CHAMELEON

FOR OUR SMARTER WORLD

Discovering Hidden Files & Vulnerabilities

Adam Sarin
217342706

Contents

Introduction:

Using OWASP ZAP and Burp-suite I will be trying to identify firstly files that are not normally accessible, We will do this using the Spidering tools offered both by Burp-suite and ZAP, and after using ZAP we will identify vulnerabilities detected by its scan.

Tools used:

- OWASP ZAP
- Burp-suite (version 1.7.36)
- Kali Linux

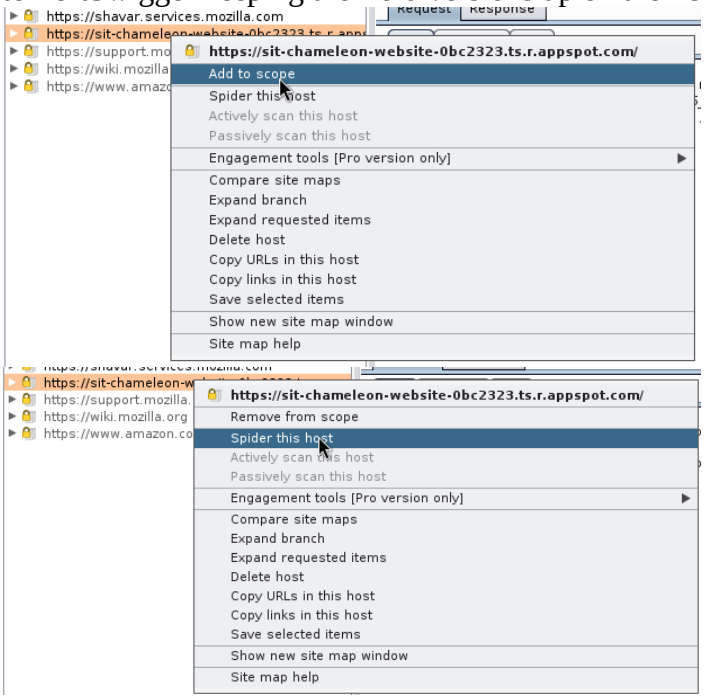
Scope

The scope of all testing will be exclusively in-regards to the Chameleon Site, as these tests are done as a user, an outsider looking in, everything tested or found is done so without the help or instruction of the administrators of the site, and done without an understanding of the actual workings of the site being known beforehand.

Site: https://sit-chameleon-website-0bc2323.ts.r.appspot.com/

Methodology

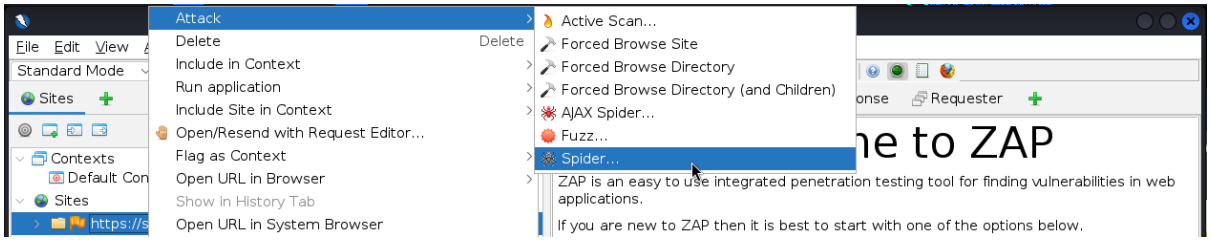
Firstly, testing was done with Burp-Suite, which I uninstalled my newest version of and instead installed an older version from 2018 (1.7.36) the reason being that the new Burp-Suite has moved all the spidering tools into their professional scan, which means Its not available in the free community version but in their paid version, but installation was a non-issue due to PortSwigger keeping their old versions up on their site.



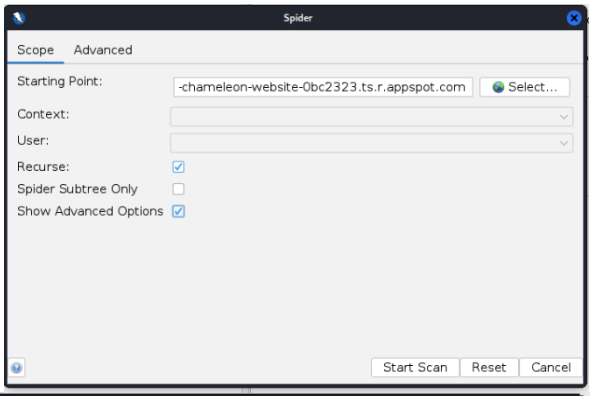
Spidering was quite simple with the old version, requiring you to just add the site to scope and then starting the spidering process with a simple option in the context menu.

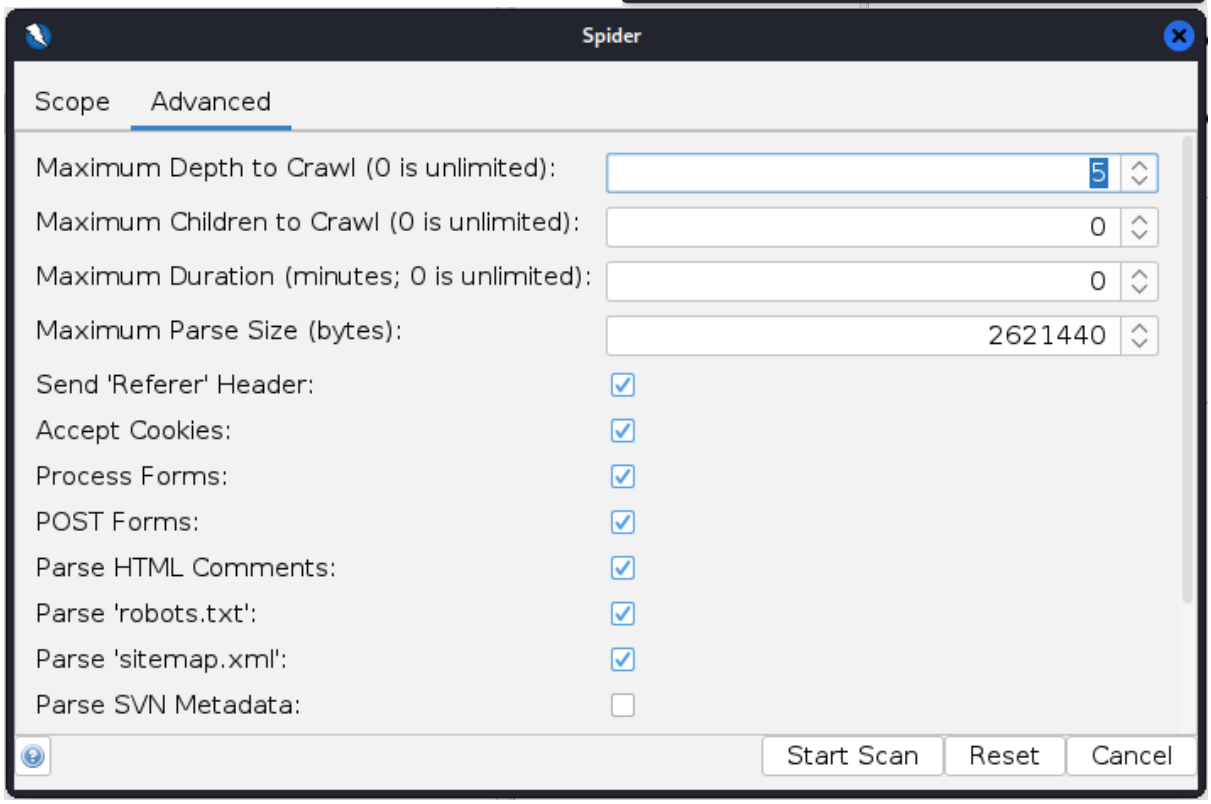
For reference, spidering or “web crawling” is when a program or bot attempts to catalogue the pages and files that are accessible from a website, search engines use their own form of web crawlers to index sites as well as pages that are not on the robots.txt file, however tools like burp-suite and ZAP have options to ignore normal spider limitations.

In OWASP ZAP the same tools are available with some additional features too, as well since ZAP is a free tool the latest version was used.

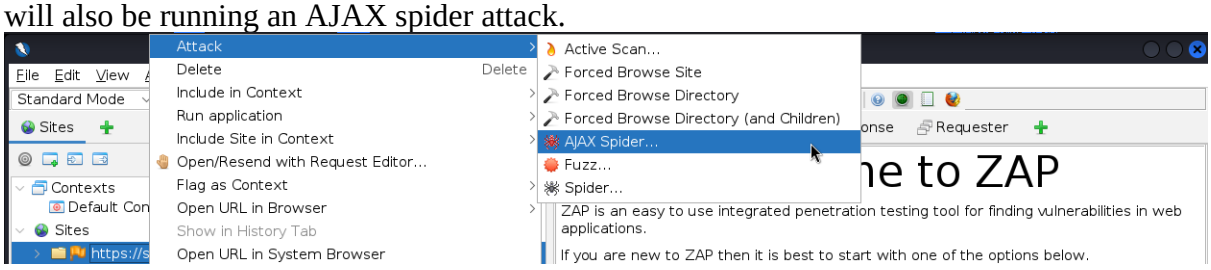


To initiate the spider search right-clicking the site in question and navigating to the Attack tab provides us with the ability to start a multitude of attack types that we will be using to identify as much as possible. The spider attack even has advanced options to configure just how far it will crawl and if it will attempt to read the robots and sitemap files to locate additional links, but for our testing we will keep everything default.

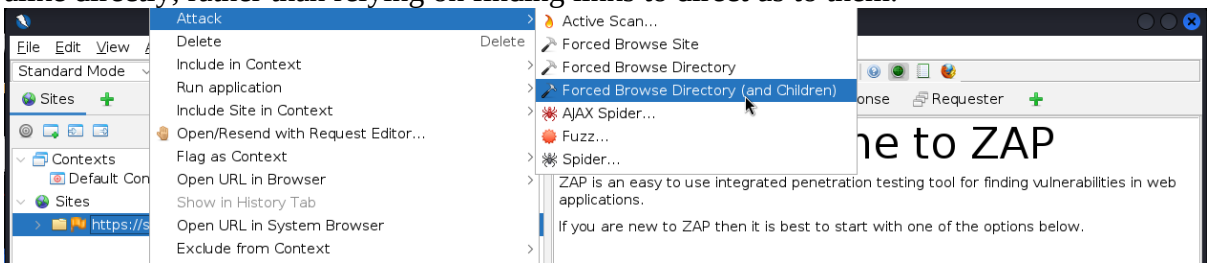




Due to some information provided by ZAP which will be addressed in the results section we will also be running an AJAX spider attack.

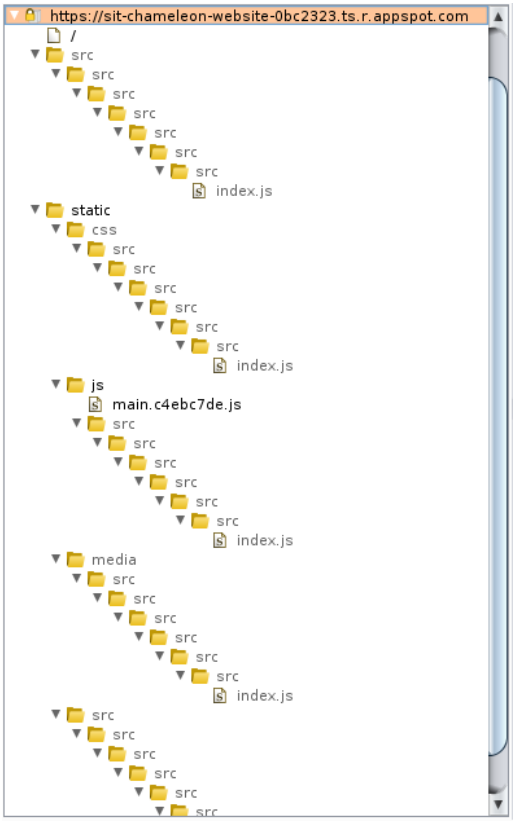


And to cover all bases we will also be using a feature that I do believe is available in Burp-Suite's scanner called "Content Discovery" but as it is a Pro only tool instead we will be using the ZAP alternative which is a "Forced Browse Directory" attack, which uses a list included with ZAP of file and directory names, and attempts to access files and directories alike directly, rather than relying on finding links to direct us to them.

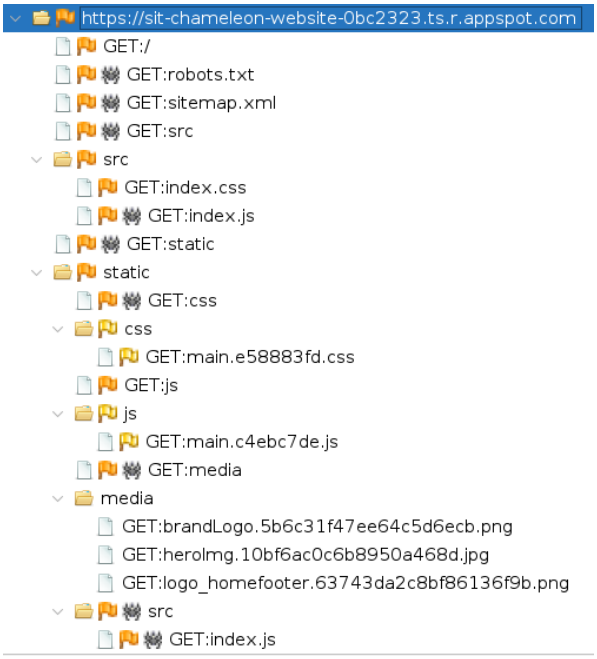


Results

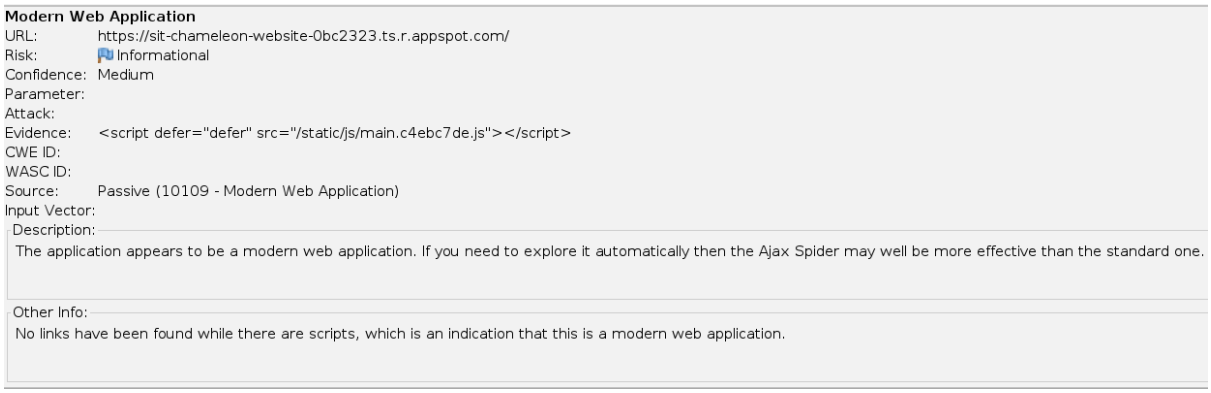
As a result of the Burp-Suite spider scan there seemed to be a lot of recursive directories all leading to the same JavaScript file and considering when navigating the site with Burp-Suite's interceptor on doesn't prompt for any additional packets when navigating to different pages it is apparent the site is not a traditional one but relies on JavaScript to function.

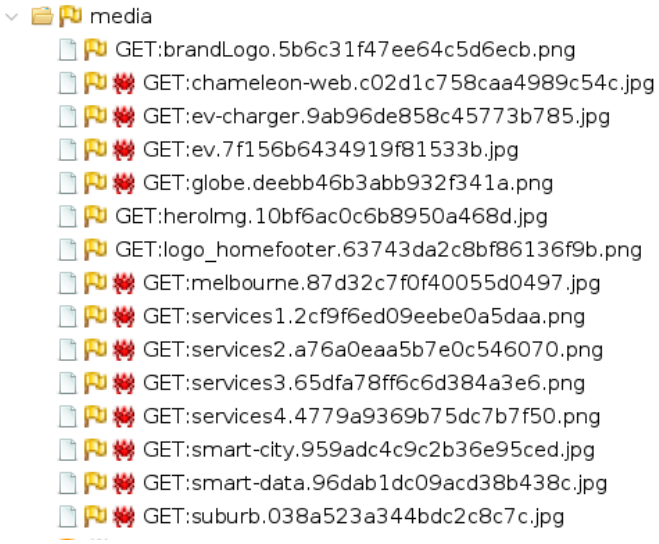


This is further confirmed by ZAP, which unlike Burp-Suite which got stuck in a recursive crawl before touching the JavaScript file, identified the JavaScript as well some additional files and images that were not picked up by Burp-Suite.



ZAP even goes one step further and in its alerts tab and informs us that it has detected the site is a modern web application, providing us evidence that the site calls on a JavaScript script, and referring us to attempt an AJAX spider attack instead with their build-in addon that integrates a crawler for AJAX rich sites known as "Crawljax" which can be combined with a normal spider attack for better results.



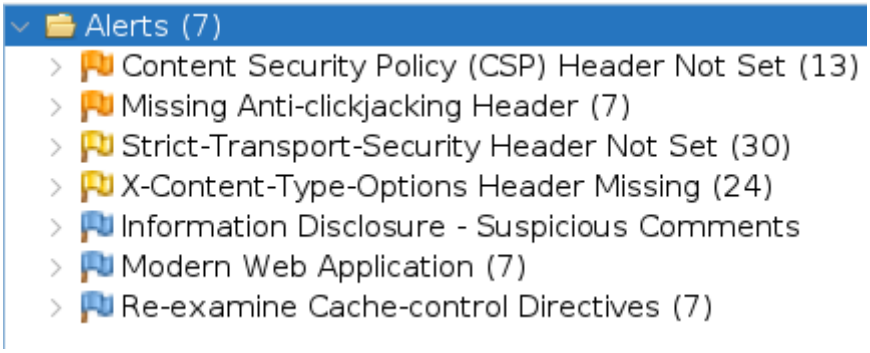


Initiating the AJAX spider attack does reveal some additional files, the images used on the site that were not picked up by the regular spider probably due to them being handled by the JavaScript on the site.

However, the AJAX attack does not reveal any additional files other then the images that Burp-Suite couldn't detect, so this is practically a dead end, but could potentially be exploited by another attack such as a traversal attack.

Lastly, we have the Forced Browse Directory attack which using the directory-list-1.0 included in ZAP attempts to directly access directories and files not picked up by the spider attacks, but unfortunately after being left to run for multiple hours, with over 90,000 different requests being made, nothing additional was found.

Additionally in the process of using ZAP for these spider attacks some extra information in the form of alerts was provided which shall be addressed in the Recommendations section.



Recommendations

In regards to what I suggest, as it pertains to the sites directory and JavaScript there seems to be no holes, or very little that can be taken advantage of, multiple spider attacks revealed nothing, and even a very brute force directory crawl resulted in nothing either, however that is not to say the site is air-tight, as previously mentioned ZAP did detect some issues that need to be addressed to further tighten security for the site:

CSP Header Not Set:
Content Security Policy is an added layer of security, helping to detect and mitigate certain types of attacks, such as Cross Site Scripting (XSS) and data injection attacks, by configuring a CSP header a site owner would be able to declare what types of approved content the page is allowed to load.

Missing Anti-clickjacking Header:
As a result of the site missing CSP as well as not alternatively having X-Frame-Options headers the site is open to potential "Click-Jacking" attacks, which results in a user being tricked into clicking something that does something other than what the user thinks it is.

Strict-Transport-Security Header Not Set:
Due to this header not being set, HSTS (HTTP Strict Transport Security) is disabled, which would disallow users to access the site using a HTTP request, but instead automatically redirect insecure requests to use HTTPS instead, further strengthening website and user security alike.

X-Content-Type-Options Header Missing:
This header is for Anti-MIME-Sniffing, and its options on the Chameleon site were not set to "nosniff" which means that older browsers can perform MIME-Sniffing, which can potentially cause content that is not meant to be displayed to be shown rather than restricting the response to only display content that the site owner intends to display.

Re-examine Cache-control Directives:
According to ZAP the cache-control header has not been set or is not present at all, which allows both browsers and proxies to cache content, which may be intended for resources such as CSS, Js or image files, but this should be looked in further and restricted or else there is a risk that sensitive content could be cached.

Conclusion

In conclusion, no extra resources were locatable via the tests performed, the Spider attack (both regular and AJAX) revealed nothing more than the images used for some of the site's pages which could just as easily be accessed via a regular browser, even the forced directory test revealed nothing, If anything the most critical information in regards to vulnerabilities was revealed by the ZAP alerts and all that I can suggest at this time is the Chameleon web admins follows the advice in the recommendations section to further secure the site and potentially the information discovered by these tests can be applied to another attack vector.

References:

What is a web crawler? | how web spiders work | cloudflare. Available at: https://www.cloudflare.com/learning/bots/what-is-a-web-crawler/ (Accessed: 28 November 2023).

HTTP strict transport security cheat sheet | HTTP Strict Transport Security - OWASP Cheat Sheet Series. Available at: https://cheatsheetseries.owasp.org/cheatsheets/HTTP_Strict_Transport_Security_Cheat_Sheet.html (Accessed: 28 November 2023).

MozDevNet *Content security policy (CSP) - http: MDN*, *MDN Web Docs*. Available at: https://developer.mozilla.org/en-US/docs/Web/Security/CSP/Introducing_Content_Security_Policy (Accessed: 28 November 2023).

MozDevNet *Types of attacks - security on the web: MDN*, *MDN Web Docs*. Available at: https://developer.mozilla.org/en-US/docs/Web/Security/Types_of_attacks#click-jacking (Accessed: 28 November 2023).

MozDevNet *X-Frame-Options - http: MDN*, *MDN Web Docs*. Available at: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options (Accessed: 28 November 2023).

Professional / community 1.7.36 (2018) *Burp Suite Release Notes*. Available at: https://portswigger.net/burp/releases/professional-community-1-7-36 (Accessed: 28 November 2023).

Reducing MIME type security risks (Windows) | Microsoft Learn. Available at: http://msdn.microsoft.com/en-us/library/ie/gg622941%28v=vs.85%29.aspx (Accessed: 28 November 2023).

Session management cheat sheet Session Management - OWASP Cheat Sheet Series. Available at: https://cheatsheetseries.owasp.org/cheatsheets/Session_Management_Cheat_Sheet.html#web-content-caching (Accessed: 29 November 2023).

Spider ZAP. Available at: https://www.zaproxy.org/docs/desktop/addons/spider/ (Accessed: 28 November 2023).

AJAX-Spider ZAP. Available at: https://www.zaproxy.org/docs/desktop/addons/ajax-spider/ (Accessed: 29 November 2023).

Forced-Browse ZAP. Available at: https://www.zaproxy.org/docs/desktop/addons/forced-browse/ (Accessed: 29 November 2023).