HITBSecConf
2022 Singapore

Exploiting Race Condition Vulnerabilities in Web Applications

Javan Rasokat – Application Security Specialist - Sage

HITBSecConf 2022 Singapore

whoami

- Senior Application Security Specialist at Sage
- Ex-Pentester
- Lecturer for Secure Coding at DHBW University, Germany
- M. Sc. IT Security Management at Aalen University (Germany), CISSP, CCSP, CSSLP, GIAC GXPN
- Open Source Projects - https://github.com/JavanXD

Javan Rasokat

Agenda

- Theory
  - What are Race Conditions?
  - Examples
  - Vulnerable PHP Code Snippet
- Vulnerable web app
  - Race condition attack scenarios
  - Secure-SDLC practices
- Attack tool
  - Proposed architecture
  - Research results
- Live Demo
- Conclusion

Theory

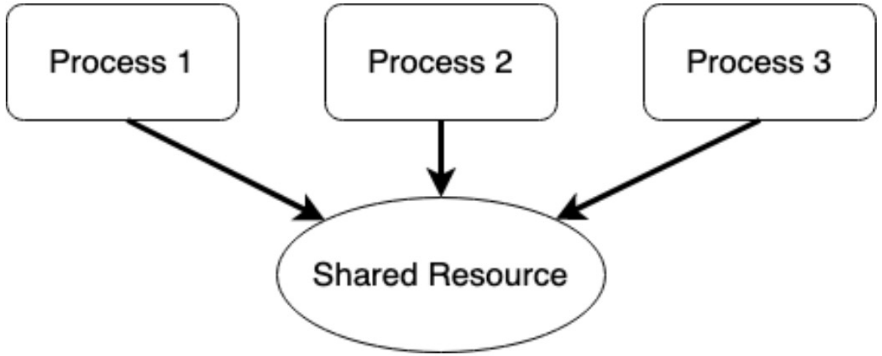
Race Condition – What?

„*A race condition is a flaw that produces an unexpected result when the timing of actions impact other actions.*
An example may be seen on a multithreaded application where actions are being performed on the same data.
Race conditions, by their very nature, are difficult to test for." OWASP [Fou09b]

„*Research Gap*" MITRE [Cor06a]

Race Condition – Again, what?

Multiple threads access shared code, variables, or data simultaneously.



[Pan16]

Knock Knock

Race Condition!

Race Condition!

Race Condition!

Who is there?

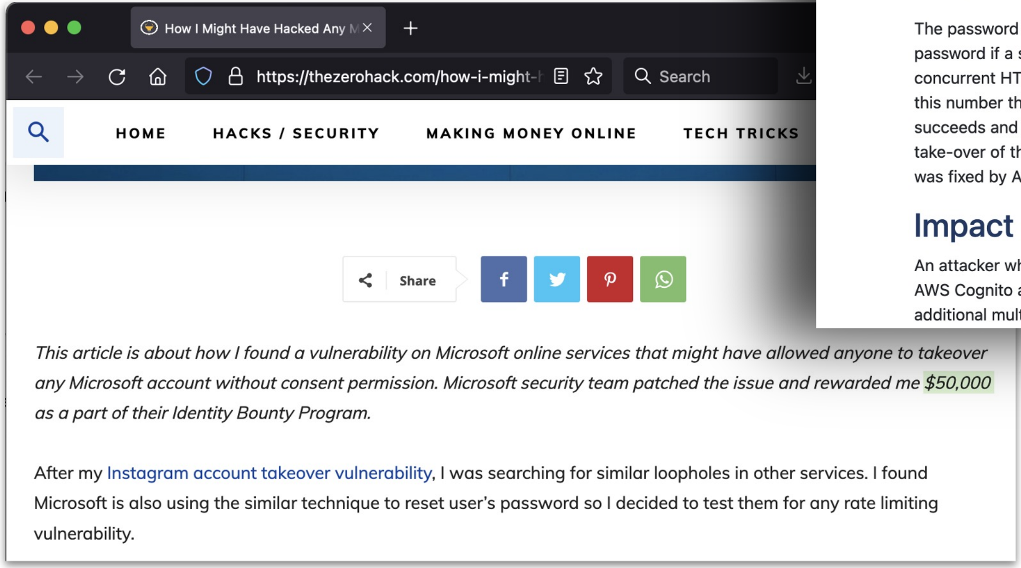
Why do I need to care?

For any actions on your application that may only be allowed to be performed in limited numbers.

- Bypassing anti-brute force mechanisms (e.g., login mechanism).
- Overdrawing limits (e.g., bank account).
- Multiple voting (e.g., online surveys).
- Multiple execution of transfers.
- Generation and redemption of coupon or discount codes.
- Anti-cross-site request forgery (CSRF) tokens.

There are plenty of other scenarios…

Examples


How I Might Have Hacked Any M

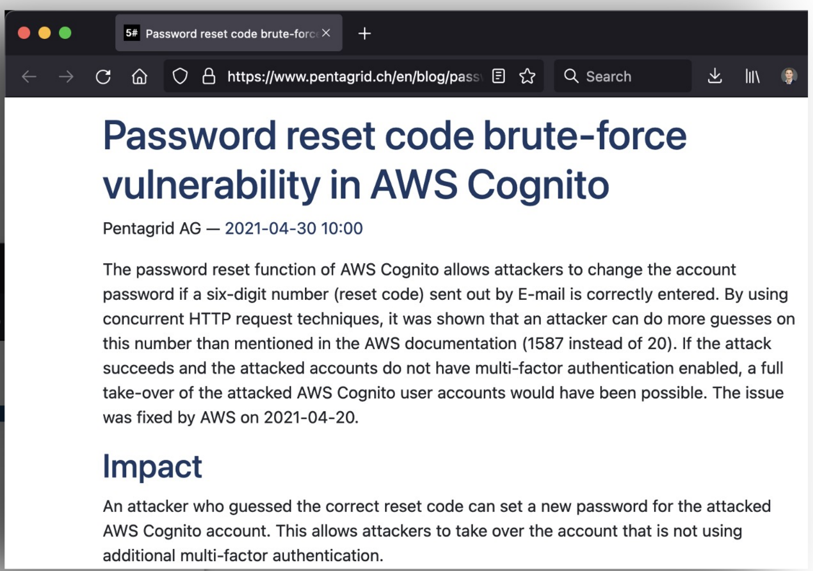
https://thezerohack.com/how-i-might-

HOME HACKS / SECURITY MAKING MONEY ONLINE TECH TRICKS

Share

This article is about how I found a vulnerability on Microsoft online services that might have allowed anyone to takeover any Microsoft account without consent permission. Microsoft security team patched the issue and rewarded me $50,000 as a part of their Identity Bounty Program.

After my Instagram account takeover vulnerability, I was searching for similar loopholes in other services. I found Microsoft is also using the similar technique to reset user's password so I decided to test them for any rate limiting vulnerability.


Password reset code brute-force

https://www.pentagrid.ch/en/blog/pass

Password reset code brute-force vulnerability in AWS Cognito

Pentagrid AG — 2021-04-30 10:00

The password reset function of AWS Cognito allows attackers to change the account password if a six-digit number (reset code) sent out by E-mail is correctly entered. By using concurrent HTTP request techniques, it was shown that an attacker can do more guesses on this number than mentioned in the AWS documentation (1587 instead of 20). If the attack succeeds and the attacked accounts do not have multi-factor authentication enabled, a full take-over of the attacked AWS Cognito user accounts would have been possible. The issue was fixed by AWS on 2021-04-20.

Impact

An attacker who guessed the correct reset code can set a new password for the attacked AWS Cognito account. This allows attackers to take over the account that is not using additional multi-factor authentication.

[Mut21b][Osp21]

Can you spot the race condition?

```php
1 $res = mysql_query('SELECT credit FROM Users WHERE id=$id');
2 $row = mysql_fetch_assoc($res);
3
4 if ($row['credit'] >= $_POST['amount']) {
5     $new_credit = $row['credit'] - $_POST['amount'];
6     $res = mysql_query('UPDATE Users SET credit=$new_credit WHERE id=$id');
7 }
```

Δt = race window [ms]

- Similar code samples can be found in the official PHP-Docs [Ras21]
- several processes could access the resource 'credit' at the same time

$\rightarrow$ How to fix it?

- Lock before line 2 and an unlock after line 6
  - No other thread can access or tamper the values
- Append the condition to the UPDATE: ‘AND credit=$row[‘credit’]’
  - You don’t update the column ‘credit’ if it got tampered
- Use a ‘SELECT FOR UPDATE’ statement if possible

Vulnerable web app

3 Attack scenarios

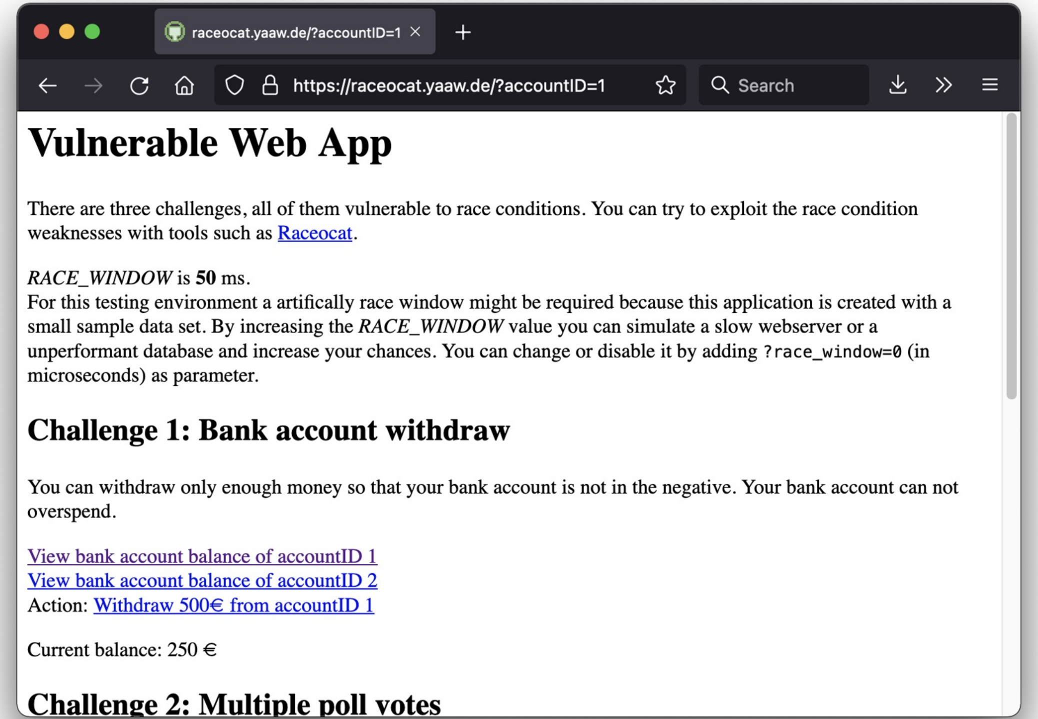
… inspired by real attack scenarios:

- Challenge 1: Bank transfer / withdraw money
  - CVSS Base Score: 6.5 (Medium)
- Challenge 2: Vote submission / "Like" indication
  - CVSS Base Score: 6.5 (Medium)
- Challenge 3: Login using 2-factor authentication
  - CVSS Base Score: 7.5 (High)

Try it by your own

- Open Source on GitHub
- PHP, MariaDB, Docker Compose

https://github.com/JavanXD/Raceocat/


raceocat.yaaw.de/?accountID=1
https://raceocat.yaaw.de/?accountID=1
Search

Vulnerable Web App

There are three challenges, all of them vulnerable to race conditions. You can try to exploit the race condition weaknesses with tools such as Raceocat.

RACE_WINDOW is **50** ms.
For this testing environment a artifically race window might be required because this application is created with a small sample data set. By increasing the *RACE_WINDOW* value you can simulate a slow webserver or a unperformant database and increase your chances. You can change or disable it by adding `?race_window=0` (in microseconds) as parameter.

Challenge 1: Bank account withdraw

You can withdraw only enough money so that your bank account is not in the negative. Your bank account can not overspend.

View bank account balance of accountID 1
View bank account balance of accountID 2
Action: Withdraw 500€ from accountID 1

Current balance: 250 €

Challenge 2: Multiple poll votes

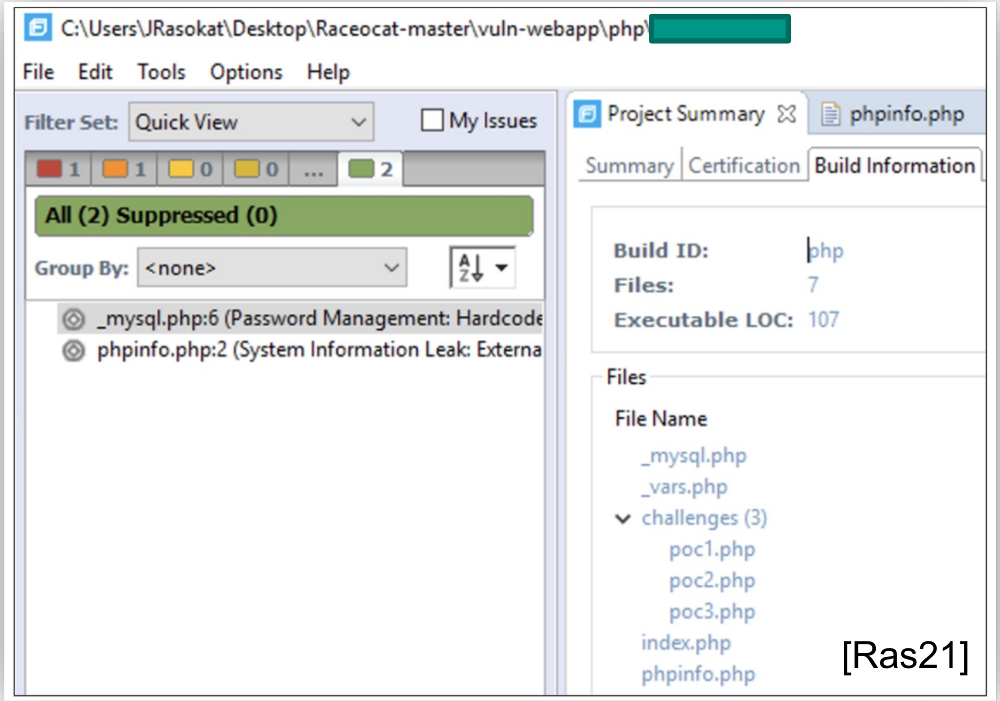
Can we detect or prevent Race Conditions?

- Do any of our Secure-SDLC practices help?

- I tried **hard**....
  - WAF, RASP, SAST, DAST

Conclusion:

- Race condition vulnerabilities go undetected and are exploited despite the deployed in-depth measures.



C:\Users\JRasokat\Desktop\Raceocat-master\vuln-webapp\php\[redacted]

_mysql.php:6 (Password Management: Hardcode
phpinfo.php:2 (System Information Leak: Externa

Build ID:	php
Files:	7
Executable LOC:	107

Files

File Name
- _mysql.php
- _vars.php
- challenges (3)
  - poc1.php
  - poc2.php
  - poc3.php
- index.php
- phpinfo.php

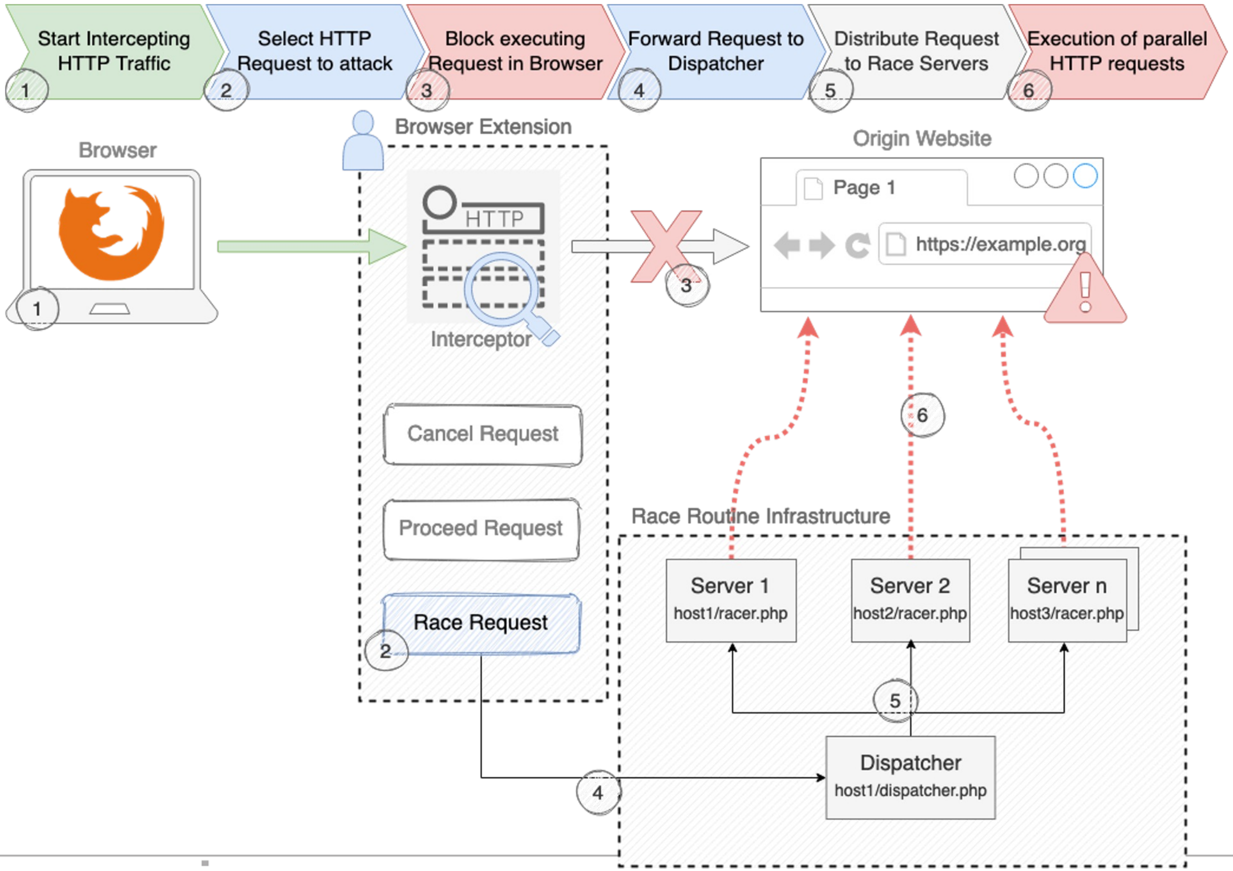
[Ras21]

Attack tool

Current attack tool landscape

- Tools: rc-exploit (2015), Race-the-Web (2016), RacePWN (2017), Sakurity Racer (2017), Burp Turbo Intruder
- Two types of sending parallel requests
  - Parallel
    - Each HTTP-Request in its own connection
    - Often last Byte of the HTTP chunk is sent delayed ("Last Byte-Sync") [LB17]
  - Pipeline
    - Glue multiple HTTP-Requests into one TCP frame/connection
- curl
  - Instead of chaining multiple curl requests (curl & curl & curl...)
  - You can use –parallel/-z and –next flag which got introduced in 2019 with v 7.68.0 [Ste19]

Proposed attack tool architecture

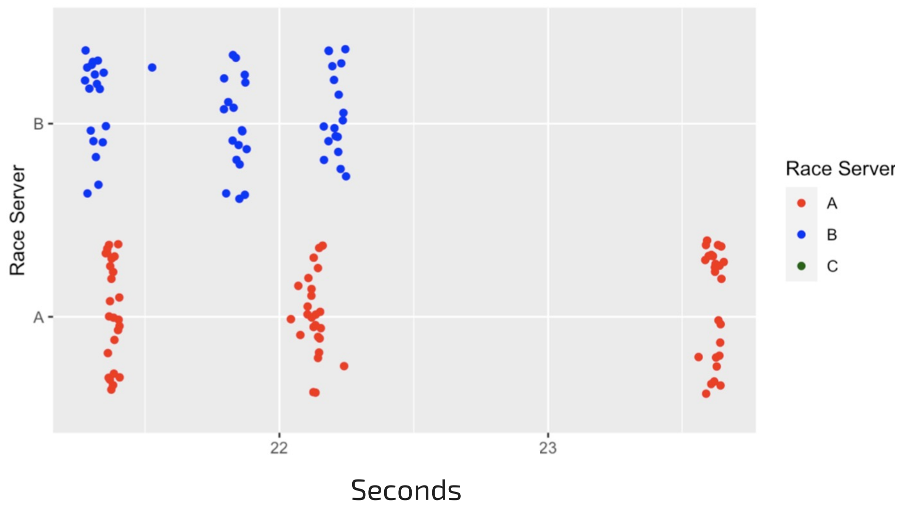


[Ras21]

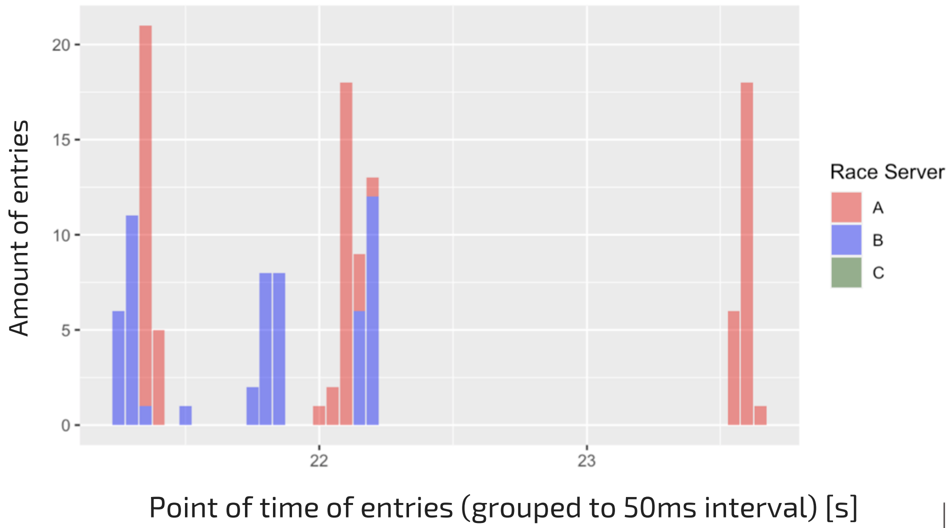
HITBSecConf 2022 Singapore

Research

Origin and Timestamp of HTTP-Requests



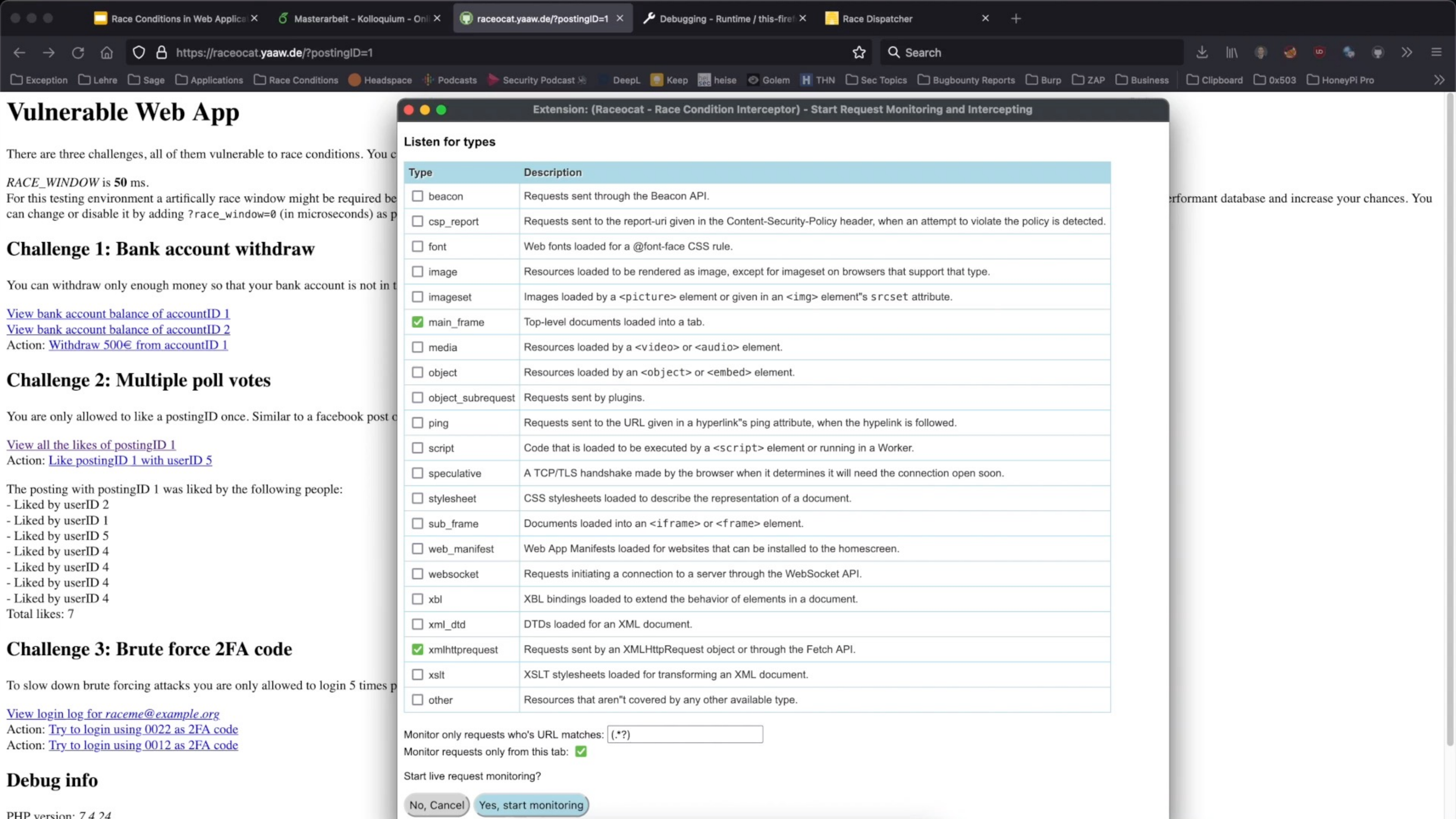
Time Distribution and Origin of HTTP-Requests



Test case: 1.92ms average elapse between processed requests.

[Ras21]

Demo time!



Vulnerable Web App

There are three challenges, all of them vulnerable to race conditions. You c

RACE_WINDOW is **50** ms.
For this testing environment a artifically race window might be required be … rformant database and increase your chances. You can change or disable it by adding `?race_window=0` (in microseconds) as p

Challenge 1: Bank account withdraw

You can withdraw only enough money so that your bank account is not in t

View bank account balance of accountID 1
View bank account balance of accountID 2
Action: Withdraw 500€ from accountID 1

Challenge 2: Multiple poll votes

You are only allowed to like a postingID once. Similar to a facebook post o

View all the likes of postingID 1
Action: Like postingID 1 with userID 5

The posting with postingID 1 was liked by the following people:
- Liked by userID 2
- Liked by userID 1
- Liked by userID 5
- Liked by userID 4
- Liked by userID 4
- Liked by userID 4
- Liked by userID 4
Total likes: 7

Challenge 3: Brute force 2FA code

To slow down brute forcing attacks you are only allowed to login 5 times p

View login log for *raceme@example.org*
Action: Try to login using 0022 as 2FA code
Action: Try to login using 0012 as 2FA code

Debug info

PHP version: 7.4.24

Extension: (Raceocat - Race Condition Interceptor) - Start Request Monitoring and Intercepting

Listen for types

Type	Description
☐ beacon	Requests sent through the Beacon API.
☐ csp_report	Requests sent to the report-uri given in the Content-Security-Policy header, when an attempt to violate the policy is detected.
☐ font	Web fonts loaded for a @font-face CSS rule.
☐ image	Resources loaded to be rendered as image, except for imageset on browsers that support that type.
☐ imageset	Images loaded by a `<picture>` element or given in an `<img>` element"s `srcset` attribute.
✓ main_frame	Top-level documents loaded into a tab.
☐ media	Resources loaded by a `<video>` or `<audio>` element.
☐ object	Resources loaded by an `<object>` or `<embed>` element.
☐ object_subrequest	Requests sent by plugins.
☐ ping	Requests sent to the URL given in a hyperlink"s ping attribute, when the hypelink is followed.
☐ script	Code that is loaded to be executed by a `<script>` element or running in a Worker.
☐ speculative	A TCP/TLS handshake made by the browser when it determines it will need the connection open soon.
☐ stylesheet	CSS stylesheets loaded to describe the representation of a document.
☐ sub_frame	Documents loaded into an `<iframe>` or `<frame>` element.
☐ web_manifest	Web App Manifests loaded for websites that can be installed to the homescreen.
☐ websocket	Requests initiating a connection to a server through the WebSocket API.
☐ xbl	XBL bindings loaded to extend the behavior of elements in a document.
☐ xml_dtd	DTDs loaded for an XML document.
✓ xmlhttprequest	Requests sent by an XMLHttpRequest object or through the Fetch API.
☐ xslt	XSLT stylesheets loaded for transforming an XML document.
☐ other	Resources that aren"t covered by any other available type.

Monitor only requests who's URL matches: (.*?)
Monitor requests only from this tab: ✓

Start live request monitoring?

Conclusion

Conclusion

- Testing needs a good understanding of your business logic
- Sometimes the only way to find them… is a pentest
  - Secure-SDLC practices have not proved to be helpful
  - Spread awareness, include it in your pentesting scope
- Still as mentioned by MITRE a “research gap” [Cor06a]
- Use a distributed attack architecture
  - Find the proposed tool on GitHub: https://github.com/JavanXD/Raceocat/

Thank You!

Bibliography

- [Cor06a] MITRE Corporation. CWE-362: Concurrent Execution using Shared Resource with Improper Synchronization ('Race Condition'). MITRE, CWE-ID CWE- 362. July 2006. url: https://cwe.mitre.org/data/definitions/362.html
- [Fou09b] OWASP Foundation. Testing for Race Conditions (OWASP-AT-010). 2009. url: https://www.owasp.org/index.php/Testing_for_Race_Conditions_(OWASP-AT-010)
- [Hna17b] Aaron Hnatiw. „Racing The Web - Slides“. In: Hackfest, June 2017. url: https://www.slideshare.net/AaronHnatiw/racing-the-web-hackfest-2016
- [LB17] Anton Lopanitsyn und Michail Badin. RacePWN (Race Condition Frame- work). 2017. url: https://github.com/racepwn/racepwn#readme
- [Mut21b] Laxman Muthiyah. How I Might Have Hacked Any Microsoft Account. March 2021. url: https://thezerohack.com/how-i-might-have-hacked-any-microsoft-account
- [Osp21] Tobias Ospelt. Password reset code brute-force vulnerability in AWS Cognito. Pentagrid. April 2021. url: https://www.pentagrid.ch/en/blog/password-reset-code-brute-force-vulnerability-in-AWS-Cognito/
- [Pan16] Sarvesh Pandey. Testing Race Conditions in Web Applications. June 2016. url: https://www.mcafee.com/blogs/technical-how-to/testing-race-conditions-web-applications/
- [Ras21] Javan Rasokat. Master thesis "Race Conditions in Webanwendungen". August 2021. url: https://opus-htw-aalen.bsz-bw.de/frontdoor/deliver/index/docId/1327/file/Rasokat-Race_Conditions_in_Webanwendungen.pdf
- [Ste19] Daniel Stenberg. curl goez parallel. July 2019. url: https://daniel.haxx.se/blog/2019/07/22/curl-goez-parallel/

Backup slides