Session Hijacking

Cookies and web authentication

- An *extremely common* use of cookies is to track users who have already authenticated
- If the user already visited `http://website.com/login.html?user=alice&pass=secret` with the correct password, then the server associates a *"session cookie"* with the logged-in user's info
- Subsequent requests include the cookie in the request headers and/or as one of the fields: `http://website.com/doStuff.html?sid=81asf98as8eak`
- The idea is to be able to say "I am talking to the same browser that authenticated Alice earlier."

Cookie Theft

- **Session cookies** are, once again, **capabilities**
  - The holder of a session cookie gives access to a site with the privileges of the user that established that session
- Thus, **stealing a cookie** may allow an attacker to **impersonate a legitimate user**
  - Actions that will seem to be due to that user
  - Permitting theft or corruption of sensitive data

Stealing Session Cookies

- **Compromise** the server or user's machine/browser
- **Predict** it based on other information you know
- **Sniff** the network
- **DNS cache poisoning**
  - Trick the user into thinking you are Facebook
  - The user will send you the cookie

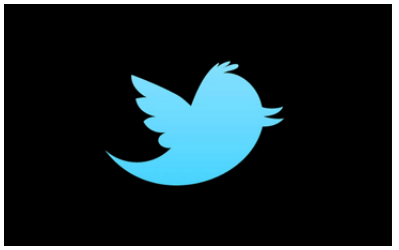
Network-based attacks

Defense: Unpredictability

- **Avoid theft by guessing**; cookies should be
  - **Randomly** chosen,
  - Sufficiently **long**
    (Same goes with hidden field identifiers)

- Can also require separate, **correlating information**
  - Only accept requests due to legitimate interactions with web site (e.g., from clicking links)
    - **Defenses for CSRF**, discussed shortly, **can do this**

Mitigating Hijack



- Sad story: **Twitter**
- Uses one cookie (**auth_token**) to validate user, which is a function of
  - User name, password
- **auth_token** weaknesses
  - *Does not change* from one login to the next
  - *Does not become invalid* when the user logs out
  - Thus: **steal this cookie once**, and you can **log in as the user any time you want** (until password change)!
- **Defense**: **Time out** session IDs and **delete** them once the session ends

http://packetstormsecurity.com/files/119773/twitter-cookie.txt

Non-defense

- **Address-based (non)defense:** Store client IP address for session; if session changes to a different address, must be a session hijack, right?
- **Problem, false positives**: IP addresses change!
  - Moving between WiFi network and 3G network
  - DHCP renegotiation
- **Problem, false negatives**: could be hijacked to different machine with same IP address
  - Both requests via same NAT box