ACM Cyber

Session Hijacking

Session 3

Overview

1. Review of XSS
2. XSS Demo
3. Quiz time! (cookies to be awarded)
4. What is a session?
5. HTTP and State Management
6. Basic principles of Session Hijacking
7. Quiz time again! (more cookies to be awarded)
8. Defending

XSS Review

- XSS: cross-site scripting

- Using XSS to obtain client cookies
  - Reflective XSS
  - Persistent XSS

What's cross-site scripting again?

- Injecting arbitrary JavaScript code into web applications
- It's generally done to web pages viewable by all users, but theoretically it can be targeted at specific users
  - Example: a website has a Private Message feature that suffers from XSS
- It's called cross-site because scripts not authored by the website are treated by the browser as if it is

What’s Reflective XSS?

- The malicious JavaScript code is sent through an HTTP request to the server
- The server puts the original code directly in the web page
- The browser thinks the code is part of the original page and executes it
- *???*
- **Profit**

What’s Persistent XSS?

- The malicious JavaScript code is sent through an HTTP request to the server
- The server puts the original code directly in the database
- Upon request, the server loads the original code from the database and puts it directly in the web page
- The browser thinks the code is part of the original page and executes it
- *???*
- **Profit**

Reflective vs Persistent XSS

- In reflective XSS, since the malicious code is specific to a request (usually embedded in the URL), it's less powerful:
  - Users can inspect the URL to find the malicious code
  - Browsers can inspect the URL to automatically block reflective XSS
- Neither users nor browsers can tell persistent XSS from genuine code

Demo Recap: DVWA

Go to kahoot.it for a review

What is a Session ID?

When you go to facebook.com, how does Facebook know it’s you?

www.facebook.com/groups/320869398280511/

(7) ACM Cyber

ACM Cyber | Joe | Home | Create

ACM Cyber
Open Group within UCLA

About
Discussion
Announcements
Chats
Members
Events
Photos
Files
Group Insights
Moderate Group

Search this group

Shortcuts
ACM Cyber
UCLA ACM 6

ACM Cyber
Cybersecurity Made Simple

Joined | Notifications | Share | More

Write Post | Add Photo/Video | Live Video | More

Write something...

Photo/Video | Get Together | Poll

ADD MEMBERS
+ Add People from School

MEMBERS

Chat (16)

579 0 0 | Search

Elements | Resources | Network | Timelines | Debugger | Storage | Console

All Storage | Application Cache

Cookies — www.facebook.com
Local Storage — www.facebook.com
Session Storage — www.facebook.com

Name	Value	D	P	E	S	H	S	S
xs	203%3A9siGF...	.f	/	1	E	✓	✓	
wd	1722x1081	.f	/	1	1		✓	
spin	r.4471830_b.t...	.f	/	1	4	✓	✓	
sb	KluDW35I7Ign...	.f	/	E	2	✓	✓	
presence	EDvF3EtimeF1...	.f	/	S	1		✓	
fr	0Uqpxmt16IK...	.f	/	1	E	✓	✓	
dpr	2	.f	/	1	4		✓	
datr	hBSBV1kjT2W...	.f	/	4	2	✓	✓	
c_user	1000030584...	.f	/	1	2		✓	
act	1540673060...	.f	/	S	2		✓	

Filter

All | Errors | Warnings | Logs

Console cleared at 3:15:57 PM

HTTP and State Management

HTTP is stateless

- A HTTP request is followed up with an HTTP response
- You don’t, by default, know whether a HTTP request is sent by the same person
  - Simplifies server programming!

Persistent State Over HTTP

- Uses valid tokens to identify who requesters are
- Server
  - Receives a token (unique identifier)
  - Pull session data from its database or a cache layer
  - (alternatively) the token IS the session data, encrypted

Session Persistence in Cookies

- Cookies are persisted by the browser on each request
- Session IDs are stored in cookies, so they are persisted
- Cookies can be stolen in XSS
- One active session per browser

Session tracking - cookies

First Response

client A

Http Response

HTTP/1.1 200 OK
Location: http://www.abcd.com/login
Set-Cookie: JSESSIONID=09AZ1
Domain=.abcd.com;path=/;HttpOnly
.....

Container

Subsequent Requests

client A

Http Request

POST/login.do HTTP/1.1
Host: www.abcd.com
Cookie: JSESSIONID=09AZ1
.....

Container

Cookies are everywhere

(websites are tracking you!)

After you log in to CCLE, how does CCLE continue to know it’s you?

ccle.ucla.edu/my/

My sites

CCLE

Find a site

Search...

Type: Collab Course

Search: Title Description

Browse by

Subject area

Division

Instructor

Collaboration sites

Elements | Resources | Network | Timelines | Debugger | Storage | Console

All Storage | Application Cache

Cookies — ccle.ucla.edu

Local Storage — ccle.ucla.edu

Session Storage — ccle.ucla.edu

Filter

Cookies

Name	Value	D...	P...	E...	S...	...	S...	S...
MoodleSession	9ccja...	c...	/	S...	3...		✓	
iwe_term_student_urn%3Amace%3Aucla.e...	181	....	/	S...	9...			
iwe_term_enrollment_urn%3amace%3aucl...	19W	....	/	S...	9...			
_shibsession_64656661756c746874747...	_bc0...	c...	/	S...	1...	✓	✓	
_gid	GA1....	....	/	1...	3...			
_gat	1	....	/	1...	5 B			
_ga	GA1....	....	/	1...	2...			
__utma	1255...	....	/	1...	6...			
__unam	6f39...	....	/	1...	3...			

All | Errors | Warnings | Logs

Console opened at 2:52:25 PM

What is Session Hijacking?

tl;dr: Log in as someone else, after stealing their cookies/session ID

Demo: CCLE

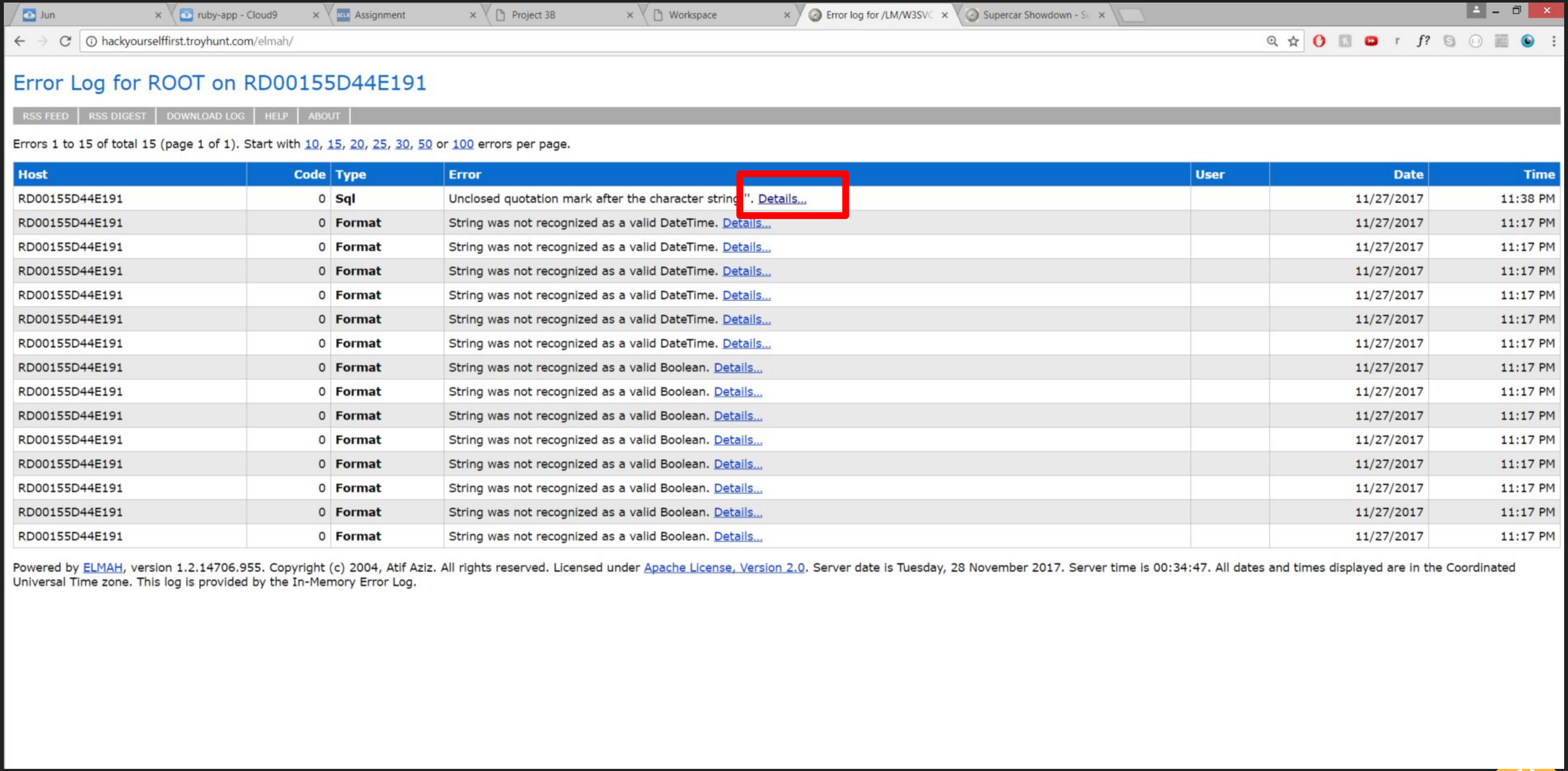
```
document.cookie.split('; ').filter(t => t.startsWith('MoodleSession='))[0]
```

Websites can protect their own cookies from being stolen

- Just like XSS is caused by sloppy coding, cookies that can be stolen is caused by sloppy coding
- Websites (like CCLE) can prevent cookies from being stolen in this way:
  - `Set-Cookie: <cookie-name>=<cookie-value>; HttpOnly`

Go to kahoot.it for a quiz

Cookies may be inadvertently revealed in error logs



Error Log for ROOT on RD00155D44E191

Errors 1 to 15 of total 15 (page 1 of 1). Start with 10, 15, 20, 25, 30, 50 or 100 errors per page.

Host	Code	Type	Error	User	Date	Time
RD00155D44E191	0	**Sql**	Unclosed quotation mark after the character string ''. Details...		11/27/2017	11:38 PM
RD00155D44E191	0	**Format**	String was not recognized as a valid DateTime. Details...		11/27/2017	11:17 PM
RD00155D44E191	0	**Format**	String was not recognized as a valid DateTime. Details...		11/27/2017	11:17 PM
RD00155D44E191	0	**Format**	String was not recognized as a valid DateTime. Details...		11/27/2017	11:17 PM
RD00155D44E191	0	**Format**	String was not recognized as a valid DateTime. Details...		11/27/2017	11:17 PM
RD00155D44E191	0	**Format**	String was not recognized as a valid DateTime. Details...		11/27/2017	11:17 PM
RD00155D44E191	0	**Format**	String was not recognized as a valid DateTime. Details...		11/27/2017	11:17 PM
RD00155D44E191	0	**Format**	String was not recognized as a valid Boolean. Details...		11/27/2017	11:17 PM
RD00155D44E191	0	**Format**	String was not recognized as a valid Boolean. Details...		11/27/2017	11:17 PM
RD00155D44E191	0	**Format**	String was not recognized as a valid Boolean. Details...		11/27/2017	11:17 PM
RD00155D44E191	0	**Format**	String was not recognized as a valid Boolean. Details...		11/27/2017	11:17 PM
RD00155D44E191	0	**Format**	String was not recognized as a valid Boolean. Details...		11/27/2017	11:17 PM
RD00155D44E191	0	**Format**	String was not recognized as a valid Boolean. Details...		11/27/2017	11:17 PM
RD00155D44E191	0	**Format**	String was not recognized as a valid Boolean. Details...		11/27/2017	11:17 PM
RD00155D44E191	0	**Format**	String was not recognized as a valid Boolean. Details...		11/27/2017	11:17 PM

Powered by ELMAH, version 1.2.14706.955. Copyright (c) 2004, Atif Aziz. All rights reserved. Licensed under Apache License, Version 2.0. Server date is Tuesday, 28 November 2017. Server time is 00:34:47. All dates and times displayed are in the Coordinated Universal Time zone. This log is provided by the In-Memory Error Log.

CERT_KEYSIZE	
CERT_SECRETKEYSIZE	
CERT_SERIALNUMBER	
CERT_SERVER_ISSUER	
CERT_SERVER_SUBJECT	
CERT_SUBJECT	
CONTENT_LENGTH	0
CONTENT_TYPE	
GATEWAY_INTERFACE	CGI/1.1
HTTP_ACCEPT	text/html, application/xhtml+xml, */*
HTTP_ACCEPT_ENCODING	gzip, deflate, peerdist
HTTP_ACCEPT_LANGUAGE	en-US
HTTP_CONNECTION	Keep-Alive
HTTP_COOKIE	ASP.NET_SessionId=3qxnhhkjdkfrkw3jey0q1fhs; VisitStart=11/27/2017 11:38:12 PM; ARRAffinity=d756a894d1ec0af96afc15fd205e14a4fbfd70ccc0abc087f24f08f3cff44916; _ga=GA1.2.1560452873.1511822226; _gid=GA1.2.869614182.1511822226; _gat=1
HTTP_DISGUISED_HOST	hackyourselffirst.troyhunt.com
HTTP_HOST	hackyourselffirst.troyhunt.com
HTTP_MAX_FORWARDS	10
HTTP_USER_AGENT	Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0) like Gecko
HTTP_WAS_DEFAULT_HOSTNAME	hackyourselffirst.azurewebsites.net
HTTP_X_ARR_LOG_ID	c35e50a1-5d42-4ade-97f0-dad7080fbbc0
HTTP_X_FORWARDED_FOR	204.11.5.125:14066
HTTP_X_ORIGINAL_URL	/CarsByCylinders?Cylinders=V8'%20or%201=1
HTTP_X_P2P_PEERDIST	Version=1.0
HTTP_X_SITE_DEPLOYMENT_ID	hackyourselffirst
HTTP_X_WAWS_UNENCODED_URL	/CarsByCylinders?Cylinders=V8'%20or%201=1
HTTPS	off
HTTPS_KEYSIZE	
HTTPS_SECRETKEYSIZE	
HTTPS_SERVER_ISSUER	

Browser consoles

Blizzard-Overwatch - Go | Introduction to Session | Home Page | Supercar Showdown - Su

hackyourselffirst.troyhunt.com

Responsive 1280 x 776 75% Online

Supercar Showdown | Leaderboard | Register | Log in | Search

Nissan best car ever

The supercar of the PlayStation generation, Nissan's "Godzilla" is one of the most technically advanced supercars of its era and one of the quickest yet around the Nürburgring.

View the best car ever

Top manufacturers

Who's the best of the best

Nissan: 5 votes

McLaren: 4 votes

Pagani: 3 votes

Elements | Console | Sources | Network | Application

Application
- Manifest
- Service Wo
- Clear stora

Storage
- Local Stora
- Session Sto
- IndexedDB
- Web SQL
- Cookies
  - http://ha

Cache
- Cache Stor
- Application

Frames
- top

Filter

Name	Value	D...	P...	E...	Si...	H...	S...
ASP.NET_Se...	c0mqhj5jymzw12p5p...	h...	/	S...	41	✓	
_ga	GA1.2.789145101.15...	.t...	/	2...	29		
_gat	1	.t...	/	2...	5		
_gid	GA1.2.1413550564.1...	.t...	/	2...	31		

DEMO

Steps

- Go to https://affiliable-forts.000webhostapp.com
- Register an account and log in
- Notice your username on the upper left side of the screen
- Insert an entry into the student database
- Wait for instructions

/

CSRF

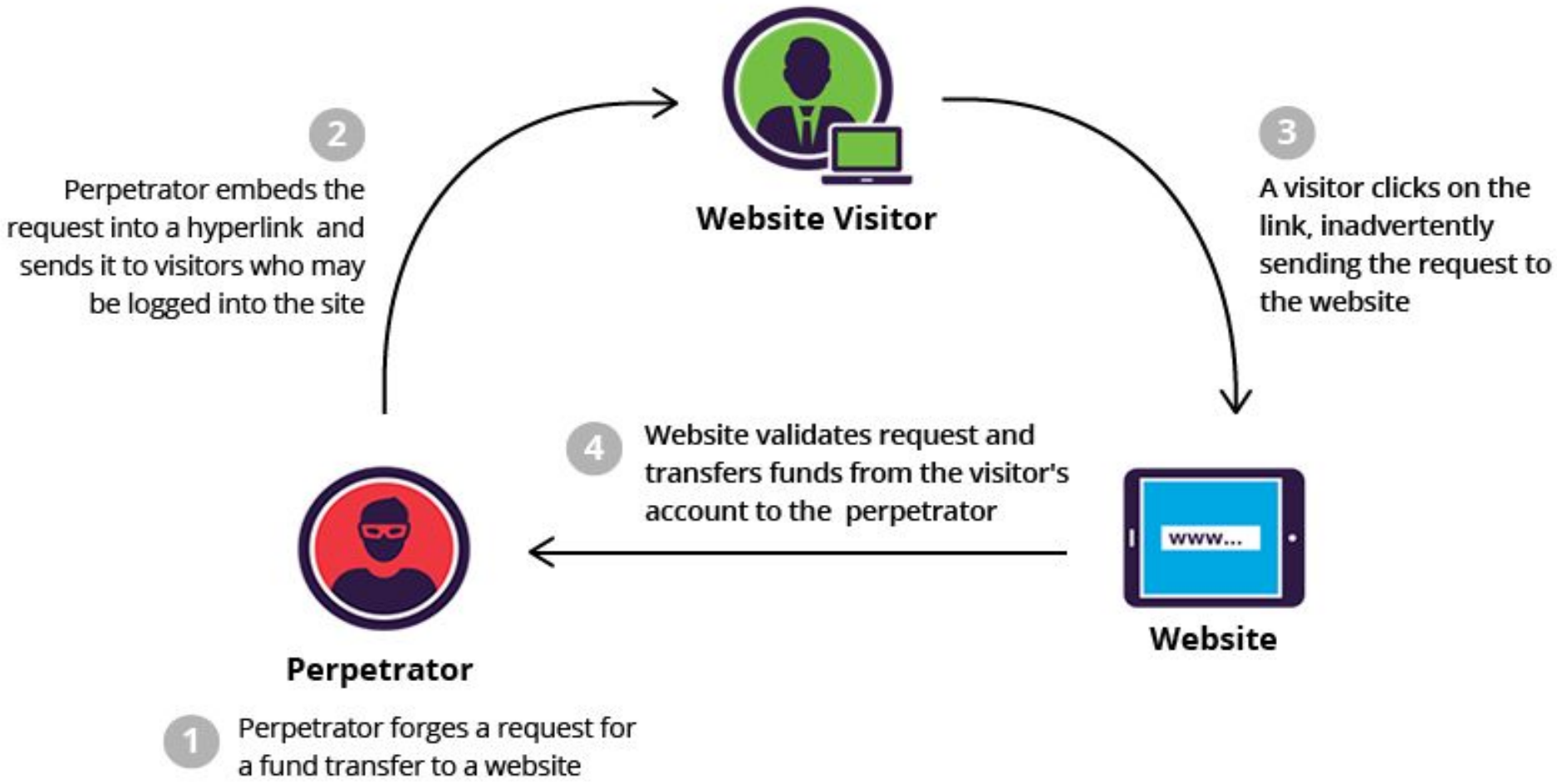
What is Cross Site Request Forgery ?

- Cookies are persisted on each request
- Session IDs can be persistent once the site is left or the browser is closed
- Cookies are vulnerable to XSS (console.log(document.cookie))
- One active session per browser

Hooray we’ve logged in! We’re safe… right?

Attacker forces an end user to execute unwanted actions on a web application in which they're currently authenticated

Target state changing requests


Website Visitor
2
Perpetrator embeds the request into a hyperlink and sends it to visitors who may be logged into the site
3
A visitor clicks on the link, inadvertently sending the request to the website
4
Website validates request and transfers funds from the visitor's account to the perpetrator
www...
Website
Perpetrator
1
Perpetrator forges a request for a fund transfer to a website

GET http://bank.com/transfer.do?acct=BOB&amount=100 HTTP/1.1

http://bank.com/transfer.do?acct=MARIA&amount=100000

```
<a href="http://bank.com/transfer.do?acct=MARIA&amount=100000">
View my Pictures!
</a>
```

What would be an even smarter way to run the exploit without the target realizing?

Defending Against CSRF

Check Origin and Referer Headers

Anti-CSRF Tokens

Re-Auth, OTPs, and Captcha

Same Site Cookies

Interactive XSS Game

Thank you!

tinyurl.com/ydbf929w