Agenda

- HTTP Request/Response
- Cookies

HTTP Methods: GET vs POST

HTTP works as a request-response protocol between a client and server. Two commonly used methods for a request-response between a client and server are: GET and POST.

- GET - Requests data from a specified resource
- POST - Submits data to be processed to a specified resource

HTTP Headers: Request vs Response

Header	Type	Contents
User-Agent	Request	Information about the browser and its platform
Accept	Request	The type of pages the client can handle
Accept-Charset	Request	The character sets that are acceptable to the client
Accept-Encoding	Request	The page encodings the client can handle
Accept-Language	Request	The natural languages the client can handle
Host	Request	The server's DNS name
Authorization	Request	A list of the client's credentials
Cookie	Request	Sends a previously set cookie back to the server
Date	Both	Date and time the message was sent
Upgrade	Both	The protocol the sender wants to switch to
Server	Response	Information about the server
Content-Encoding	Response	How the content is encoded (e.g., gzip)
Content-Language	Response	The natural language used in the page
Content-Length	Response	The page's length in bytes
Content-Type	Response	The page's MIME type
Last-Modified	Response	Time and date the page was last changed
Location	Response	A command to the client to send its request elsewhere
Accept-Ranges	Response	The server will accept byte range requests
Set-Cookie	Response	The server wants the client to save a cookie

image source: http://www.computing.dcu.ie/~humphrys/Notes/Networks/tanenbaum/7-43.jpg

Exercise: Live HTTP Headers

- A chrome extension
- install it on your host machine
- turn on capture in the extension, go to https://shop.arianagrande.com/ and find one GET request and its corresponding response. Find out information about: User-Agent, Host, Server.
- go to https://shop.arianagrande.com/ and figure out how to generate a post request - the request must go to shop.arianagrande.com.

Exercise: Test HTTP Methods

- type in the following 3 lines from your command line: (Try it 4 times, and each time replace www.victim.com with: www.boisestate.edu, cs.boisestate.edu, www.americanexpress.com, www.arianagrande.com, respectively.)
- nc www.victim.com 80 (press enter here)
- OPTIONS / HTTP/1.1 (press enter here)
- Host: www.victim.com (press enter here)
- press ctrl-c to stop.
- one of the above websites tells you some detailed information about the web server, like software verson, OS, etc. Which one?

Cookie Attributes

- Name:
- Value:
- Domain:
- Path: the Domain and Path attributes define the scope of the cookie.
- Expires/Expiration: states when the cookie should be discarded.
- HostOnly:
- Session: A session cookie, also known as an in-memory cookie or transient cookie, exists only in temporary memory while the user navigates the website. Web browsers normally delete session cookies when the user closes the browser.
- Secure
- HttpOnly

Read/Write Cookies

- Server: using "set-cookie" header
- Client side Javascript: document.cookie. Try this: in your browser address bar, manually type in (no copy and paste): javascript:alert(document.cookie)
- When sending a request to a server, a web browser includes all unexpired cookies whose domains and paths match the requested URL, excluding those marked as secure from the inclusion in an HTTP request.

Exercise: EditThisCookie

- A chrome extension allows us to read/write cookies.
- install it on your host machine
- create a test account on www.arianagrande.com, use EditThisCookie to figure out which cookie or cookies is/are used for this site's authentication.

Cookie Attributes: Domain and Path

- the Domain and Path attributes define the scope of the cookie.
- e.g., the website maroon5.com cannot set a cookie that has a domain of chase.com because this would allow the maroon5.com website to control the cookies of chase.com.
- if a cookie's Domain and Path attributes are not specified by the server, they default to the domain and path of the resource that was requested.

Cookie Attribute: HostOnly

- this flag is true when a cookie is set from the server without specifying a domain.
- HostOnly cookie means that the cookie should be handled by the browser only to the same host/server that firstly sent it to the browser, i.e., the request's host must exactly match the domain of the cookie.
- Without this flag:
- if a cookie is set from boisestate.edu without a domain, the cookie will only be sent for requests to boisestate.edu, it will not be sent for requests to my.boisestate.edu, or any other subdomains of boisestate.edu.
- If the server set a cookie with a domain specified (e.g., the boisestate.edu domain), the cookie will be sent for requests to all other subdomains, such as my.boisestate.edu.

Cookie Attribute: Session

- Identify one cookie (with session flag set) on shop.arianagrande.com, close the tab, open shop.arianagrande.com again and see if that cookie still exists.

Cookie Attribute: Secure

- the browser will only send cookies with the secure flag when the request is going to an HTTPS page. Said in another way, the browser will not send a cookie with the secure flag set over an unencrypted HTTP request.
- By setting the secure flag, the browser will prevent the transmission of a cookie over an unencrypted channel.

Cookie Attribute: HttpOnly

- If the HttpOnly flag is set, the cookie cannot be accessed through client side script (like Javascripts). As a result, even if a cross-site scripting (XSS) flaw exists, and a user accidentally accesses a link that exploits this flaw, the browser will not reveal the cookie to a third party.
- Experiment: On the shop.arianagrande.com page, try this: in your browser address bar, manually type in (no copy and paste): javascript:alert(document.cookie)
- And see if secure_customer_sig is displayed. This is an HttpOnly cookie.