tags

```
<pet>ginger</pet>
<pets><dog>spot</dog><cat>paws</cat></pets>
```

Tags may include attributes, which are name/value pairs:

```
<data version="2.1"><pets>...</pets></data>
```

Web Services and SOAP (Simple Object Access Protocol)

- SOAP uses HTTP and XML to exchange data

```
POST /doTransfer.asp HTTP/1.0
Host: mdsec-mgr.int.mdsec.net
Content-Type: application/soap+xml; charset=utf-8
Content-Length: 891
<?xml version="1.0"?>
<soap:Envelope xmlns:soap="http://www.w3.org/2001/12/soap-envelope">
  <soap:Body>
      <pre:Add xmlns:pre=http://target/lists soap:encodingStyle=
"http://www.w3.org/2001/12/soap-encoding">
      <Account>
        <FromAccount>18281008</FromAccount>
        <Amount>1430</Amount>
        <ClearedFunds>False</ClearedFunds>
        <ToAccount>08447656</ToAccount>
      </Account>
    </pre:Add>
  </soap:Body>
</soap:Envelope>
```

SOAP

- If user-supplied data is incorporated into SOAP requests, it can have code injection vulnerabilities
- Server usually publishes available services and parameters using Web Services Description Language (WSDL)
- soapUI and other tools can generate requests based on WSDL file

Client-Side Functionality (in browser)

HTML
Hypertext Markup Language

- HTML used for formatting "markup"
- XHTML is based on XML and is stricter than old versions of HTML

Hyperlinks

- Clickable text that go to URLs

  Clicking this link:

```
<a href="129S/129S_S22.shtml" target="_blank">
CNIT 129S: Securing Web Applications</a>
```

- Makes this request

```
GET /129S/129S_S22.shtml HTTP/2
Host: samsclass.info
```

HTML Forms

```
<form action='cookielogin.php' method='GET' name='idform'>

Name: <INPUT type='text' name='n' size=60>
<p>
Password: <INPUT type='text' name='p' size=60>
<p>
<input type='submit'>
</form>
```

HTTP Request

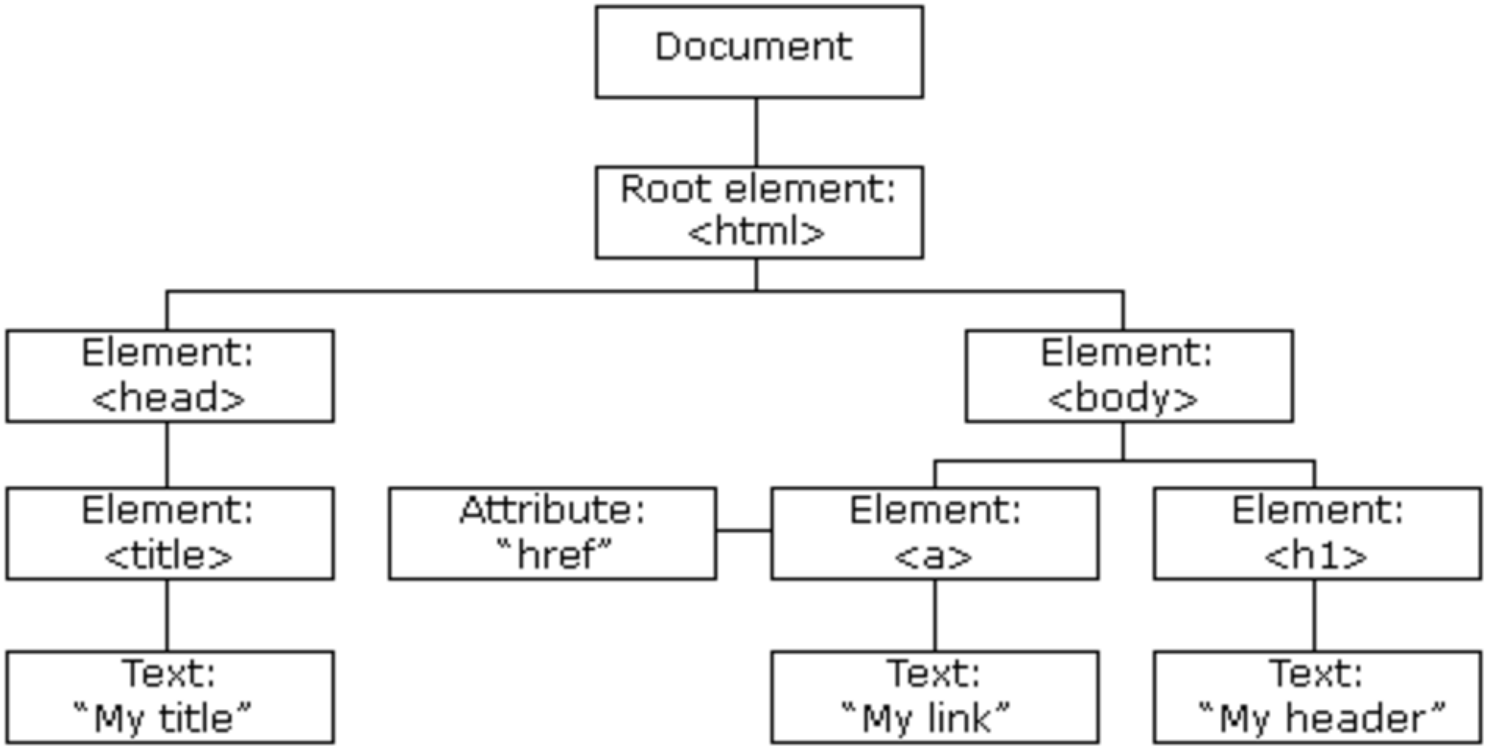
```
GET /cookielogin/cookielogin.php?n=root&p=P%40ssw0rd HTTP/1.1
Host: attack.samsclass.info
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_4) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/52.0.2743.116 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Referer: http://attack.samsclass.info/cookielogin/
Accept-Encoding: gzip, deflate, sdch
Accept-Language: en-US,en;q=0.8
Cookie: __cfduid=d9bcadd4725c25185e7270b90dc73eb101466470070;
password=1mck39g83kgngj30t9gigh4bo0
```

Javascript

- Scripts that run in the client's browser
- Used to validate user-entered data before submitting it to the server
- Dynamically modify UI in response to user action, such as in drop-down menus
- Using Document Object Model (DOM) to control the browser's behavior

Document Object Model DOM

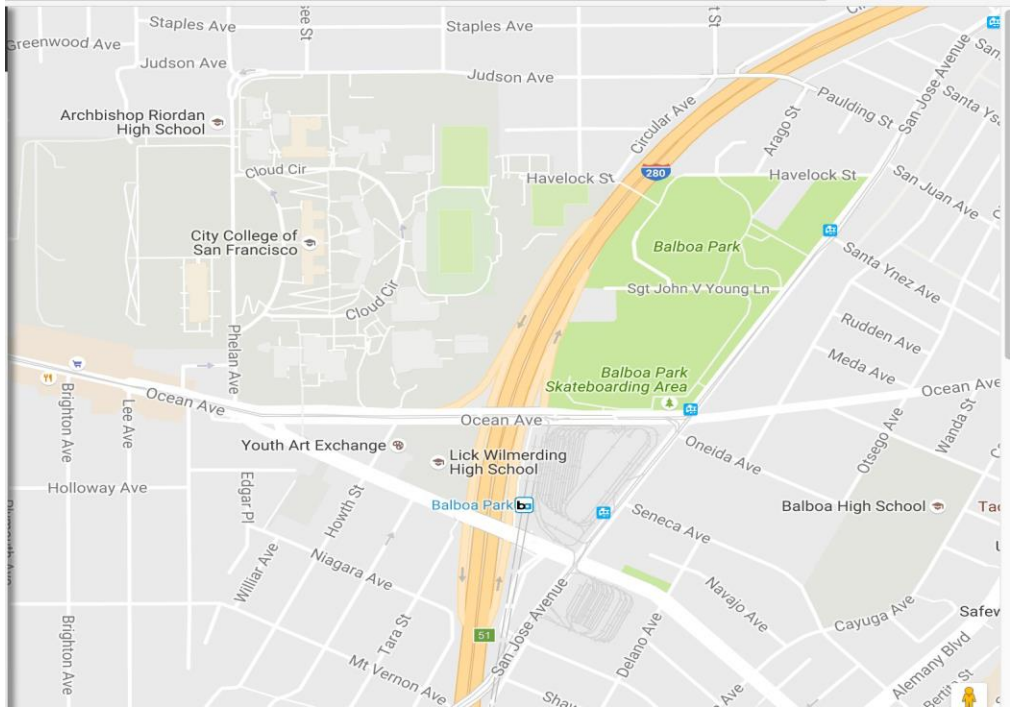
The HTML DOM Tree of Objects



Ajax
Asynchronous JavaScript and XML

- Client-side scripts can fetch data without reloading the entire page
- Allow you to drag Google Maps around

Example



http && frame contains GET

Expression...

No.	Time	Source	Destination	Protocol	Length	Info
80	1…	2601:645:c100:323f:84…	2607:f8b0:4007:8…	HTTP	925	GET /maps/vt?pb=!1m5!1m4!1i17!2i20955!3i50684!4i256!2m3!1e0!2sm!3i36…
81	1…	2601:645:c100:323f:84…	2607:f8b0:4007:8…	HTTP	926	GET /maps/vt?pb=!1m5!1m4!1i17!2i20955!3i50685!4i256!2m3!1e0!2sm!3i36…
82	1…	2601:645:c100:323f:84…	2607:f8b0:4007:8…	HTTP	925	GET /maps/vt?pb=!1m5!1m4!1i17!2i20955!3i50683!4i256!2m3!1e0!2sm!3i36…
83	1…	2601:645:c100:323f:84…	2607:f8b0:4007:8…	HTTP	925	GET /maps/vt?pb=!1m5!1m4!1i17!2i20955!3i50686!4i256!2m3!1e0!2sm!3i36…
84	1…	2601:645:c100:323f:84…	2607:f8b0:4007:8…	HTTP	1006	GET /maps/vt?pb=!1m4!1m3!1i17!2i20955!3i50683!1m4!1m3!1i17!2i20955!3…
252	1…	2601:645:c100:323f:84…	2607:f8b0:4007:8…	HTTP	925	GET /maps/vt?pb=!1m5!1m4!1i17!2i20956!3i50685!4i256!2m3!1e0!2sm!3i36…
253	1…	2601:645:c100:323f:84…	2607:f8b0:4007:8…	HTTP	925	GET /maps/vt?pb=!1m5!1m4!1i17!2i20956!3i50684!4i256!2m3!1e0!2sm!3i36…
254	1…	2601:645:c100:323f:84…	2607:f8b0:4007:8…	HTTP	926	GET /maps/vt?pb=!1m5!1m4!1i17!2i20956!3i50683!4i256!2m3!1e0!2sm!3i36…
255	1…	2601:645:c100:323f:84…	2607:f8b0:4007:8…	HTTP	925	GET /maps/vt?pb=!1m5!1m4!1i17!2i20956!3i50686!4i256!2m3!1e0!2sm!3i36…
256	1…	2601:645:c100:323f:84…	2607:f8b0:4007:8…	HTTP	1005	GET /maps/vt?pb=!1m4!1m3!1i17!2i20956!3i50683!1m4!1m3!1i17!2i20956!3…

JSON
JavaScript Object Notation

- Client-side JavaScript uses the XMLHttpRequest API to request data from a server
- Data is returned in JSON format:

```
{
    "name": "Mike Kemp",
    "id": "8041148671",
    "email": "fkwitt@layerone.com"
}
```

Updating Data with JSON


```
POST /contacts HTTP/1.0
Content-Type: application/x-www-form-urlencoded
Content-Length: 89

Contact={"name":"Mike Kemp","id":"8041148671","email":"
pikey@clappymonkey.com"}
&submit=update
```

Same-Origin Policy

- Prevents content from different origins interfering with each other in a browser
- Content from one website can only read and modify data from the same website
- Ex: scripts on Facebook can't read or write to data on your online banking page
- When this process fails, you get Cross-Site Scripting, Cross-Site Request Forgery, and other attacks

Same-Origin Policy

- A page residing on one domain can cause an arbitrary request to be made to another domain (for example, by submitting a form or loading an image). But it cannot itself process the data returned from that request.
- A page residing on one domain can load a script from another domain and execute this within its own context. This is because scripts are assumed to contain code, rather than data, so cross-domain access should not lead to disclosure of any sensitive information.
- A page residing on one domain cannot read or modify the cookies or other DOM data belonging to another domain.

HTML5

From a security perspective, HTML5 is primarily of interest for the following reasons:

- It introduces various new tags, attributes, and APIs that can be leveraged to deliver cross-site scripting and other attacks, as described in Chapter 12.
- It modifies the core Ajax technology, `XMLHttpRequest`, to enable two-way cross-domain interaction in certain situations. This can lead to new cross-domain attacks, as described in Chapter 13.
- It introduces new mechanisms for client-side data storage, which can lead to user privacy issues, and new categories of attack such as client-side SQL injection, as described in Chapter 13.

Web 2.0

- Heavy use of Ajax for performing asynchronous, behind-the-scenes requests
- Increased cross-domain integration using various techniques
- Use of new technologies on the client side, including XML, JSON, and Flex
- More prominent functionality supporting user-generated content, information sharing, and interaction

Browser Extensions

- Java applets
- ActiveX controls
- Flash objects
- Silverlight objects

- Many security problems
- More and more restricted in modern browsers

State and Sessions

- Stateful data required to supplement stateless HTTP
- This data is held in a server-side structure called a session
- The session contains data such as items added to a shopping cart
- Some state data is stored on the client, often HTTP cookies or hidden form fields