Web Hacking 101

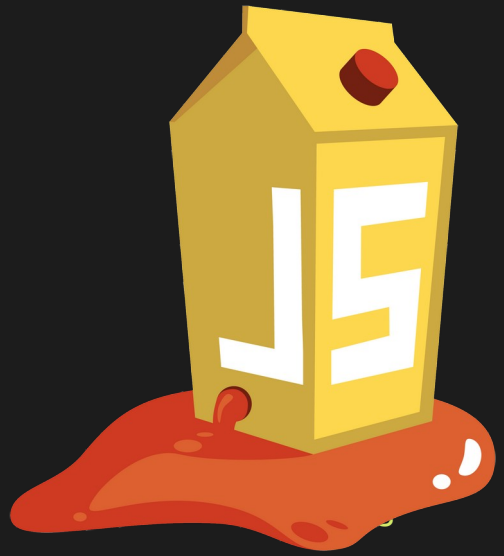
Thursday (6th of July) 19:00 at DC Krov

Why should you learn Web Hacking?

- Modern companies host their services on the web, opposed to creating standalone applications
- Most Bug Bounty programs feature web resources
- Web Hacking is accessible to everyone via browser DevTools (yes, even on windows)
- Unlike working with actually good technology it gets bills paid (see also: javascript)

OWASP Juice Shop

OWASP Juice Shop is probably the most modern and sophisticated insecure web application. Juice Shop encompasses vulnerabilities from the entire OWASP Top Ten along with many other security flaws found in real-world applications!



Installation Steps

I’m going to be using Debian but you can follow those steps anywhere. If you are unable to follow those steps you can use the online version at https://demo.owasp-juice.shop

0. Install node and npm:

$ sudo apt install nodejs && sudo apt install npm

1. Download latest release from https://github.com/juice-shop/juice-shop/releases

2. Unpack with tar/unzip

3. cd into the dir and run npm start

4. Go to http://127.0.0.1:3000/

Web Hacker Tools

FFUF Installation:

```
$ sudo apt install ffuf
```

SecLists Installation:

```
$ git clone
https://github.com/danielmiessler/SecLists.git
```

DevTools

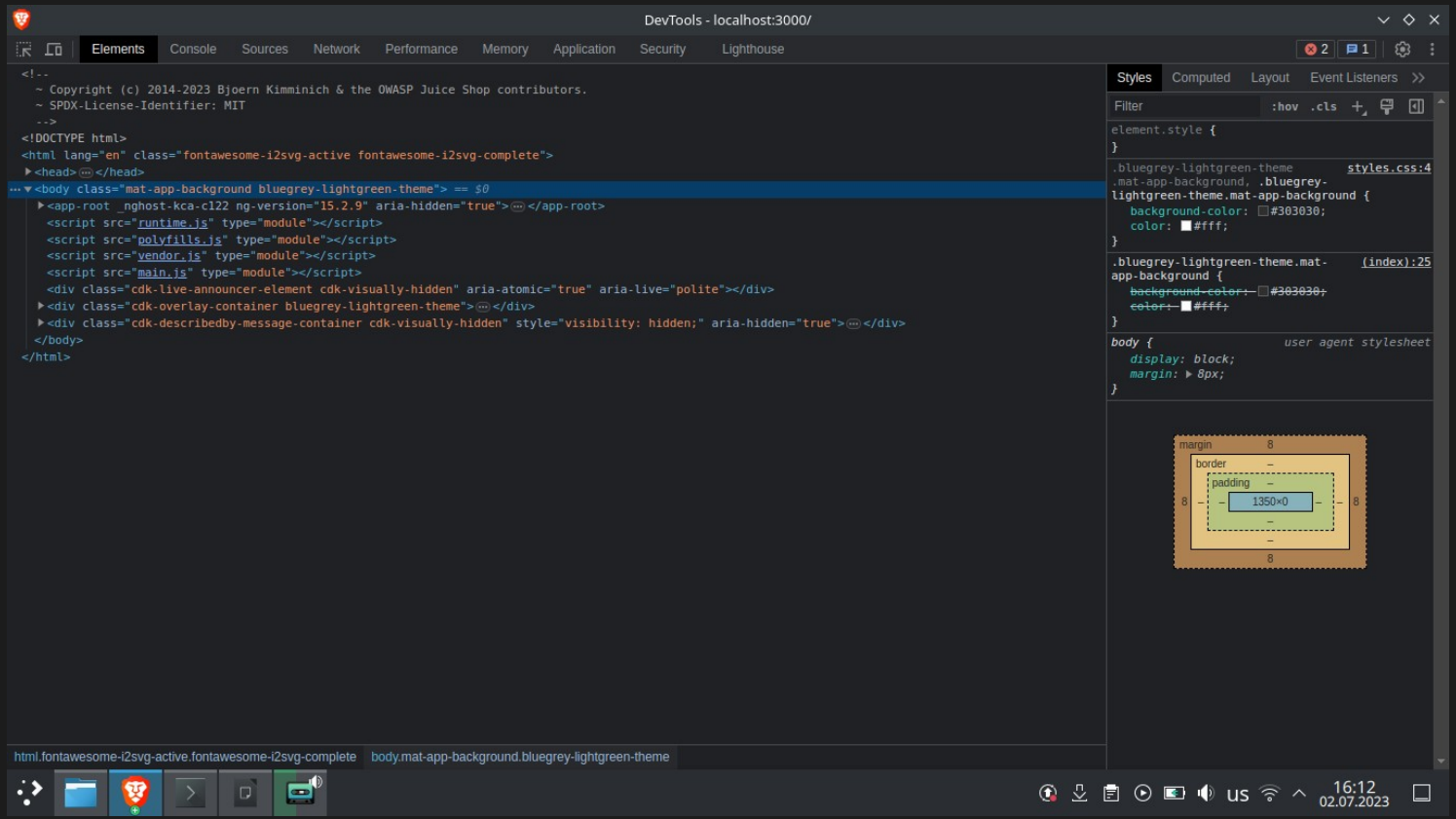
DevTools is going to be our main instrument in analysing websites and finding vulnerabilities.

Shortcut: Ctrl + Shift + I

Mac shortcut: Command + Shift + I

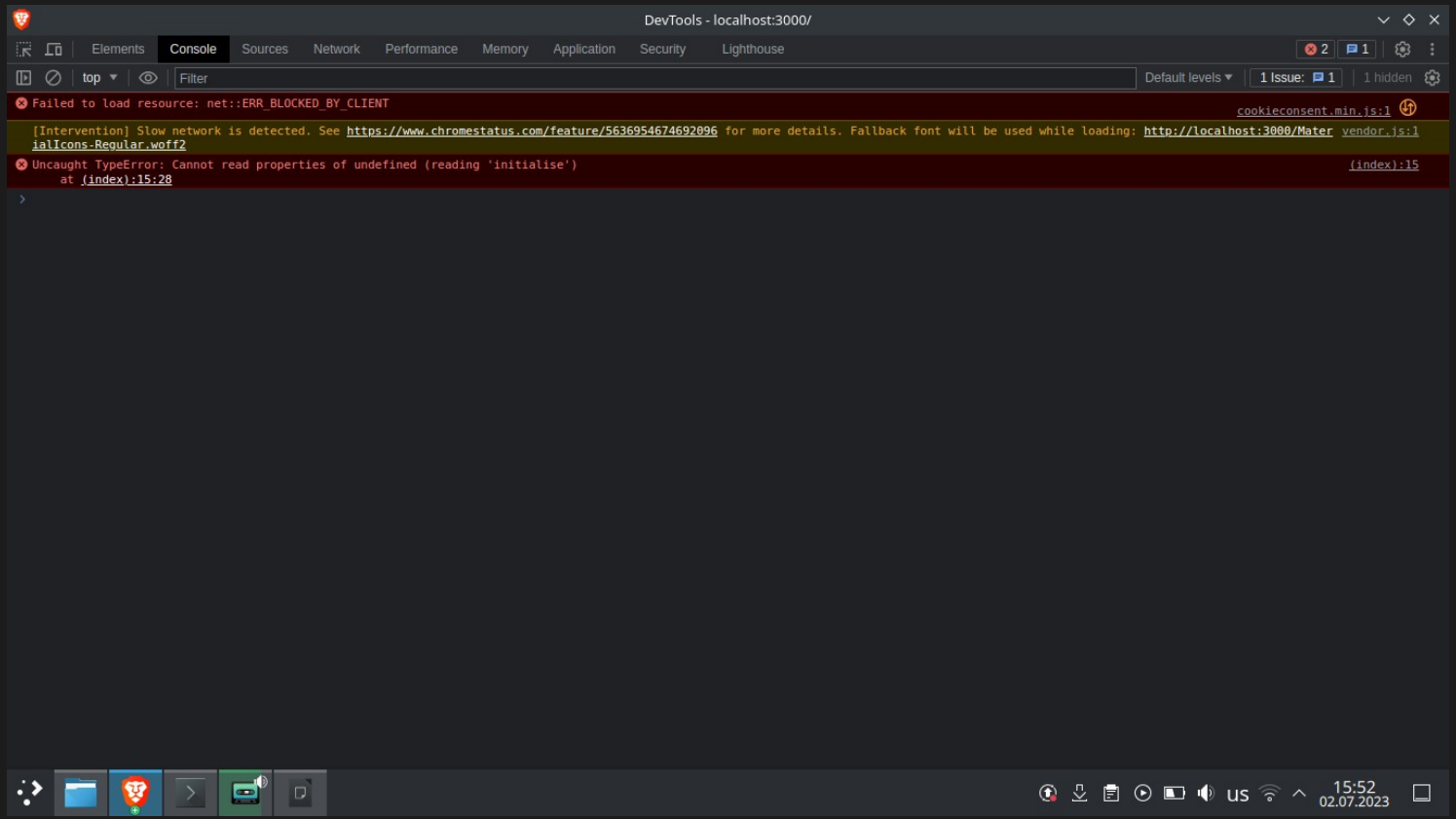
Main DevTools tabs: Elements, Console, Sources, Network, Application

Elements

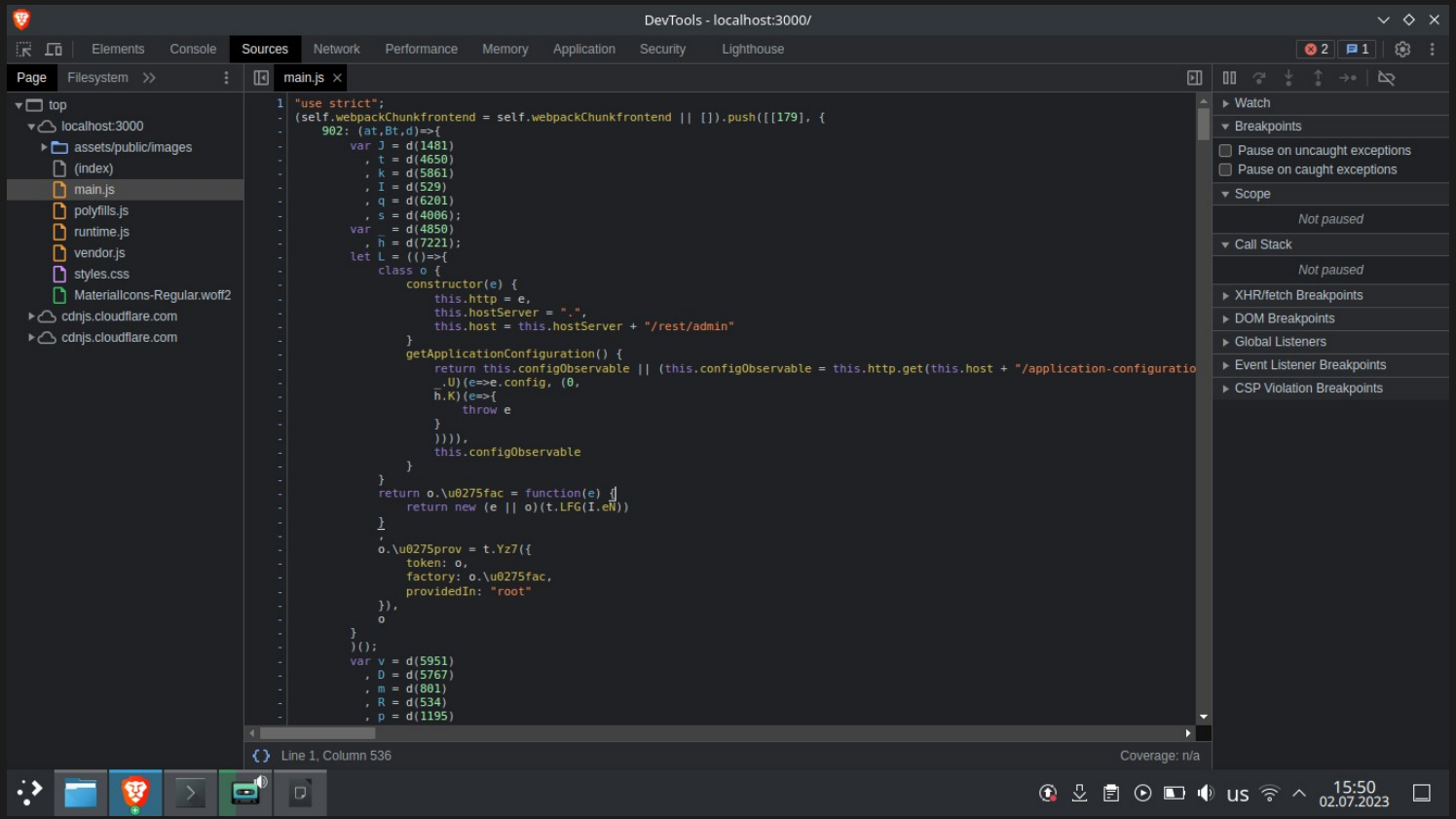

DevTools - localhost:3000/
Elements
Console
Sources
Network
Performance
Memory
Application
Security
Lighthouse
<!--
~ Copyright (c) 2014-2023 Bjoern Kimminich & the OWASP Juice Shop contributors.
~ SPDX-License-Identifier: MIT
-->
<!DOCTYPE html>
<html lang="en" class="fontawesome-i2svg-active fontawesome-i2svg-complete">
<head></head>
<body class="mat-app-background bluegrey-lightgreen-theme"> == $0
<app-root _nghost-kca-c122 ng-version="15.2.9" aria-hidden="true"></app-root>
<script src="runtime.js" type="module"></script>
<script src="polyfills.js" type="module"></script>
<script src="vendor.js" type="module"></script>
<script src="main.js" type="module"></script>

</body>
</html>
Styles
Computed
Layout
Event Listeners
Filter
:hov .cls
element.style {
}
.bluegrey-lightgreen-theme .mat-app-background, .bluegrey-lightgreen-theme.mat-app-background {
background-color: #303030;
color: #fff;
}
styles.css:4
.bluegrey-lightgreen-theme.mat-app-background {
background-color: #303030;
color: #fff;
}
(index):25
body {
display: block;
margin: 8px;
}
user agent stylesheet
margin
8
border
padding
1350×0
html.fontawesome-i2svg-active.fontawesome-i2svg-complete
body.mat-app-background.bluegrey-lightgreen-theme
us
16:12
02.07.2023

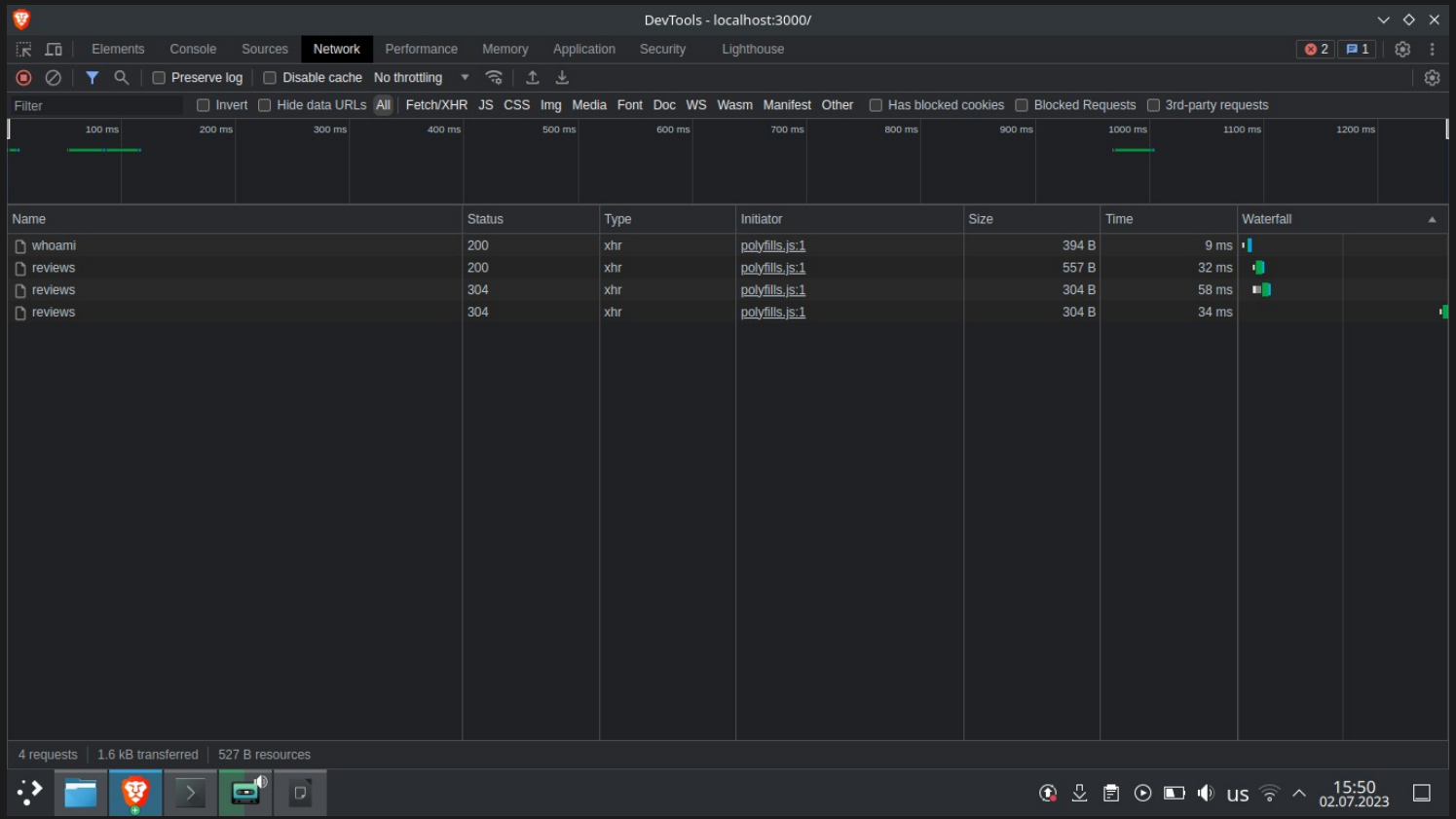
Console Tab


DevTools - localhost:3000/
Elements
Console
Sources
Network
Performance
Memory
Application
Security
Lighthouse
top
Filter
Default levels
1 Issue: 1
1 hidden
Failed to load resource: net::ERR_BLOCKED_BY_CLIENT
cookieconsent.min.js:1
[Intervention] Slow network is detected. See https://www.chromestatus.com/feature/5636954674692096 for more details. Fallback font will be used while loading: http://localhost:3000/Mater
vendor.js:1
ialIcons-Regular.woff2
Uncaught TypeError: Cannot read properties of undefined (reading 'initialise')
(index):15
at (index):15:28
us
15:52
02.07.2023

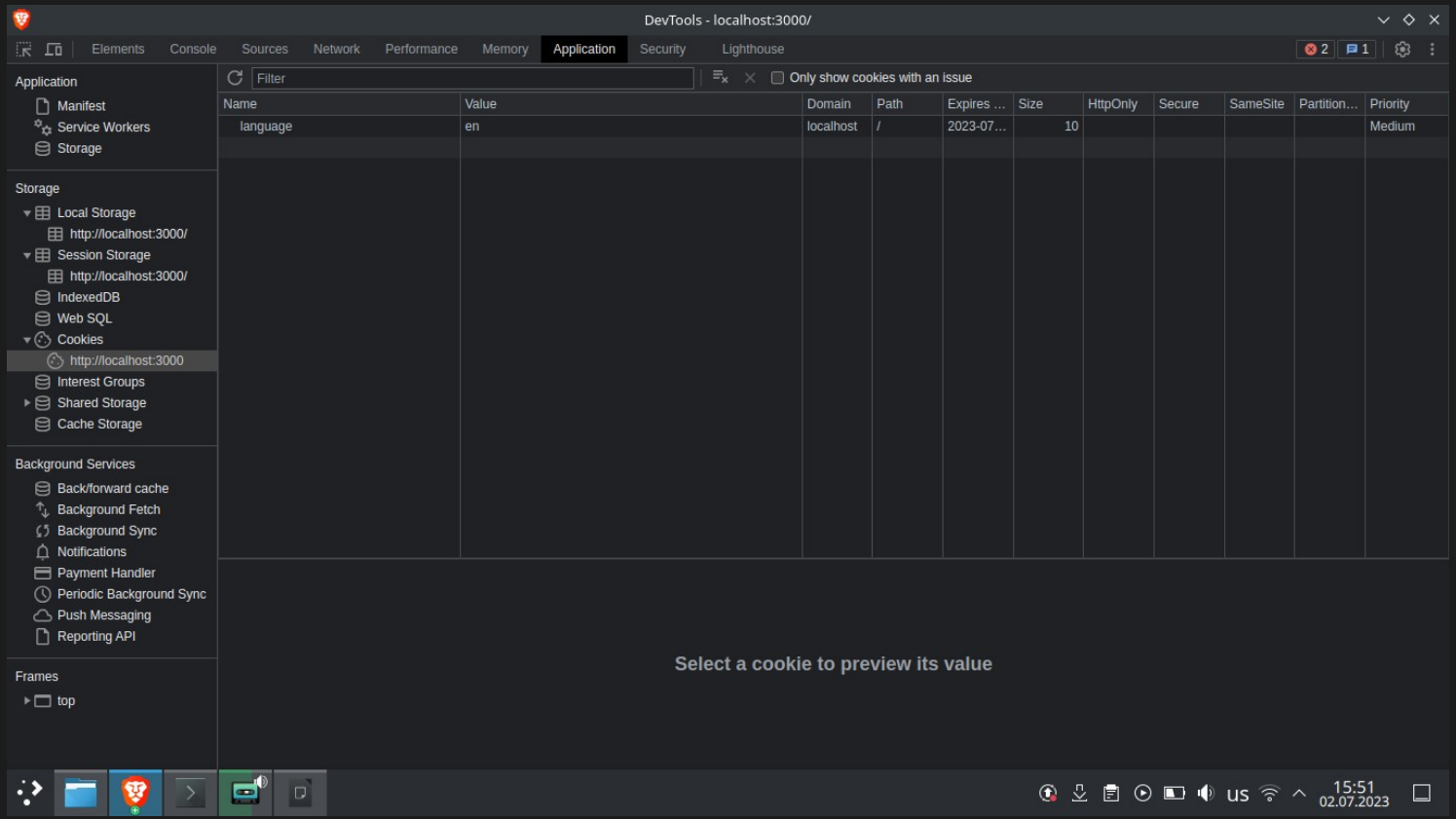
Sources Tab


DevTools - localhost:3000/
Elements
Console
Sources
Network
Performance
Memory
Application
Security
Lighthouse
Page
Filesystem
main.js
top
localhost:3000
assets/public/images
(index)
main.js
polyfills.js
runtime.js
vendor.js
styles.css
MaterialIcons-Regular.woff2
cdnjs.cloudflare.com
cdnjs.cloudflare.com
"use strict";
(self.webpackChunkfrontend = self.webpackChunkfrontend || []).push([[179], {
902: (at,Bt,d)=>{
var J = d(1481)
, t = d(4650)
, k = d(5861)
, I = d(529)
, q = d(6201)
, s = d(4006);
var _ = d(4850)
, h = d(7221);
let L = (()=>{
class o {
constructor(e) {
this.http = e,
this.hostServer = ".",
this.host = this.hostServer + "/rest/admin"
}
getApplicationConfiguration() {
return this.configObservable || (this.configObservable = this.http.get(this.host + "/application-configuratio
.U)(e=>e.config, (0,
h.K)(e=>{
throw e
}
)))),
this.configObservable
}
}
return o.\u0275fac = function(e) {
return new (e || o)(t.LFG(I.eN))
}
,
o.\u0275prov = t.Yz7({
token: o,
factory: o.\u0275fac,
providedIn: "root"
}),
o
}
)();
var v = d(5951)
, D = d(5767)
, m = d(801)
, R = d(534)
, p = d(1195)
Line 1, Column 536
Coverage: n/a
Watch
Breakpoints
Pause on uncaught exceptions
Pause on caught exceptions
Scope
Not paused
Call Stack
Not paused
XHR/fetch Breakpoints
DOM Breakpoints
Global Listeners
Event Listener Breakpoints
CSP Violation Breakpoints
us
15:50
02.07.2023

Network Tab


DevTools - localhost:3000/
Elements
Console
Sources
Network
Performance
Memory
Application
Security
Lighthouse
Preserve log
Disable cache
No throttling
Filter
Invert
Hide data URLs
All
Fetch/XHR
JS
CSS
Img
Media
Font
Doc
WS
Wasm
Manifest
Other
Has blocked cookies
Blocked Requests
3rd-party requests
100 ms
200 ms
300 ms
400 ms
500 ms
600 ms
700 ms
800 ms
900 ms
1000 ms
1100 ms
1200 ms
Name
Status
Type
Initiator
Size
Time
Waterfall
whoami 200 xhr polyfills.js:1 394 B 9 ms
reviews 200 xhr polyfills.js:1 557 B 32 ms
reviews 304 xhr polyfills.js:1 304 B 58 ms
reviews 304 xhr polyfills.js:1 304 B 34 ms
4 requests
1.6 kB transferred
527 B resources
us
15:50
02.07.2023

Application Tab



Access control vulnerability

Access control enforces policy such that users cannot act outside of their intended permissions. Failures typically lead to unauthorized information disclosure, modification, or destruction of all data or performing a business function outside the user's limits.

Business Logic Vulnerabilities

Weaknesses in this category identify some of the underlying problems that commonly allow attackers to manipulate the business logic of an application. Errors in business logic can be devastating to an entire application. They can be difficult to find automatically, since they typically involve legitimate use of the application’s functionality. However, many business logic errors can exhibit patterns that are similar to well-understood implementation and design weaknesses.

Injection

Injection is a bug that happens when user input gets processed without proper validation and sanitization. There are client side injections, such as XSS, as well as server side injections like SQLi and XXE.

Happy hacking!