Cookies, Sessions, & Local Storage
Keeping state with distributes systems

Session and State
What’s going on?
• Recall that the HTTP protocol is stateless.
• Each HTTP request is separate and isolated from any other ones.
• How does an application keep track of someone being logged in? User data?
• Options
  • HTTP Cookies
  • Shared Secret / Signed Tokens
  • Local Storage

HTTP Cookies
History
• Cookies were introduced in 1994 with Netscape Navigator
Cookies Preserve State Between Requests

HTTP /login HTTP/1.1
Host: dev.local

Response
Session ID
Session Data
username
session_data
...

Client Browser
Web Server

Cookies
Preserve
State
Between
Requests

Client Browser
Web Server

Cookies
Preserve
State
Between
Requests

Client Browser
Web Server
Cookies Preserve State Between Requests

GET /login HTTP/1.1
Host: dev.local

Session ID
Session Data
username
session_data
...
Cookies Preserve State Between Requests

Client Browser

Web Server

GET /login HTTP/1.1
Host: dev.local

HTTP/1.1 200 OK
content-type: text/html;
content-length: 762
set-cookie: AWSALB=6MUWIBgZmmL
set-cookie: _opensaml=_cf4e13; SameSite=None

<!doctype html>
<html>

<

Cookies Preserve State Between Requests

Client Browser

Web Server

GET /login HTTP/1.1
Host: dev.local

HTTP/1.1 200 OK
content-type: text/html;
content-length: 762
set-cookie: AWSALB=6MUWIBgZmmL
set-cookie: _opensaml=_cf4e13; SameSite=None

<!doctype html>
<html>

<

Cookies Preserve State Between Requests

Client Browser

Web Server

GET /login HTTP/1.1
Host: dev.local

HTTP/1.1 200 OK
content-type: text/html;
content-length: 762
set-cookie: AWSALB=6MUWIBgZmmL
set-cookie: _opensaml=_cf4e13; SameSite=None

<!doctype html>
<html>

<
Cookies Preserve State Between Requests

HTTP/1.1 200 OK
content-type: text/html;
content-length: 762

set-cookie: AWSALB=6MUWIBgZmmL
set-cookie: _opensaml_=cf4e13; SameSite=None

<!doctype html>
<html>

HTTP Cookies
Odds and Ends

- A client cannot request a cookie
- Server decides whether to send a cookie back with a response or not
- Cookies are set with an HTTP response header of `set-cookie`
- Cookies can be set to expire at a given time, or when the browser is closed
- Browser enforces cookie separation by domain
- Cookies can be sent and restricted to `https` requests
- Can be set to exclude from JavaScript access

HTTP Cookies


Set-Cookie: cookie-name=cookie-value
Set-Cookie: cookie-name=cookie-value; Expires=date
Set-Cookie: cookie-name=cookie-value; Max-Age=number
Set-Cookie: cookie-name=cookie-value; Domain=domain-value
Set-Cookie: cookie-name=cookie-value; Path=path-value
Set-Cookie: cookie-name=cookie-value; Secure
Set-Cookie: cookie-name=cookie-value; HttpOnly
Set-Cookie: cookie-name=cookie-value; SameSite=Strict
Set-Cookie: cookie-name=cookie-value; SameSite=Lax
Set-Cookie: cookie-name=cookie-value; SameSite=None; Secure; HttpOnly

// Multiple attributes are also possible, for example:
Set-Cookie: cookie-name=cookie-value; Domain=domain-value; Secure; HttpOnly
HTTP Cookies

D2L Login Example

• Used to track login to an application
• Used to track users across many visits
• Used to track users across many applications
• Used by 3rd party for data tracking

D2L Login Example

HTTP Cookies

Tracking Users Across Sessions

• Cookies can be set for the requested domain by any HTTP response.
• Cookies set by the domain of the parent Document are known as first-party cookies
• Cookies set by domains other than the parent Document are known as third-party cookies
  • The user/browser is the second-party
• Cookies are sent back to the originating domain on future requests to that domain
HTTP Cookies
Tracking Users Across Sessions

- If a service can get its resources in to many web pages, say by offering free image hosting, that service can gain a great deal of information about what sites an individual user visits.
  - User A visited example.com
  - User A then visited bank.com
  - This correlated user data is very valuable
HTTP Cookies

Security

- Cookies are designed to be a trusted way for a host to know that the incoming request should be connected in some way to a previous request.
- This is how state is shared across discrete independent requests.
- If a bad actor can somehow gain access to a cookie value, they can impersonate the real user.
HTTP Cookies

Security

- How does an attacker steal cookies?
- Physical access to devices
- Compromised software on user’s devices
- Exploiting vulnerabilities in a Website to include attacker’s JavaScript code along with authorized code

- Consider a poorly secured comment form
- If comments can be entered and displayed to others, and if the website does not properly sanitize input, an attacker can trick the website into embedding the attacker’s JavaScript code
- Attacker code can now read cookies from the main Document and send them to the Attacker

GET /transfer_money HTTP/1.1
Host: example.com
cookie: SESSION_ID=12345

"OK, I know who you are. Welcome back User B."

This is a really great website!
<script src="attacker.net/b.js">
HTTP Cookies
XSS - Cross Site Scripting Attack

- How do you protect against?
- Set a cookie to only be accessible with HTTP requests

```javascript
set-Cookie: SESSION_ID=12345; HttpOnly
```

- Content Security Policies
  - https://cheatsheetseries.owasp.org/cheatsheets/Cross_Site_Scripting_Prevention_Cheat_Sheet.html