



Analyzing JS files

BY UNCLE RAT

Agenda

- ▶ What is a JS files
- ▶ Show me your secrets!
- ▶ Attack strategy
- ▶ Defense mechanisms



What is a JS files



What is a JS files



- ▶ Client side object oriented scripting language
- ▶ Makes static websites dynamic
- ▶ Can change a website without reloading it

Show me
your secrets!



Show me your secrets!

- ▶ New endpoints
- ▶ Hidden parameters
- ▶ API keys
- ▶ Business logic
- ▶ HTML/Javascript sinks
- ▶ Secrets/passwords
- ▶ Potentially dangerous areas in code (eval, dangerouslySetInnerHTML etc)

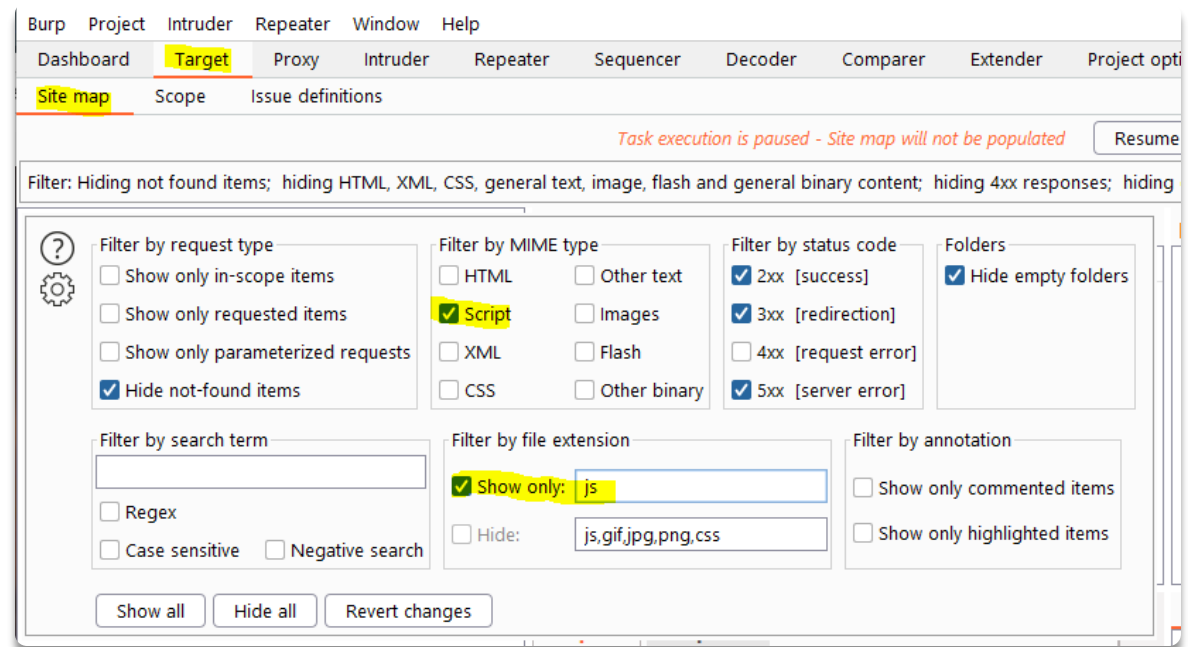


Attack strategy



Attack strategy – Gathering JS files

- ▶ Use burp filters
 - ▶ Explore the website with burp in the background
 - ▶ Open site map
 - ▶ Set filter



Attack strategy – Gathering JS files

- ▶ Use burp suite PRO
 - ▶ Explore the application with burp open
 - ▶ Right click the target in site map
 - ▶ Engagement tools > Find scripts

The screenshot shows the Burp Suite interface. At the top, there are tabs for 'Dashboard', 'Target', 'Proxy', 'Intruder', 'Repeater', 'Sequencer', and 'Decoder'. Below these are 'Site map', 'Scope', and 'Issue definitions'. A filter is applied: 'Filter: Hiding not found items; hiding HTML, XML, CSS, general text, image, flash and general binary content; hid'. The Site Map tree shows three hosts: 'https://cdn.ebayclassifieds.net', 'https://www.google-analytics.com', and 'https://www.google.com'. The 'https://www.google-analytics.com/' host is selected, and a context menu is open. The 'Engagement tools' option is highlighted, and a sub-menu is displayed with 'Find scripts' highlighted. Other options in the sub-menu include 'Search', 'Find comments', 'Find references', 'Analyze target', 'Discover content', 'Schedule task', and 'Simulate manual testing'. The main table below the Site Map shows a request to 'https://www.google-analytics.com/.../analytics.js' with a GET method.

URL	Method	Content
https://www.google-analytics.com/.../analytics.js	GET	...

Defense mechanisms



Defense mechanisms

- ▶ JS minification
 - ▶ Decompress using UglifyJS
- ▶ JS obfuscation
 - ▶ There is no one-size fits all solution
- ▶ JS chunking
 - ▶ Manual de-chunking

