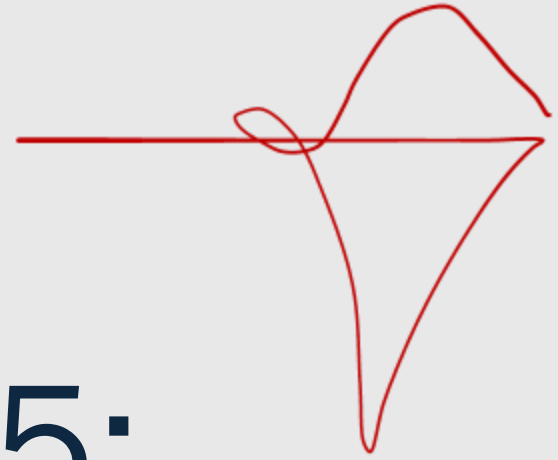


COMP435: *SECURITY CONCEPTS!*

Lecture 22: Cross-Site Forgery Requests

tinyurl.com/comp435-fa25

Final Project/Logistics Updates

- I reviewed everyone's topics! If you received a 0, please submit a regrade with a new topic.
- The outline is due soon! November 12th.
- Next lab is posted. It is about packet sniffing, spoofing, filtering. Due November 17th.
- Today's class will walk through cross-site scripting attacks. This will be the final lab. Out Nov 17th due December 1st.

CROSS SITE FORGERY REQUEST

Cross-Site Request Forgery (CSRF)

Def'n: An attacker injects a transaction request to a site from an authenticated user's browser

a.k.a. XSRF, session riding, "C surf"

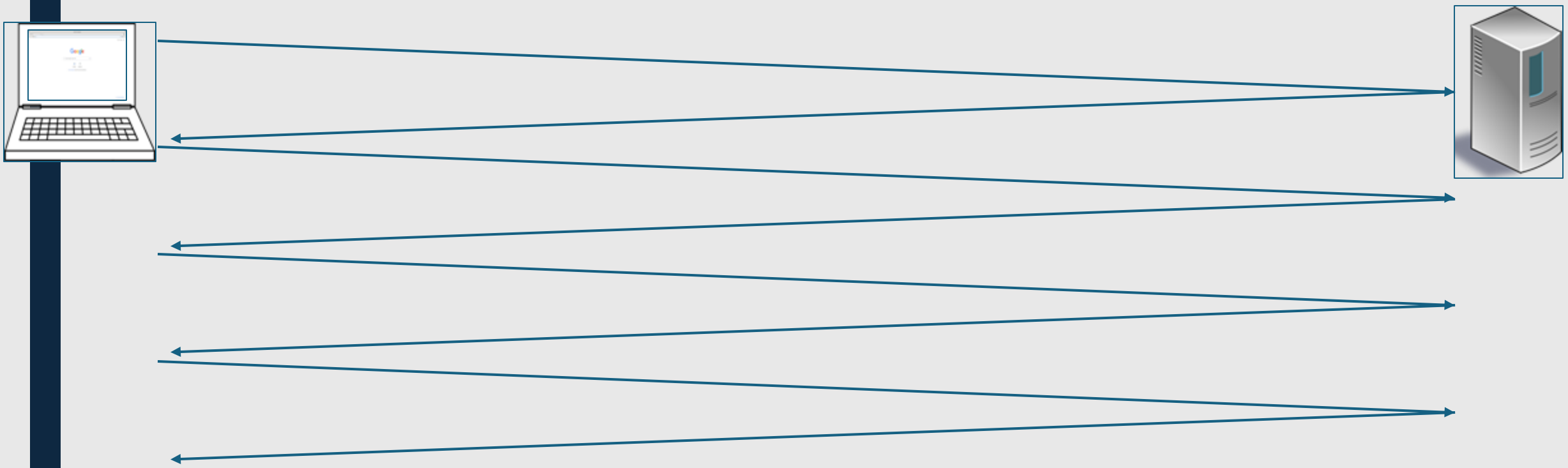
COOKIES

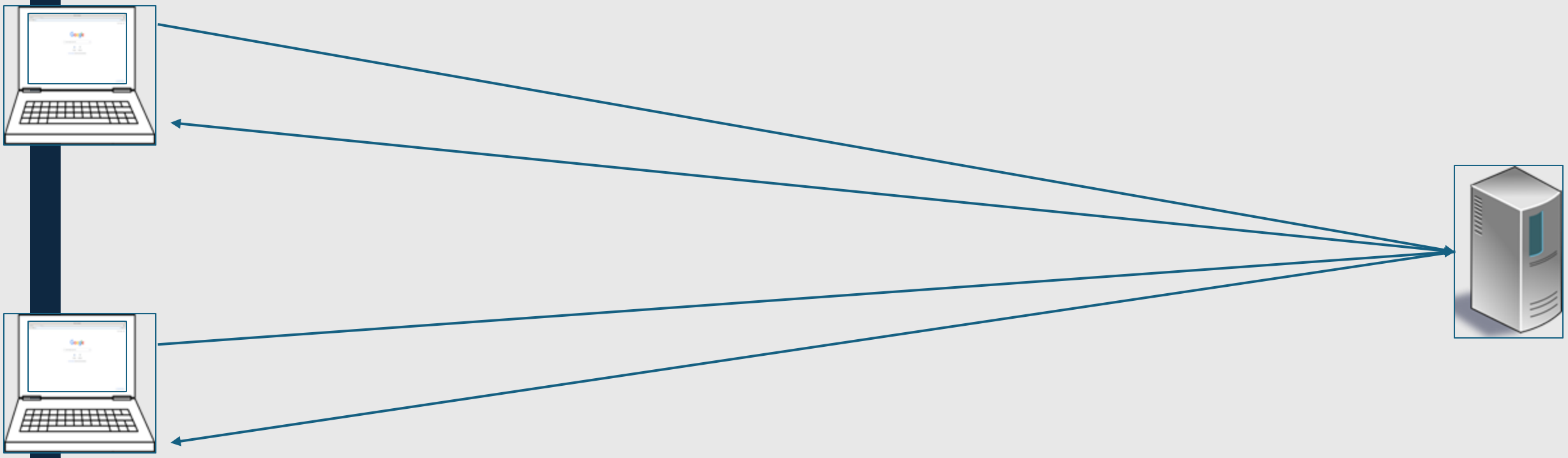
Browser Cookies

Def’n: data strings that enable a stateful browsing session between client and server

E.g., language=english; cartID=ba3fa9dea0;

HTTP is stateless!! So cookies help us keep track of some state ☺





1. browser inits w/ GET req
2. Server responds, sets cookies
3. browser sends cookies in subsequent reqs.

GET www.target.com/s?searchTerm=masks

server sets a cookie to start linking future requests from this client together

HTTP response header contains Set-Cookie keyword followed by the cookie

A cookies just a key-value pair!

HTTP/1.1 200 OK

Set-Cookie: sessionID=9df3ecad22f7
Set-Cookie: userLocation=27599.NC

<html><title>...

GET www.target.com/s?searchTerm=kidsmasks

Cookie:
sessionID=9df3ecad22f7;userLocation=27599.NC

By default, cookies

- can be accessed by JavaScript
- are sent in the clear
- are returned to the requesting server directory

By default, cookies

- can be accessed by JavaScript
- are sent in the clear
- are returned to the requesting server directory

URI setting the cookie:	Cookie returned to:	Cookie not returned to:
sub.site.com/dir/file	sub.site.com/dir/fileA sub.site.com/dir/subdir/fileB	sub.site.com bub.site.com site.com

Cookie WS Problems

Threat Model

- Attacker is not “in the browser”
- Attacker is not eavesdropping on client--server communication
- Attacker can induce victim to click on a link

Alice

bank.com

1. Alice logs into bank.com
2. bank sends session ID to Alice's browser

Alice

bank.com

1. Alice logs into bank.com
2. bank sends session ID to Alice's browser

```
GET http://bank.com/fundxfer.php?to=csturton&value=2500 HTTP/1.1

Cookie: sessionID=2hdkg7s0zwlsl.-S_sfhl25JLLKx
User-Agent: Mozilla/5.0
```

Alice

bank.com

1. Alice logs into bank.com
2. bank sends session ID to Alice's browser

GET http://bank.com/fundxfer.php?to=kakiryan&value=2500 HTTP/1.1

Cookie: sessionID=2hdkg7s0zwlsl.-S_sfhl25JLLKx
User-Agent: Mozilla/5.0

Alice

bank.com

1. Alice logs into bank.com
2. bank sends session ID to Alice's browser

```
GET http://bank.com/fundxfer.php?to=kakiryan&value=2500 HTTP/1.1

Cookie: sessionID=2hdkg7s0zwlsl.-S_sfhl25JLLKx
User-Agent: Mozilla/5.0
```

Possible link locations: on a website Alice visits, embedded in an ad, in an image, in an email or text…

```
<a href="http://bank.com/fundxfer.php?to=kakiryan&value=2500">
Click here … Incredible News!!!
</a>
```

```
<img width="0" height="0" border="0"
src="http://bank.com/fundxfer.php?to=kakiryan&value=2500"/>
```

Notes on CSRF

- Requires that Alice is already logged in to bank.com
- Used for state-changing requests
- Request is indistinguishable from a legitimate request

Notes on CSRF

- Requires that Alice is already logged in to bank.com
- Used for state-changing requests
- Request is indistinguishable from a legitimate request

This is an example of a confused deputy attack!

Not a Defense

- HTTPOnly cookie attribute
- HTTP over TLS (HTTPS)
- Checking the referer header

CSRF Defense: synchronizer tokens

Alice

HTTP/1.1 200 OK

Set-Cookie: sessionID=9df3ecad22f7

<html><body><input type="hidden"
name="csrftoken"
value="bda35f355ffca3fe"/>

bank.com

CSRF Defense: cookie to header

Alice

bank.com

HTTP/1.1 200 OK

Set-Cookie: sessionID=9df3ecad22f7;
csrftoken=bda35f355ffca3fe

<html><body>....

CSRF Defense: cookie to header

Alice

bank.com

POST http://bank.com/fundxferform HTTP/1.1

Cookie: sessionID=9df3ecad22f7; csrftoken=bda35f355ffca3fe
CSRFToken: bda35f355ffca3fe
User-Agent: Mozilla/5.0

....

CSRF Defense: double submitting cookies

Alice

bank.com

HTTP/1.1 200 OK

Set-Cookie: sessionID=9df3ecad22f7;
csrftoken=bda35f355ffca3fe

<html><body><input type="hidden"
name="csrftoken"
value="bda35f355ffca3fe"/>