

SpiderSuite

User's Guide

v1.0.2

Additional information on SpiderSuite can be found on the official online wiki which is updated regularly: <https://github.com/3nock/SpiderSuite/wiki>

Also see: <https://SpiderSuite.github.io/docs/>

TABLE OF CONTENTS

TABLE OF CONTENTS	1
OVERVIEW	3
Product Overview	3
Features.....	3
Requirements.....	3
INSTALLATION	4
To Download	4
To Install	5
To Uninstall	6
USER INTERFACE	7
Main Window.....	7
Menu & Tool Bar.....	8
Sitemap View	10
Request View	10
Structure View.....	11
Source View	12
Graph View.....	12
CONFIGURATION	13
Limits Configuration	13
Crawler Configuration.....	16
Request Headers Configuration.....	21
Input Fields Configuration	22
Exclusion Configuration.....	23
Authentication Configuration	24
Proxy Configuration	25
Graph Configuration.....	26
Configure Exports.....	27
Passive Crawler Configuration.....	28
Misc Configuration	29
CRAWLING.....	30

Configure the crawler.....	30
Crawling from a Single Link.....	31
Advance Crawling	33
Crawling target with initial seed links	34
Fetching list of links.....	35
Bruteforcing pages / directories	36
Sitemap.....	37
Graph	39
TOOLS	42
Passive Crawler Tool.....	42
SSL Certificates Tool	43
Decoder Tool.....	44
Search Tool.....	45
Compare Tool	46
CONTACTS	48

OVERVIEW

Product Overview

Spider Suite is an Advance Multi-feature web Crawler/Spider for Cyber Security professionals. It contains a powerful crawler which can crawl even the most sophisticated web pages and produce a readable output and an intuitive user interface to interact with the results.

SpiderSuite is a tool that can be easily utilized by web application developers, penetration testers, bug bounty hunters and cyber security researchers to map a target website and inspect each individual page and assets.

SpiderSuite contains a suite of tools aimed at easing the recon phase of web penetration testing and gives a detailed overview of the attack surface of a web application.

Features

- Advance crawler which can crawl an entire target site fast.
- Bruteforce Crawling. Can crawl a target by bruteforcing pages.
- Extract important contents from the crawled pages such as scripts, styles and comments embedded in the webpage.
- Graph visualization of the entire crawled surface or a branch of the crawled surface.
- Import crawled pages from other web security crawlers and tools such as burp suite, Fiddler, Katana and Caido.
- Compare crawled pages and entire crawl projects.
- Export crawled links to different output formats such as CSV, JSON, XML, HTML and Sitemap.xml.

Requirements

SpiderSuite runs on 64 bit machines only, it does not support x32 system and currently available for Windows and Linux operating systems.

INSTALLATION

Installing SpiderSuite involves a series of simple steps which must be completed in the correct sequence for the installation to be successful.

NOTE: *vX.X.X* refers to the version number e.g. **v1.0.1**

To Download

The installers and portable executable files are available on the SpiderSuite's Github repository <https://github.com/3nock/SpiderSuite/releases> page or official website <https://spidersuite.github.io/download/> download page

SpiderSuite currently supports Windows and Linux operating systems with x64 architecture.

On the release page there are two types of download packages for each of the two systems i.e. **installers** and **portable executables**.

Installers:

Installers will install SpiderSuite and its dependencies in default chosen location on your machine.

The installers are ***SpiderSuite_vX.X.X_win64_installer.exe*** for windows and ***SpiderSuite_vX.X.X_linux_installer.run*** for linux.

Portable executables:

Portable executables do not need any installation; you simply download and use directly.

The portable executables are ***SpiderSuite_vX.X.X_win64.zip*** for Windows and ***SpiderSuite_vX.X.X_linux.Applmage*** and ***SpiderSuite_vX.X.X_linux.tar.gz*** for Linux

To Install

Please Note: SpiderSuite is a Graphical User Interface application so all of the steps in this guide refer to the SpiderSuite GUI.

For Windows Portable Executables:

Download and extract **SpiderSuite_vX.X.X_win64.zip** archive and place it at your chosen location, extract the archive then run SpiderSuite.exe simply by double clicking on the program.

For Linux Portable Executable:

Download **SpiderSuite_vX.X.X_linux.Applmage** and place it at your chosen location, then run it simply by **double clicking or** using command line with the command:

```
./SpiderSuite_vX.X.X_linux.Applmage
```

Download **SpiderSuite_vX.X.X_linux.tar.gz** archive and place it at your chosen location.

Extract the archive on desired location (you can use GUI option or command line:

```
tar -xzf SpiderSuite_vX.X.X_linux.tar.gz
```

Then run it simply by double clicking on the **SpiderSuite/SpiderSuite** or using command line with the command

```
./SpiderSuite/SpiderSuite
```

For Windows Installer:

Download **SpiderSuite_vX.X.X_win64_installer.exe** then run the installer, then fill in the required information such as installation location and shortcut names procedurally until you finish the installation procedure.

For Linux Installer:

Download **SpiderSuite_vX.X.X_linux_installer.run** then run the installer by double clicking it or using the command line :

```
./SpiderSuite_vX.X.X_linux_installer.run
```

Then fill in the required information such as installation location and shortcut names procedurally until you finish the installation procedure.

NOTE:

In most windows environment SpiderSuite will run on the first try but case of SpiderSuite fails to run on the first try on windows, this could be an indication that your machine does not contain the required MSVC-redistributable package.

Worry not SpiderSuite comes packaged with the required MSVC-redistributable package in case of this.

*Simply Install the MSVC-redistributable package which comes with SpiderSuite (**SpiderSuite/vcredist_x64.exe**)*

*Also In case SpiderSuite fails to connect to the internet and shows SSL errors, install the OpenSSL package which comes packaged with SpiderSuite (**SpiderSuite/Win64 OpenSSL v1.1.1n Light.msi**).*

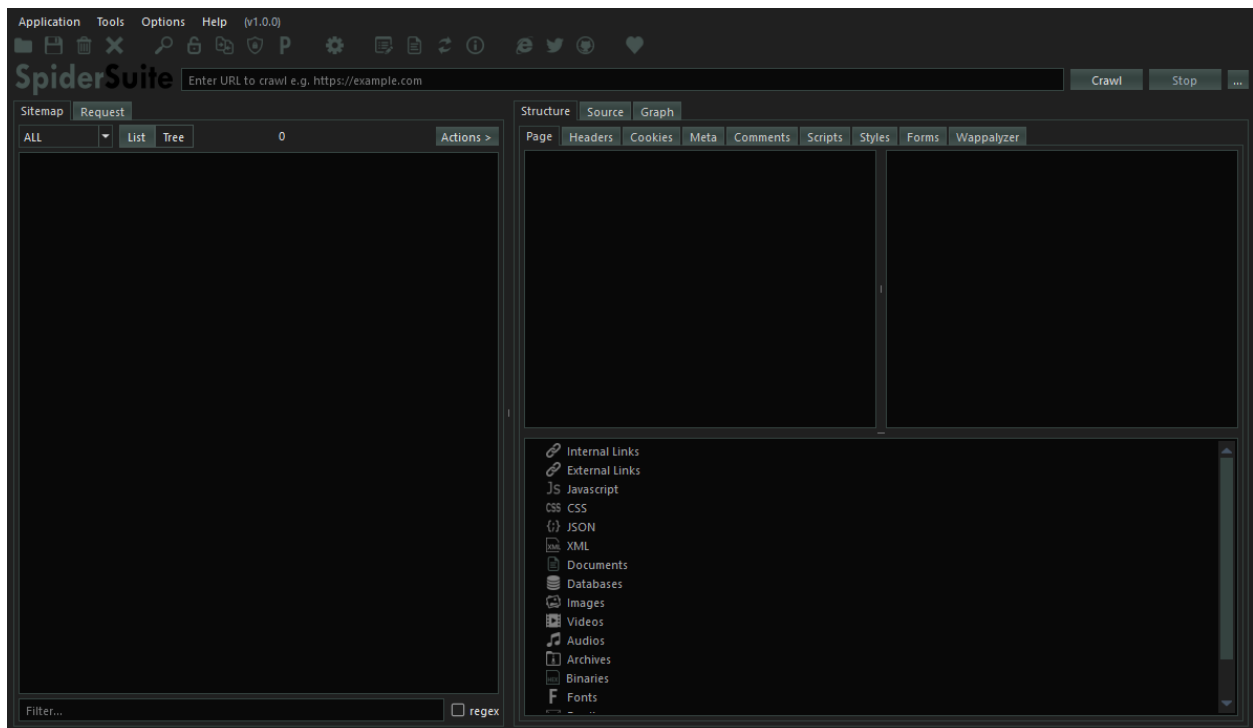
To Uninstall

To uninstall SpiderSuite for portable SpiderSuite, just delete the **SpiderSuite** folder or the **SpiderSuite.Applmage** and you are done. For installed SpiderSuite run the maintenance tool located in the SpiderSuite's installation directory (**SpiderSuite/maintanancetool.exe** in windows and **SpiderSuite/maintanancetool** in linux).

USER INTERFACE

Main Window

SpiderSuite's Main window contains the following information.



- Menu & Tool bar
- Sitemap View
- Request View
- Structure View
- Source View
- Graph View

Menu & Tool Bar

Contains actions that can be performed on the application and the project.

- **Application**

- **Open** - opens a project from the file system.
- **Open Recent** - shows and opens recent SpiderSuite projects you've been working on.
- **Import From** - Imports links and their data from other tools and file types such as:
 - Links from CSV files
 - Links from Sitemap files
 - Links from Zed Attack Proxy(ZAP),
 - Links from acunetix(.xml files),
 - Data from Fiddler(.saz files),
 - Data exported from BurpSuite(.xml files),
 - Data from HTTP Archives(.har files)
 - Data from Caido (CSV & JSON files)
 - Data from Katana crawler (INDEX and JSON files)
- **Save** - Saves the current project you're working on without closing it.
- **Clear** – Clears and deletes all the data of the current loaded project.
- **Close** - closes the current loaded project without deleting any data from that particular project.
- **Exit** - Closes the application.

- **Tools**

- **Search Tool** – Searches the current project's data in the database and returns all the pages that contains that particular search query. The search can take a long or short period depending on the size of the project.
- **Decoder Tool** – Encodes, Decodes or hashes the input data using the chosen encoding, decoding or hashing algorithm. Many other encoding, decoding and hashing algorithms will be added in the coming versions.

- **Compare Tool** - compares two different pages or crawls then highlights the differences and similarities of the two pages or crawls.
- **SSL certificates Tool** - Fetches SSL certificates of a particular target hostname. It does this by trying to establish a secure connection to the hostname and when done it returns the target hostname SSL certificate and closes the connection.
- **Passive Crawler Tool** - Uses OSINT (open source intelligence) sources such as *waybackmachine* to obtain all publicly available url links of a particular target. You can use the obtained links as seed links for the next crawl as it broadens the crawlers scope.
- **Options**
 - **Preferences** - All program's settings and scan configurations.
- **Help**
 - **Log Viewer** - Displays all scan and program logs.
 - **Documentation** - Takes you to the official documentation of SpiderSuite
 - **Donate** - Takes you to a page for donating to the SpiderSuite project.
 - **Website** - Takes you to SpiderSuite official website where you can find all the all information, blog and documentation on SpiderSuite.
 - **Twitter** - Takes you to official Spider Suite's twitter page.
 - **Github** - Takes you to official Spider Suite's Github repository page
 - **Check For Updates** - checks for any available SpiderSuite updates from the repository.
 - **About** - Information about SpiderSuite.
 - **About Qt** - Information about Qt C++ Framework used to create SpiderSuite.

Sitemap View

Displays the crawled pages or pages imported from other tools.

- **Types ComboBox**

Filters the types of pages to display on the sitemap i.e:

- All (displays all page type)
- HTML (displays only html files)
- Javascript (displays only javascript files)
- CSS (displays only CSS files)
- JSON (displays only JSON files)
- XML (displays only XML files)
- Image (displays only image files)
- Audio (displays only audio files)
- Video (displays only video files)
- Documents (displays only document files)
- Database (displays only database files)
- Archive (displays only archive files)
- Binary (displays only binary files)
- Font (displays only font files)
- Misc (displays miscellaneous files)

- **List** - Displays the pages crawled in a list format
- **Tree** - Displays the pages crawled in a tree hierarchical structure
- **[Action >] button** - Provides a menu of actions that you can perform on the displayed sitemap pages.

Request View

Performs HTTP(S) request on the provided target then saves and display the results.

This feature is useful for performing manual tests on a target. The requests are sent one after another and you can only send another request when the previous request has elicited a response.

- **HTTP version ComboBox** - Choose the HTTP version (HTTP/1.X or HTTP/2.X) to use for request.
- **Method type ComboBox** - Choose the Method (GET, POST, PUT, DELETE) to use for HTTP request.
- **Headers List** - Input request headers to use for request.
- **Query data TextEdit** - Input the query data to send with request for POST & PUT methods only.
- **History List** - Displays the pages that you have requested, you can access the pages content by simply clicking on them.

Structure View

Displays the contents extracted from the crawled pages. Click on a page in Sitemap or Request History to access the page's contents in structure view.

- **Page Tab** - Displays basic information extracted from the page such as size, content type, method used and all links extracted from the page.
- **Headers Tab** - Displays all the request and response headers extracted from the http request and response.
- **Cookies Tab** - Displays all the request and response cookies extracted from the http request and response headers.
- **Meta Tab** - Displays all the meta information extracted from the ``<meta>`` tag in html page.
- **Comments Tab** - Displays all the code comments extracted from the page source.
- **Scripts Tab** - Displays all the Javascript code extracted from the ``<script>`` html tag.

- **Styles Tab** - Displays all the CSS code extracted from the ``<style>`` html tag.
- **Forms Tab** - Displays all the html forms extracted from the html ``<form>`` tag.
- **Wappalyzer Tab** - Detects and Displays all the web technologies used on that page.

Source View

Displays the source (response body) of the crawled page. Click on a page in Sitemap or Request History to access the page's source in Source view.

- **Text View** - Displays the page's source in Text format.
- **Hex View** - Displays the page's source in Hexadecimal format.
- **Tree view** - Displays the page's source in a tree format (available for html, xml, JSON and CSS files only for now).
- **Image View** - Displays the image (available for image files only)

Graph View

The Graph View Visualizes the crawled pages on a graph. It can visualize either the entire sitemap of a branch of the pages from the sitemap tree view.

Uses two types graph layouts (Dot and SFDP layout), three types of node shapes (Rectangle, elliptical and point shapes) and node colors of your choosing.

CONFIGURATION

SpiderSuite Configurations are stored in JSON format in a ***SpiderSuite.conf*** configuration file that is located in the installation folder of SpiderSuite.

You can easily modify the configuration values and save for the changes to take place. You can also easily reset the configurations to default values that come with SpiderSuite, any changes saved or reset will also reset the SpiderSuite.conf file.

Limits Configuration

Configurations for Crawler's limitations.

The screenshot shows the 'Limits' configuration window in SpiderSuite. The window has a dark theme and a tabbed interface. The 'Limits' tab is selected, showing the following settings:

- Configure Crawler limitations**
- Parallel Connections**: Input field with value 6.
- ☐ **Request Timeout (milliseconds)**: Input field with value 5000.
- ☐ **Follow Redirects**: Input field with value 3.
- ☐ **Delay Between Consecutive Requests (milliseconds)**: Input field with value 200.
- ☐ **Maximum Pages To Crawl**: Input field with value 10000.
- ☐ **Maximum Page Depth To Crawl**: Input field with value 10.
- ☐ **Maximum Page Size To Crawl (bytes)**: Input field with value 10000000.
- ☐ **Maximum URL Length (characters)**: Input field with value 500.

At the bottom of the window, there are three buttons: 'Reset', 'Save', and 'Cancel'.

Parallel Connections:

Sets the number of parallel connections to connect to the target server with during crawling. According to Spider Suite's design, 6 connections is the most optimal configuration value as Spider Suites employs other techniques to increase the efficiency of the spider such as; use of HTTP pipelining and HTTP2 multi-plexing.

Timeout:

Sets the maximum waiting duration in milliseconds for a request to elicit a response from the target server.

If checked (set to true), when maximum timeout is reached and there is no response from target server the request will be aborted and closed.

If unchecked (set to false), the request will remain active until it gets a response from server or until the 30 seconds threshold is reached.

Follow Redirects:

Sets the following of the redirect URL in case of a 3XX response status and the maximum number of redirects to follow.

If checked (set to true), when a request elicits a 3XX redirect response, the crawler will automatically redirect to the received redirection URL. You can also set the maximum number of redirects to follow.

If unchecked (set to false), when a 3XX response is received the crawler will not redirect the request, it will simply save the result and continue to crawl other links

Delay between Consecutive Requests:

Sets the wait time between sending consecutive requests. The delay is configured in milliseconds.

If checked (set to true), when spider sends a request to the target server it will wait for XXX milliseconds before sending another request to that target server.

If unchecked (set to false), the spider will send multiple requests in a tight loop to the target server.

This should only be checked (set to true) in cases of targets that

- Can't handle too many requests
- Detects and prevents high speed crawling
- Has strict crawling rules

Otherwise it slows down the crawling process, as the spider has to delay for XXX milliseconds before sending another request.

Maximum Pages to Crawl:

Sets the maximum number of successful result pages to be crawled. This only takes into account the successfully crawled pages from the target server.

If checked (set to true), when the spider successfully crawls the XXX number of pages from the target server the spider will automatically stop regardless whether there are still crawlable links available

If not checked (set to false), the spider will crawl the target site until either you stop the crawling manually or until the spider finishes crawling all the links.

Maximum Crawl Depth:

Sets the maximum page depth to crawl for a particular target. Depth refers to the position of the page from the host domain.

e.g. https://www.example_domain.com/depth_1/depth_2/depth_3?name=value

If checked (set to true), the spider will not crawl a page whose depth exceeds the maximum crawl depth. Only pages whose depth are between 0 and max crawl depth will be crawled.

If not checked (set to false), the spider will crawl all pages of all depths of the particular target.

Maximum Page Size to Crawl:

Sets the maximum page size in bytes for download.

If checked (set to true), pages whose size is above the maximum page download size will not be downloaded.

If not checked (set to false), pages of all sizes will be downloaded and crawled.

Maximum URL Length:

Sets the maximum allowed length in bytes/characters for a URL link.

If checked (set to true), links whose length is above the maximum link length value will not be crawled.

If not checked (set to false), links of all lengths will be crawled.

Crawler Configuration

Crawler specific configurations.

The screenshot shows the 'Crawler Configuration' window in Spider Suite. The 'Crawler' tab is active. The 'config' section includes the following settings:

- ☒ Use HTTP2
- ☒ Use HTTP pipelining
- ☒ Accept Cookies
- ☐ Ignore query parameters in html links
- ☒ Ignore query parameters in resource links (js,css,xml,json,img,...)
- ☒ Crawl Sitemap file
- ☐ Follow "robots.txt" Exclusions
- ☒ Crawl Linked files only
- ☒ Crawl resource links (js,css,img) from external domains
- ☐ Strip trailing slash on links
- ☐ Extract links from <style>
- ☐ Extract links from <script>
- ☐ Extract links from comments
- ☐ Add failed requests (4XX and 5XX) to Sitemap

The 'FileTypes To Crawl' section shows the following file types and their selection status:

File Type	Selected
Javascript	Yes
CSS	Yes
XML	Yes
JSON	Yes
Fonts	No
Images	No
Audio	No
Video	No
Archives	No
Documents	No
Databases	No
Binaries	No

At the bottom of the window are three buttons: 'Reset', 'Save', and 'Cancel'.

Use HTTP2:

Configures the use of HTTP version 2 protocol for request to the target server. If a server turns out to not support HTTP/2 the spider will fall back to using HTTP version 1 by default.

HTTP/2 is the second version of the HTTP protocol aiming to make applications faster, simpler, and more robust by improving many of the drawbacks of the first HTTP version. Has features including;

- **Binary protocols** – Binary protocols consume less bandwidth, are more efficiently parsed and are less error-prone than the textual protocols used by HTTP/1.1.
- **Multiplexing** – HTTP/2 is multiplexed, i.e., it can initiate multiple requests in parallel over a single TCP connection. As a result, web pages containing several elements are delivered over one TCP connection.
- **Header compression** – HTTP/2 uses header compression to reduce the overhead caused by TCP's slow-start mechanism.
- **Server push** – HTTP/2 servers push likely-to-be-used resources into a browser's cache, even before they're requested. This allows browsers to display content without additional request cycles.
- **Increased security** – Web browsers only support HTTP/2 via encrypted connections, increasing user and application security.

HTTP Pipelining:

Configures the spider to use http-pipelining technique to increase speed and efficiency. Works when using HTTP/1.

HTTP pipelining is a feature of HTTP/1.1 which allows multiple HTTP requests to be sent over a single TCP connection without waiting for the corresponding responses.

HTTP/1.1 requires servers to respond to pipelined requests correctly, with non-pipelined but valid responses even if server does not support HTTP pipelining. Despite this requirement, many legacy HTTP/1.1 servers do not support pipelining correctly, forcing most HTTP clients to not use HTTP pipelining.

The technique was superseded by multiplexing via HTTP/2.

Accept Cookies:

Configures the spider to accept and use cookies sent by the target server. The spider will store the cookie and send it back to the same server with later requests.

An HTTP cookie (web cookie, browser cookie) is a small piece of data that a server sends to a user's web browser (in this case SpiderSuite). Cookies are mainly used for three purposes: Session management, Personalization & Tracking.

Ignore query parameters in html links:

Configures the crawler to ignore all query parameters in all html links it crawls.

e.g. *https://example.com?param1=1¶m2=2 => https://example.com*

Ignore query parameters in Resource links:

Configures the crawler to ignore all query parameters in resource links only. Resource links here refers to links which are not of html file type e.g. js, css, xml, json links.

Crawl sitemap File:

Configures the crawler to first fetch sitemap.xml file if the site contains one. If the site does possess the sitemap.xml file then the spider will use the links from the sitemap.xml file as seed links for crawling the target site.

An XML sitemap is a file that lists a website's essential pages, making sure Google can find and crawl them all. It also helps search engines understand your website structure. You want Google to crawl every important page of your website.

Follow “robots.txt” Exclusions:

Configures the crawler to follow the “**robots.txt**” exclusions if available. The robots.txt exclusion is highly dependent on the user-agent used hence the crawler will follow the exclusions according to the user-agents you've chosen.

The robots exclusion standard, also known as the robots exclusion protocol or simply robots.txt, is a standard used by websites to communicate with web crawlers and other web robots.

Crawl linked files only:

Configures the crawler to crawl only the links that it extracts from the crawled pages meaning it will not crawl other directories in the link structure.

E.g. For a linked link <https://example.com/dir1/dir2/dir3>, the crawler will only fetch for page <https://example.com/dir1/dir2/dir3> and won't try to crawl other directory pages found in this link such as <https://example.com/dir1> and <https://example.com/dir/dir2>.

Crawl Resource Links from external domains:

Configures the spider to crawl resources (images, css, js, json, xml & fonts) hosted from a different host. Most webpages uses resources hosted from different hosts. If this feature is selected external resources will be crawled if this feature not selected only resources from the target host will be crawled.

Strip trailing slash on links:

Configures the filter to strip/remove all trailing slashes in URL links it crawls

e.g. <https://example.com/> ==> <https://example.com>

This helps preventing duplicate pages as most of the time the link with and without trailing slash are the same page.

Use with Caution: because the two pages might be different

Extract links from script tag

Configures the crawler to extract links from Javascript code inside the **<script>** tag.

If the pages contains many, long and repetitive Javascript scripts it slow might slow down the extraction process hence this configuration disables extraction of links from these scripts.

The disadvantage to this is that the scripts may contain useful links that could expand the target scope.

Extract links from style tag

Configures the crawler to extract links from CSS code inside the **<style>** tag.

If the pages contains many, long and repetitive CSS code it slow might slow down the extraction process hence this configuration disables extraction of links from these styles.

The disadvantage to this is that the styles may contain useful links that could expand the target scope.

Extract links from comments:

Configures the crawler to extract links from code comments.

If the pages contains many, long and repetitive code comments it might slow down the extraction process hence this configuration disables extraction of links from these comments.

The disadvantage to this is that the comments may contain useful links that could expand the target scope.

Add failed requests (4XX and 5XX) to sitemap:

Configures the crawler not to discard 4XX and 5XX error responses and instead, add them to database and send them to sitemap.

Connection, SSL and other system errors will not be added to sitemap only error responses with 4XX and 5XX.

Allow POST and PUT requests:

Configures the crawler to allow sending POST and PUT requests when crawling. Since POST and PUT request may have an effect on the target it is not allowed by default.

Filter out recurring parameters:

Configures the crawler to remove repeating parameters even those with different values. This configuration prevents crawling of the same pages with just different parameter values.

Although it might be helpful most of the time, sometimes the parameter values might be essential.

File types to Crawl:

Configures which files types are allowed to be crawled. HTML files are crawled by default, but for other file types you must allow (by checking the particular checkbox) for them to be crawled. If not allowed the file type will not be added to the spider seed.

Request Headers Configuration

Crawler's request headers configuration.

Limits Crawler **Headers** Input Fields Exclusion Authentication Proxy Graph Export Passive Crawler Misc

HTTP Headers and User-Agents to be sent with each Crawler/Spider request

Request Headers

Header	Value	Action
<input type="checkbox"/> Referer		Remove
<input type="checkbox"/> Accept	/*/*	
<input type="checkbox"/> Accept-Charset	utf-8	
<input type="checkbox"/> Accept-Encoding	br	
<input type="checkbox"/> Accept-Encoding	gzip	
<input type="checkbox"/> Accept-Language	*	
<input type="checkbox"/> Cache-Control	no-cache	
<input type="checkbox"/> Cache-Control	no-cache, no-store	
<input type="checkbox"/> Connection	closed	
<input type="checkbox"/> Connection	keep-alive	
<input type="checkbox"/> Pragma	no-cache	

Enter header name... Enter header value... Add

☐ **User Agent Header**

User Agent	Action
<input type="checkbox"/> Alcatel-OT-708/1.0 Profile/MIDP-2.0 Configuration/CLDC-1.1 ObigoInternetBrowser/Q03C	Remove
<input type="checkbox"/> BlackBerry7100i/4.1.0 Profile/MIDP-2.0 Configuration/CLDC-1.1 VendorID/103	
<input type="checkbox"/> BlackBerry7130e/4.1.0 Profile/MIDP-2.0 Configuration/CLDC-1.1 VendorID/104	
<input type="checkbox"/> BlackBerry7230/3.7.0	
<input type="checkbox"/> BlackBerry7250/4.0.0 Profile/MIDP-2.0 Configuration/CLDC-1.1	
<input type="checkbox"/> BlackBerry7520/4.0.0 Profile/MIDP-2.0 Configuration/CLDC-1.1	
<input type="checkbox"/> BlackBerry7730/3.7.0	
<input type="checkbox"/> BlackBerry8130/4.3.0 Profile/MIDP-2.0 Configuration/CLDC-1.1 VendorID/109	
<input type="checkbox"/> BlackBerry8310/4.2.2 Profile/MIDP-2.0 Configuration/CLDC-1.1 VendorID/121	
<input type="checkbox"/> BlackBerry8320/4.3.1 Profile/MIDP-2.0 Configuration/CLDC-1.1	
<input type="checkbox"/> BlackBerry8700/4.1.0 Profile/MIDP-2.0 Configuration/CLDC-1.1 VendorID/100	

Enter user_agent... Add

Reset Save Cancel

Request Headers:

Choose (check the header's check box of) the headers you want to use for crawling the target site. You can add your own custom headers by introducing the header name and value then click add button to add the new header to the headers list.

When checked (in use) the Referrer header value will be added automatically by the spider during the scan depending on where the link was extracted from.

User Agent Header:

The User-Agent request header is a characteristic string that lets servers and network peers identify the application, operating system, vendor, and/or version of the requesting user agent.

Check the (User Agent Header) checkbox to allow the use of user agents in crawling. Then choose the user agent(s) to use. If multiple user agents are chosen, they will all be used randomly for each request sent.

Input Fields Configuration

Configures the default values for the listed common input name and types for HTML forms.

Configure List of input field's values to be set automatically when submitting html forms during crawling

Parse Input Fields From URL

Automatically extracts all input fields names from the provided URL and manually add values

Target URL... Fetch

Accept-Language	en	Remove
Client-IP	127.0.0.1	
DOB	1967/1/1	
Origin	http://www.test.com/	
Referer	http://www.google.com/search?hl=en&q=testing	
Via	1.1 wa.www.test.com	
X-Forwarded-For	127.0.0.1	
X-Forwarded-Host	localhost	
accounts	1001160141	
addr	3137 Laguna Street	
address	3137 Laguna Street	
age	20	
alter	20	
anno	1967	
area	555	
awayyear	9999	
benutzer	aaabbbccc	
beruf	programmer	
birth	01/01/1967	
birth_year	1967	
birthday	01/01/1967	
cardnum	4111111111111111	
cc	4111111111111111	

input name... input value... Add

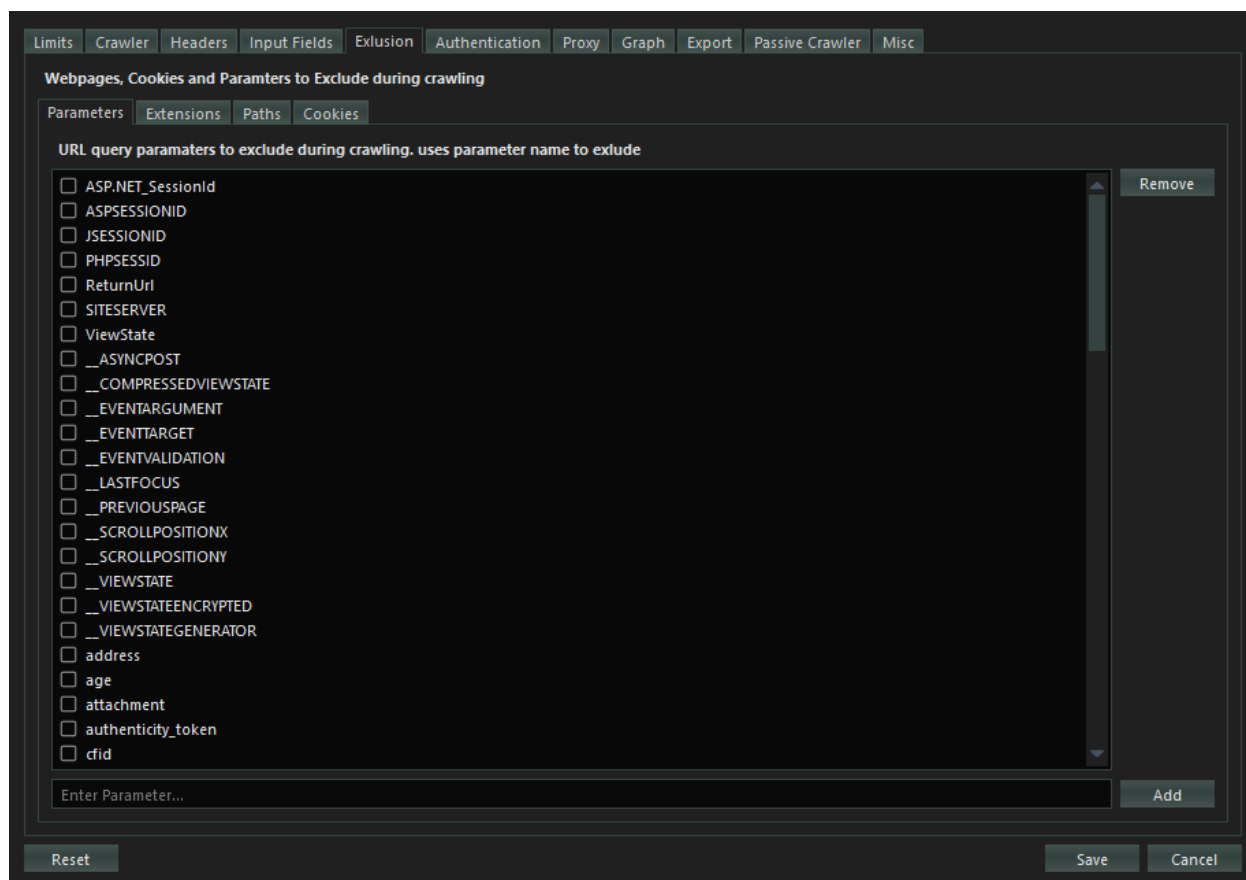
Reset Save Cancel

When crawling the HTML forms, values from all empty input values will be filled by the values from this list.

You can also add your own custom input name/type and value to be used in HTML form's inputs. You can also parse and extract input fields from a URL automatically by fetching it then adding values to the input fields manually. ***This is more efficient and precise.***

Exclusion Configuration

Configures what link's query parameters, paths, extensions and cookies should be excluded from crawling.



Exclude Parameters:

Configures which query parameters should be excluded from a network request if present. If exclude all parameters is chosen then query parameters from all links will be removed.

Exclude File Extensions:

Configures the Crawler to exclude crawling all links with file types of the specified extensions (pdf, exe, docx). You can add a file type extension to be excluded

Exclude Paths:

Configures the Crawler to exclude crawling the paths which match to the exclusion patterns. You can add custom excluded paths patterns (regular expression patterns) for each scan.

Exclude Cookies:

Configures the spider to exclude the chosen cookie patterns from the cookie jar. You can add custom cookie patterns (regular expression patterns) to exclude from the scan.

Authentication Configuration

Configures the credentials for authenticated crawl.

Authentication Configuration window showing the 'Standard Based' tab. The window displays a table for adding authentication credentials (Username, Password, Host, Path) and includes an 'Add' button at the bottom right. The 'Form Based' tab is also visible but not selected.

For current version 1.0.0 of SpiderSuite, it only supports standard based authentications which include `Basic`, `NTLM version 2`, `Digest-MD5` and `SPNEGO/Negotiate`. Simply add the credential values; username, password, host site and the path that requires the authentication. After that you can check the credentials for it to be usable when the crawler detects authentication is required.

Form based authentication will be introduced in the coming versions of SpiderSuite.

Proxy Configuration

Configures http and socks5 proxy connection for the crawler.

The screenshot shows the 'Proxy' tab in the Spider Suite configuration window. The window has a dark theme and a tabbed interface at the top with the following tabs: Limits, Crawler, Headers, Input Fields, Exclusion, Authentication, Proxy (selected), Graph, Export, Passive Crawler, and Misc. The main content area is titled 'HTTP Proxy Configurations. All crawling requests and responses with pass through the provided proxy'. It contains a checkbox labeled 'Use Proxy' which is currently unchecked. Below this is a text label: 'Proxy Server configurations, when allowed all requests will be sent and received through the proxy server.' There are two radio buttons: 'HTTP Proxy' (selected) and 'SOCK5 Proxy'. Below the radio buttons are four input fields: 'Host' with the value '127.0.0.1', 'Port' with the value '8080', 'Username' (empty), and 'Password' (empty). At the bottom of the window, there are three buttons: 'Reset', 'Save', and 'Cancel'.

In case of anonymity or integration with other tools such as Burp Suite & Zed Attack Proxy (ZAP) and bypassing various IP related crawling drawbacks.

All crawler request and responses will pass through the configured proxy.

Graph Configuration

Configures Graph visualizing the Crawled sitemap.

Limits Crawler Headers Input Fields Exclusion Authentication Proxy **Graph** Export Passive Crawler Misc

Configures graph characteristics/Attributes

Graph Attributes

Graph Type: SFDP

Graph Direction: Left To Right

Node Shape: Point

Color By: Status Code

Colors

HTTP Status	Color
Informational	yellow
Successful	green
Redirection	orange
Error	red
Notknown	gray

Reset Save Cancel

You can modify the graph appearance on your own liking for better data presentation.

Configure Exports

Configures what data/information about a page should be exported on various export types.

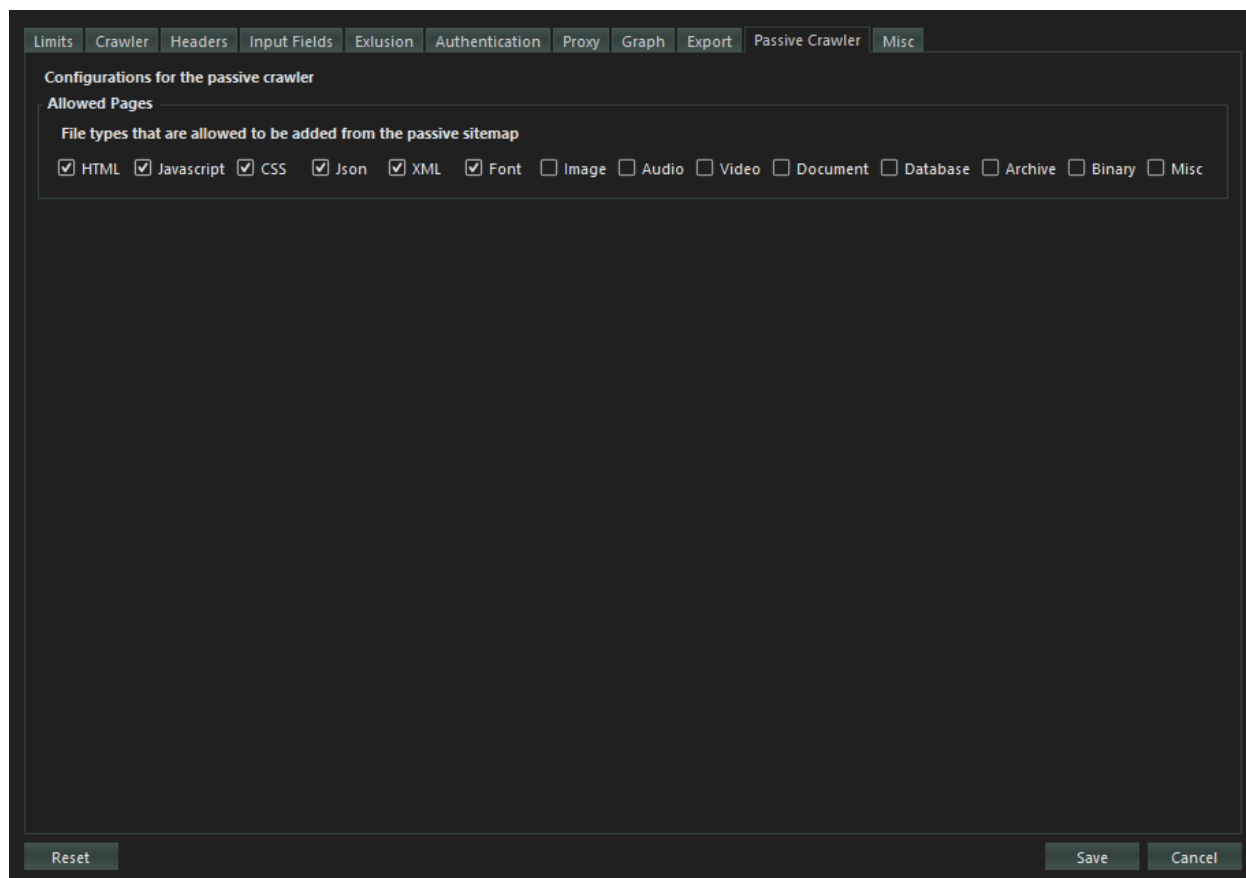
The screenshot shows the 'Configure Exports' window in Spider Suite. The 'Export' tab is active. The window is titled 'Configures what extra information to export for each export type'. It contains four sections for different export formats: CSV, XML, JSON, and HTML. Each section has four checkboxes: 'Status Code', 'Size', 'Title', and 'Content Type', all of which are checked. Below these is a 'Sitemap' section with three checkboxes: 'lastmod', 'changefreq', and 'priority'. Each checkbox has a corresponding text input field with a placeholder example (e.g., 'e.g 2004-10-01T18:23:17+00:00' for lastmod, 'e.g monthly' for changefreq, and 'e.g 0.7' for priority). At the bottom of the window are three buttons: 'Reset', 'Save', and 'Cancel'.

The page URLs will all be exported by default but other data about the webpage such as Status Code, Size, title and Content Type will only be exported when you choose them to be exported.

This feature gives control to the user on what data to be included in the export file.

Passive Crawler Configuration

Configures the SpiderSuite passive crawler.



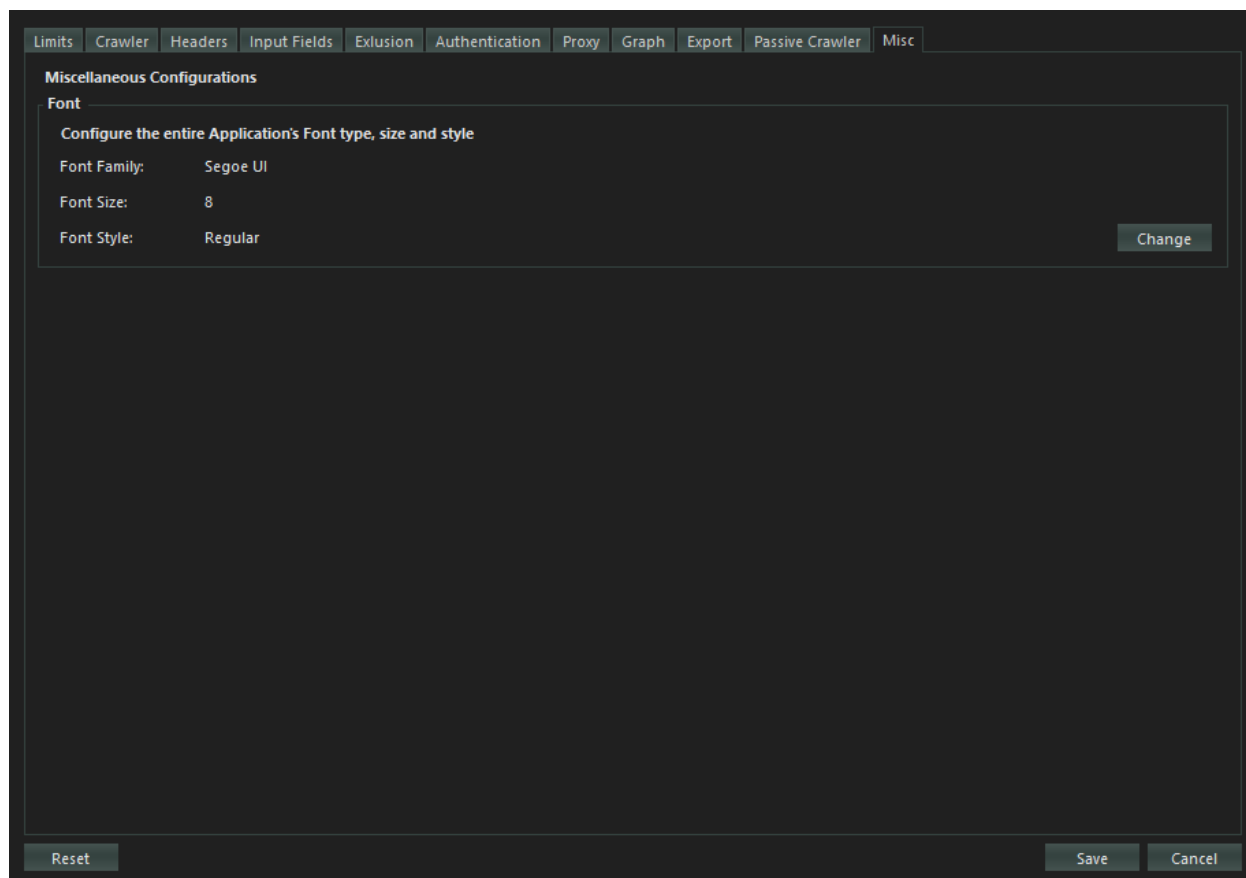
The screenshot shows the 'Passive Crawler' configuration window in Spider Suite. The window has a dark theme and a tabbed interface at the top. The 'Passive Crawler' tab is selected. Below the tabs, the title 'Configurations for the passive crawler' is displayed. Underneath, the section 'Allowed Pages' is visible, followed by the instruction 'File types that are allowed to be added from the passive sitemap'. A list of file types with checkboxes follows: HTML (checked), Javascript (checked), CSS (checked), Json (checked), XML (checked), Font (checked), Image (unchecked), Audio (unchecked), Video (unchecked), Document (unchecked), Database (unchecked), Archive (unchecked), Binary (unchecked), and Misc (unchecked). At the bottom of the window, there are three buttons: 'Reset', 'Save', and 'Cancel'.

Allowed Pages:

Configures which pages (page type e.g. html, js, css) are allowed to be added to the sitemap after the crawling is finished or when you add links.

Misc Configuration

Miscellaneous configurations.



Font:

Configures the Spider Suite's entire application font type, size and style, simply change the application's font by using the font dialog.

The changes will take effect after restarting SpiderSuite.


CRAWLING

The crawl phase of a scan involves navigating around the target web application, following links and submitting forms to catalog the content of the entire target web application and the navigational paths within it to create an accurate map of the target web application.

Configure the crawler

Next step is to set the configurations for the crawler to run on.

This is a very crucial step as the performance and success of the crawler depends on the configurations you have set. Take time to study the effects of all the Configurations

Click the  **config action** on the toolbar to get access to the configuration dialog where you can set preferred configurations.

The configurations that affect the crawler are:

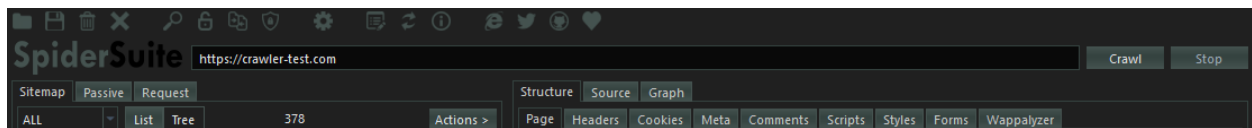
- **Limits**- Sets all the limitations for the crawler
- **Crawler**- Sets all the essential crawler configurations
- **Headers** - Sets the http request headers to be used by the crawler request.
- **Input Fields** - Sets the values for the page's input fields for automatic filling and submission.
- **Exclusion** - Sets the paths, cookies, file extensions and url parameters to be excluded for the crawl.
- **Authentication** - Sets values for automatic authentication by the crawler.
- **Proxy** - Sets the proxy address and port where all crawl request will pass through.

Crawling from a Single Link

Spider Suite can crawl an entire target web application from a single link which acts as the root entry page for target web application.

To start crawling requires you to follow the following simple and few procedures.

1. **Input the target link.** Add the target URL link.



Few things to Note:

Always make sure that you have input a valid URL with its protocol /schema.

- Valid Links are:

https://example.com

https://example.com/

https://example.com/path1/path2

https://example.com:443/path1

https://127.0.0.1:80

- Invalid Links are:

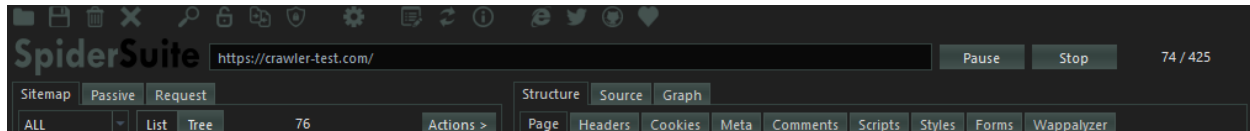
example.com

https://example/

https://example.com?param1=value1¶m2=value2

2. Start Crawler

Start the crawler by clicking on the `Crawl` button and the crawler will immediately start crawling the target web application.



After starting the crawler, you can observe the crawler's `progress` on the far right corner which shows the number of pages crawled per all the pages available:

progress = <pages_crawled> / <total_pages>.

After crawler has started you have options to **Pause** or **Stop** the crawler.

Pause Crawler:

Spider Suite allows you to pause the crawler at any point during crawling and for any duration of time by pressing the **Pause** button.

After pressing **Pause** button the crawler immediately pauses sending the requests to the server or processing new replies `but` all the already finished and processed pages will still be added to the sitemap, so it's not uncommon to pause the crawler and still seeing a few pages being added to the sitemap.

Resume Crawler:

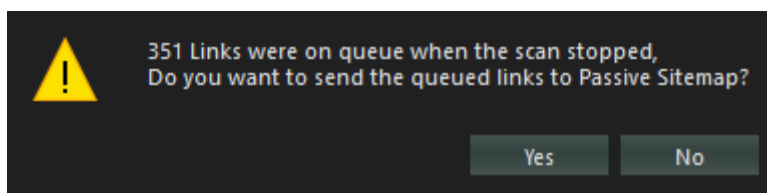
After pausing the crawler you can resume crawling by pressing the `Resume` button and the crawler will immediately resume crawling the target web pages where it left off.

Stop Crawler:

You can stop the crawler at any point in time by pressing the `Stop` button. Stopping the crawler means that you terminate the crawler and you can no longer resume that particular crawl, all resources allocated are cleaned hence you can only start afresh from there.

After pressing **Stop** button the crawler immediately stops sending the requests to the server `but` it will wait until all the already sent request to be processed and added to the sitemap before it kills all the crawler threads. So it's not uncommon to stop the crawler and still seeing a few pages being added to the sitemap as it will wait for all responses from the target server to be processed.

After stopping the crawler you may be prompted to save all the remaining target links that had'nt been crawled yet.



If you accept all the pending links will be added to the passive crawler tool.

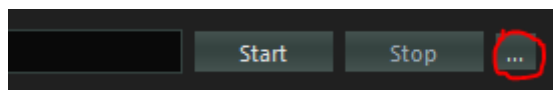
If you deny all the pending links will be discarded.

Advance Crawling

SpiderSuite has advance crawling options:

- Crawling target with initial seed links
- Fetching list of links
- Bruteforcing pages/directories

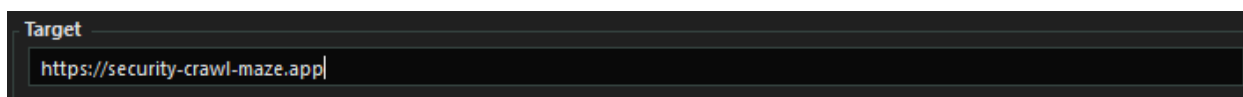
This advance crawling can be accessed by clicking on the [...] button.



Crawling target with initial seed links

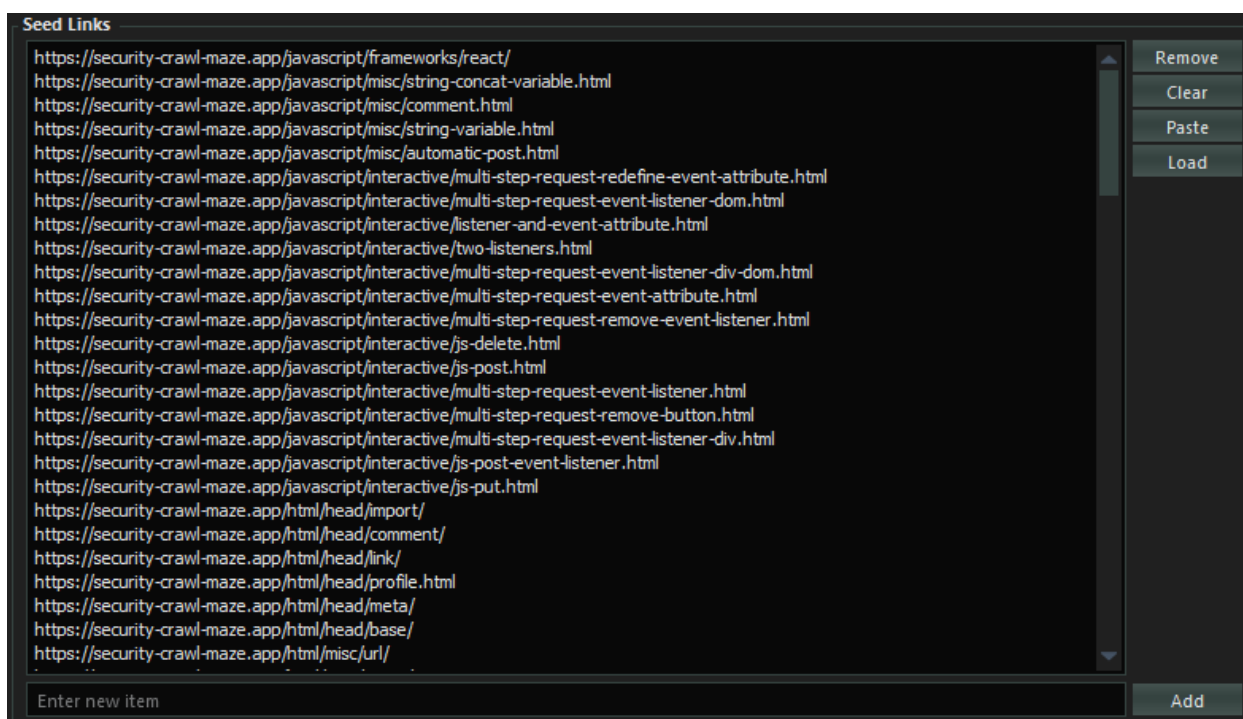
With SpiderSuite you have the ability to provide an initial list of links (‘seed links’) of the particular target and they will be added to the crawler queue of links to be crawled.

- Enter Target link



A screenshot of the 'Target' input field in the Spider Suite interface. The field is a dark grey rectangle with a lighter grey border. Inside the field, the text 'https://security-crawl-maze.app/' is entered in a white monospaced font. Above the field, the word 'Target' is written in a small, light grey font.

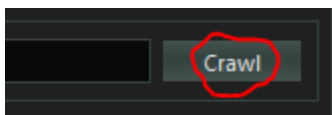
- Add Seed links



A screenshot of the 'Seed Links' section in the Spider Suite interface. It features a list of 25 URLs, all starting with 'https://security-crawl-maze.app/'. The URLs are categorized into JavaScript frameworks (react, string-concat-variable, comment, string-variable, automatic-post), interactive events (multi-step-request-redefine-event-attribute, multi-step-request-event-listener-dom, listener-and-event-attribute, two-listeners, multi-step-request-event-listener-div-dom, multi-step-request-event-attribute, multi-step-request-remove-event-listener, js-delete, js-post, multi-step-request-event-listener, multi-step-request-remove-button, multi-step-request-event-listener-div, js-post-event-listener, js-put), HTML head elements (import, comment, link, profile, meta, base), and a misc URL. To the right of the list are four buttons: 'Remove', 'Clear', 'Paste', and 'Load'. At the bottom of the list is an input field labeled 'Enter new item' and an 'Add' button.

Note: The provided list of links should all relate (have the same hostname) to the main target link as the crawler uses the main target link as the reference point for all the links to be crawled.

- Click on **Crawl** to begin crawling

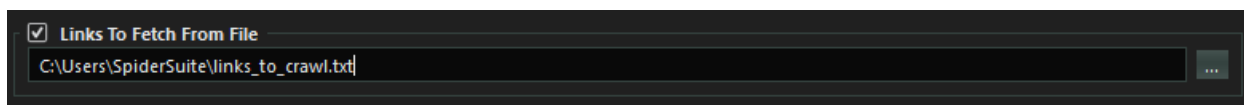


Fetching list of links

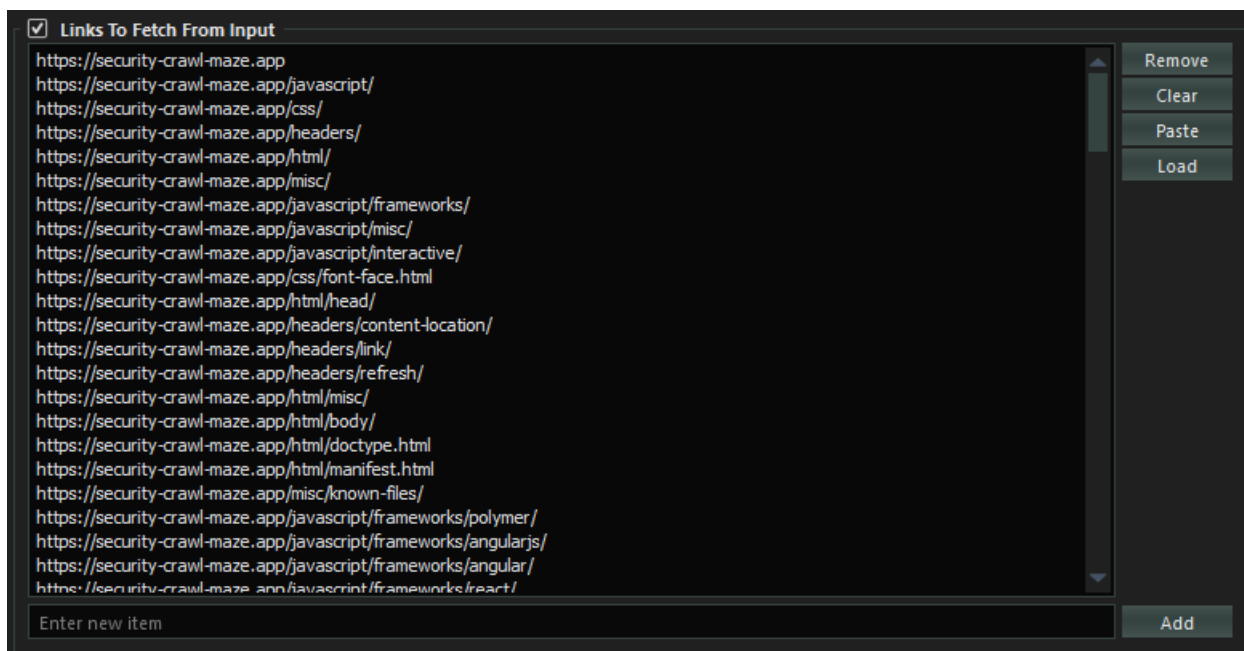
With SpiderSuite you also have the ability to fetch a provided list of links.

This type of crawling (fetching) does not crawl any additional links extracted from the crawled pages. Only the provided links will be crawled.

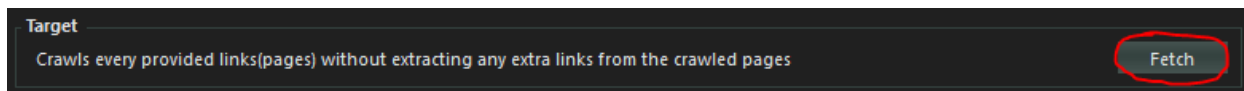
- You can provide file containing the list of links to be fetched. This is ideal for fetching a very long list of links as the file is not loaded into memory (it uses a streaming api to get line after line of link from the file and fetches it)



- Or you can input the list of links to be fetched. This is ideal for fetching a small to medium list of links as the list is stored in memory.



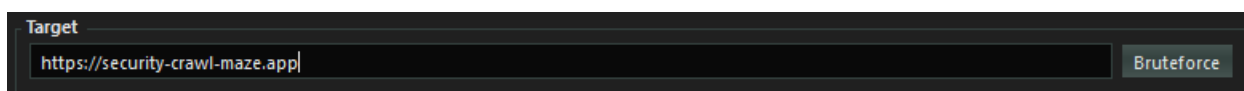
- Start the crawling by clicking on **Fetch** button.



Bruteforcing pages / directories

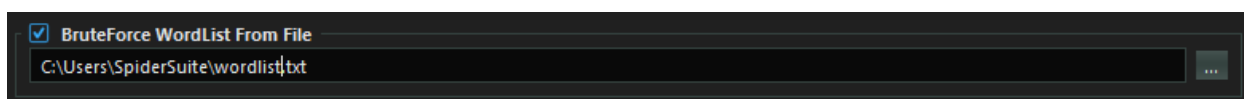
Lastly SpiderSuite also has the ability to bruteforce target sites's directories(pages). This is a useful feature for scoping directories and files that may be hidden.

- Set the target link (URL).



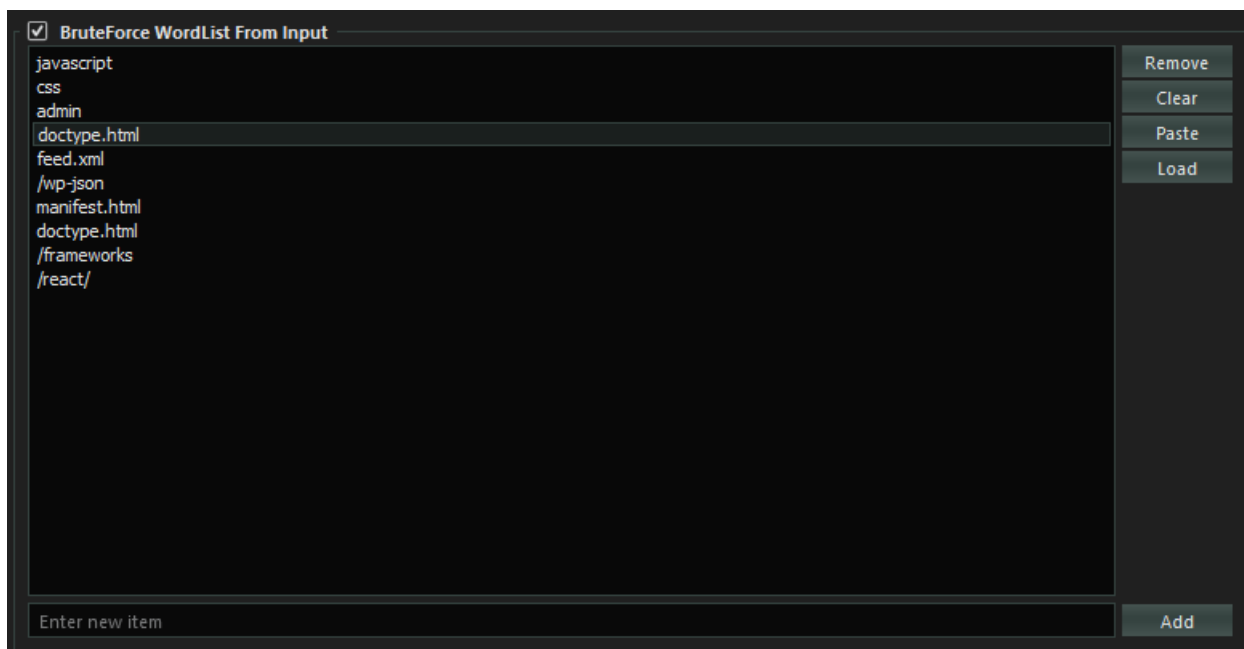
A screenshot of the 'Target' input field in Spider Suite. The field contains the URL 'https://security-crawl-maze.app|'. To the right of the field is a button labeled 'Bruteforce'.

- You can provide file containing the wordlist pages/directories to be used for bruteforcing. This is ideal for bruteforcing with a very long wordlist as the file is not loaded into memory(it uses a streaming api to get line after line of page name from the file, append it to the target link and fetch it).



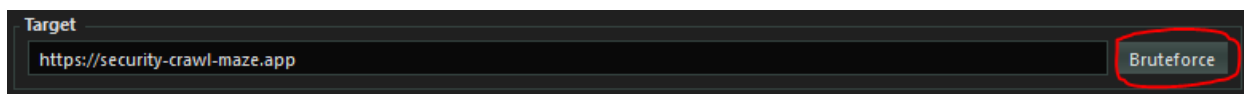
A screenshot of the 'BruteForce WordList From File' option. The checkbox is checked. Below it, a text field contains the file path 'C:\Users\SpiderSuite\wordlist.txt'. To the right of the field is a button with three dots '...'.

- Or you can input the wordlist for bruteforce. This is ideal for small to medium wordlist.



A screenshot of the 'BruteForce WordList From Input' option. The checkbox is checked. Below it, a list box contains the following items: 'javascript', 'css', 'admin', 'doctype.html', 'feed.xml', '/wp-json', 'manifest.html', 'doctype.html', '/frameworks', and '/react/'. To the right of the list box are buttons labeled 'Remove', 'Clear', 'Paste', and 'Load'. At the bottom of the list box is an input field labeled 'Enter new item' and an 'Add' button.

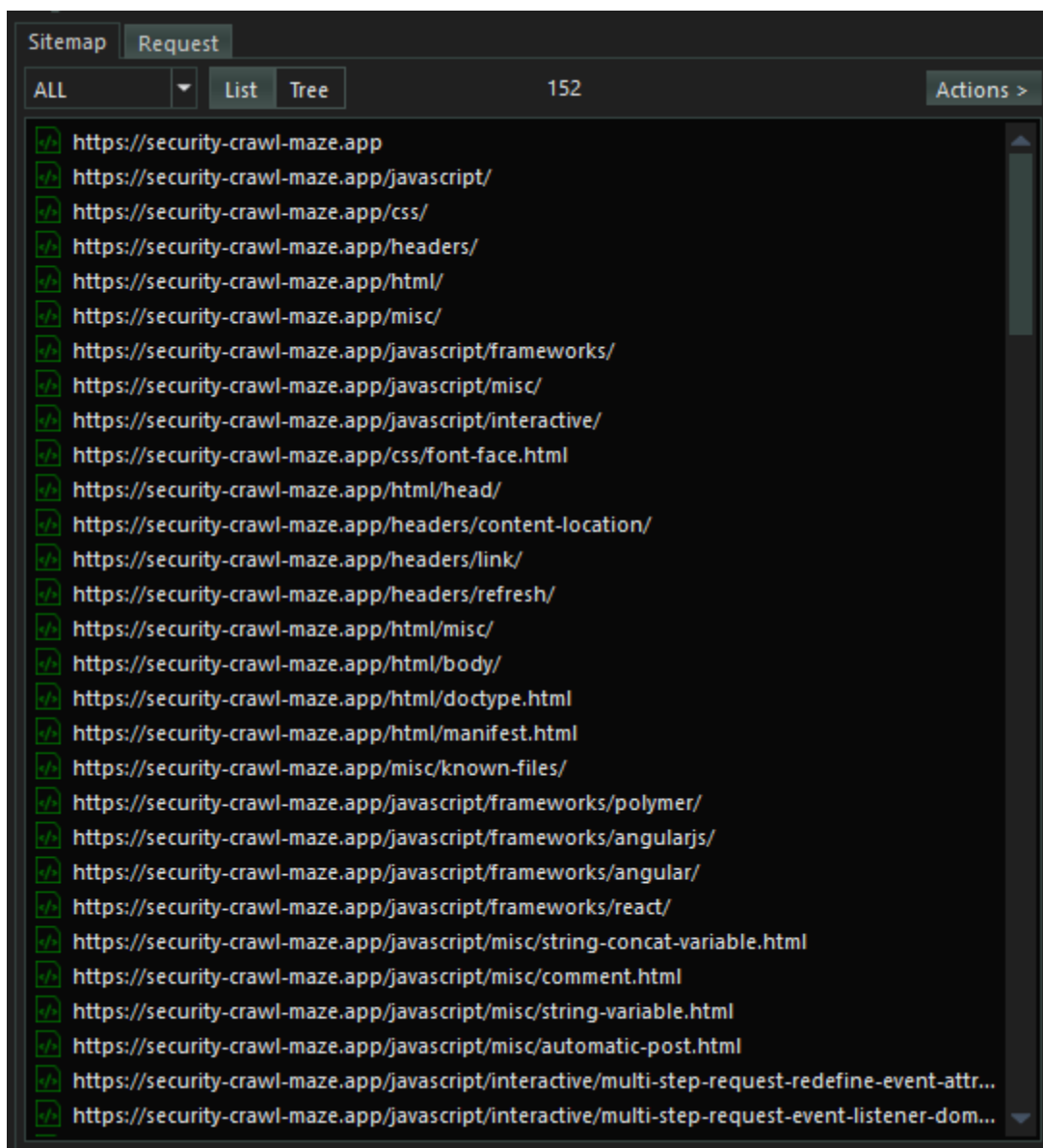
- Start bruteforcing by clicking on **Bruteforce** button.



A screenshot of the 'Target' input field in Spider Suite. The field contains the URL 'https://security-crawl-maze.app'. To the right of the field is a button labeled 'Bruteforce', which is highlighted with a red circle.

Sitemap

All the results from **Crawling**, **Fetching** and **Bruteforcing** will be displayed on Spider Suite's Sitemap.



The actual pages are already saved on Spider Suite's current project database (.sspd) file.

To view content of any page on the sitemap simply click on it and all its content will be displayed on the structure and source tab.

The screenshot displays the Spider Suite application interface. At the top, there are three tabs: 'Structure', 'Source', and 'Graph'. Below these, a row of sub-tabs includes 'Page', 'Headers', 'Cookies', 'Meta', 'Comments', 'Scripts', 'Styles', 'Forms', and 'Wappalyzer'. The 'Page' sub-tab is currently selected.

The main content area is divided into two panels. The left panel displays page metadata:

- Link: <https://security-crawl-maze.app>
- Path:
- Title: CrawlMaze - Testbed for Web Crawlers.
- Http Version: HTTP/2.0
- Method: GET
- Status: OK
- Status Code: 200
- Depth: 0
- Size: 762
- Time: 624
- Content Type: text/html; charset=utf-8

The right panel shows a donut chart representing the link distribution. The chart is divided into two segments: a larger green segment labeled 'Internal' and a smaller red segment labeled 'External'.

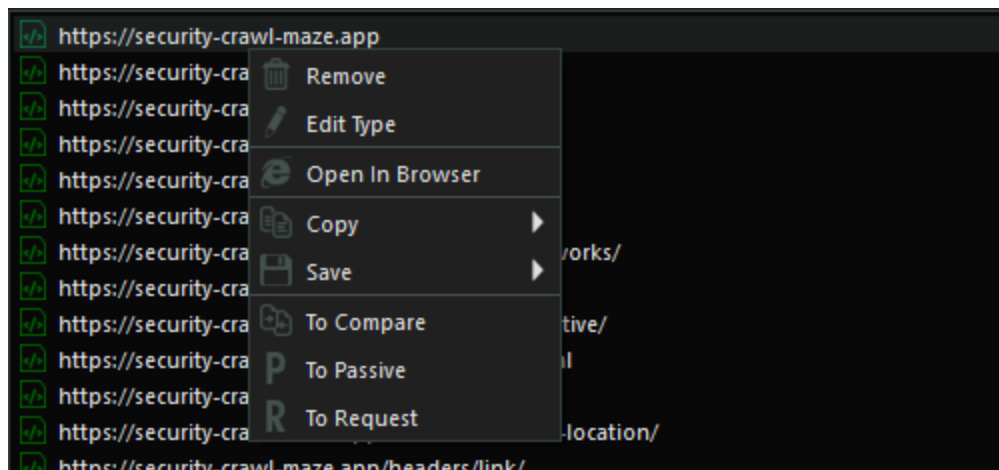
Below the main content area, there is a list of links categorized into 'Internal Links' and 'External Links'.

- Internal Links:**
 - <https://security-crawl-maze.app/headers/>
 - <https://security-crawl-maze.app/misc/>
 - <https://security-crawl-maze.app/css/>
 - <https://security-crawl-maze.app/javascript/>
 - <https://security-crawl-maze.app/html/>
- External Links:**
 - <https://github.com/google/security-crawl-maze>

At the bottom of the interface, there is a vertical sidebar with icons for various file types: Javascript (JS), CSS, JSON, XML, Documents, Databases, and Images.

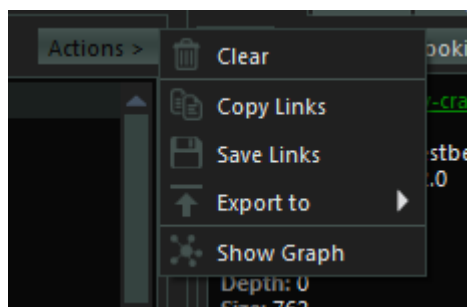
You can browse the structure and source tab to view all the content extracted from the particular page you clicked on.

You can perform desired actions on a page on sitemap by right clicking on it and choosing the action you want to perform.



Or

You can perform desired actions on the entire list on sitemap by clicking on the **Actions** button. Please note that the Actions button is only activated if there are links on the sitemap.

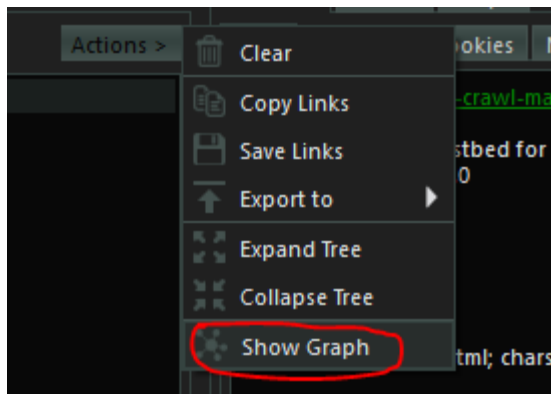


Graph

SpiderSuite has the capabilities of visualizing the links on the sitemap using a graph and also ability to manipulate the graph to your liking.

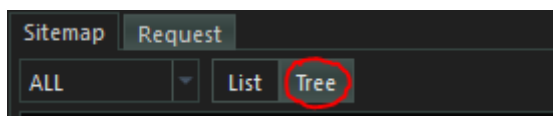
- Visualize the Entire sitemap's on a graph.

Simply click on the **Actions** button and click on **Show Graph** action to visualize the graph.

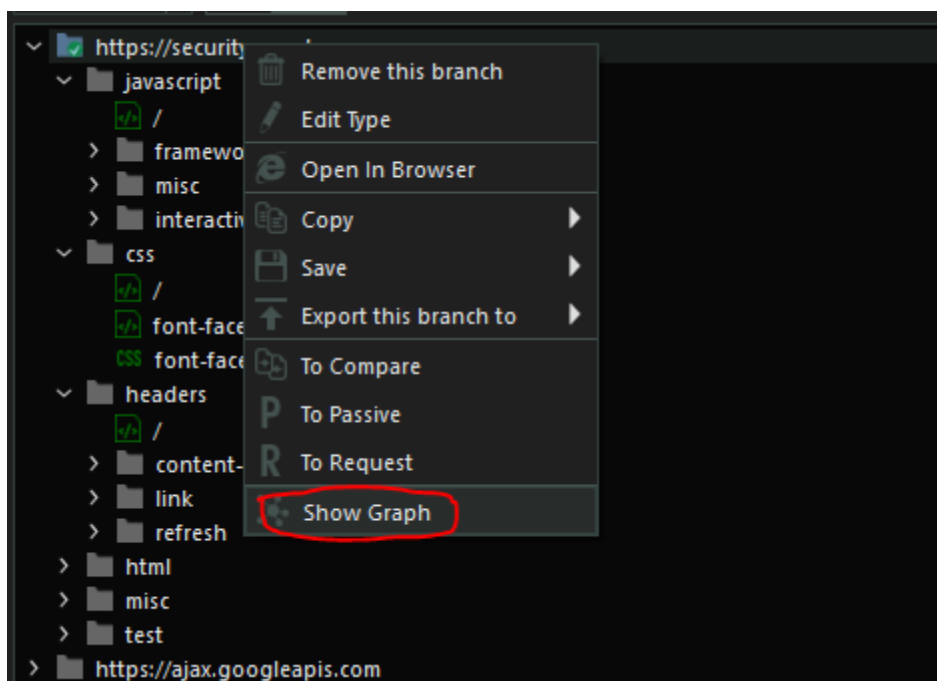


- Visualize a sitemap branch on a graph.

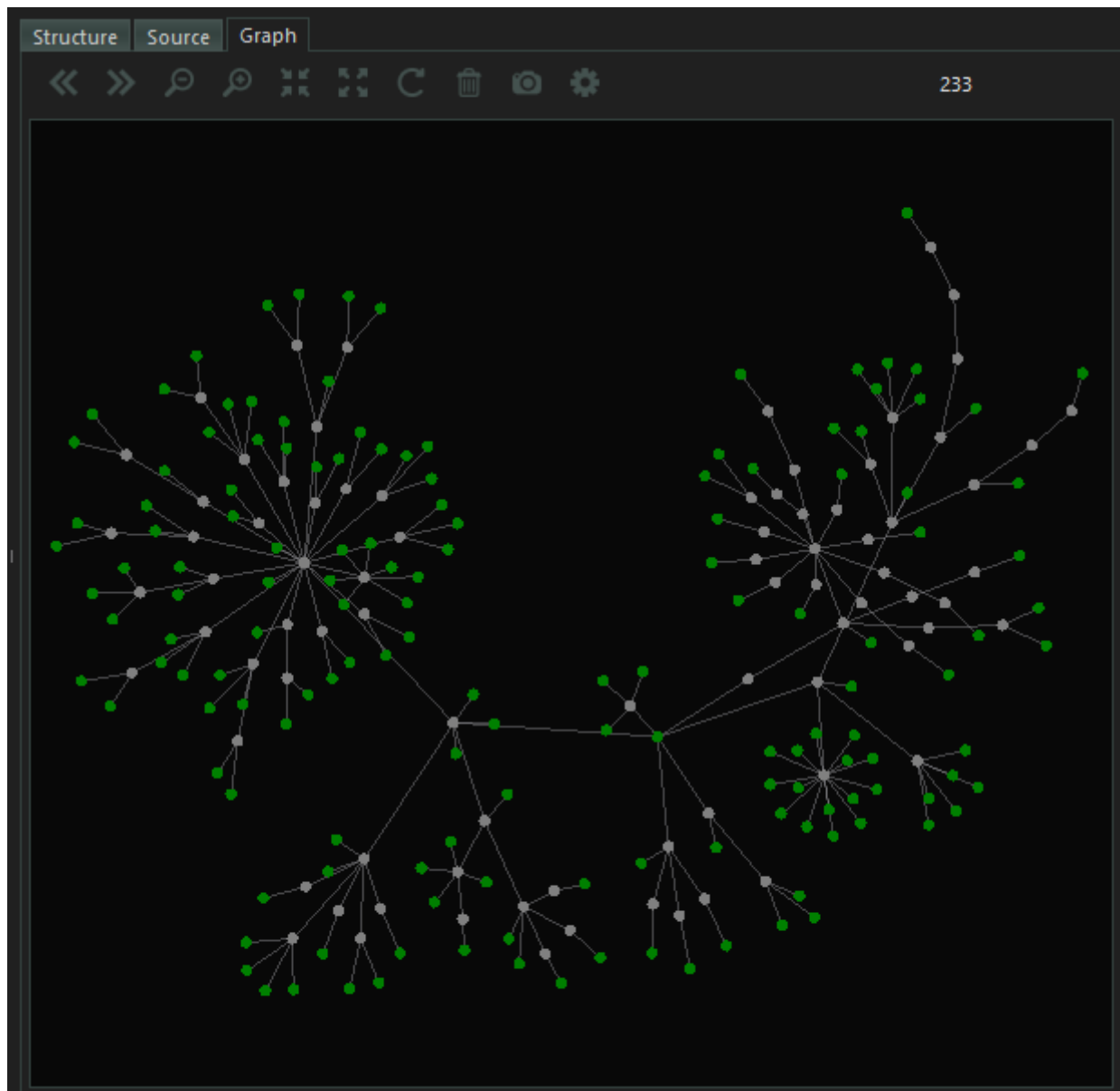
First change the Sitemap's view to **Tree** view.




Then simply **right click** on the branch you want to visualize and click on **Show Graph**. This will only show the graph for that particular chosen branch.



The Graph:



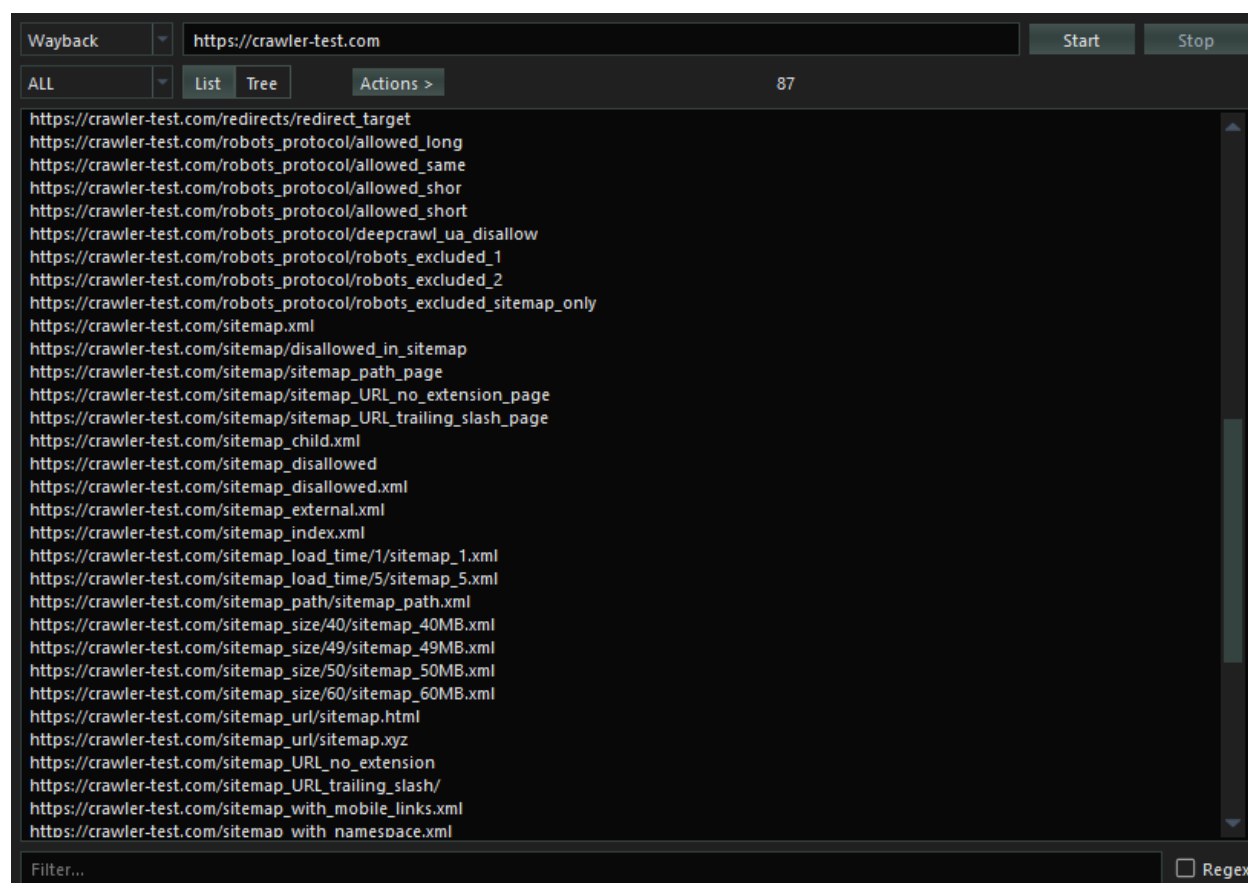
- You can manipulate the graph to your liking by simply clicking on the  **config** **action** icon on graph menu bar and set your desired configurations.

TOOLS

List of Tools Available in Spider Suite version 1.0.0

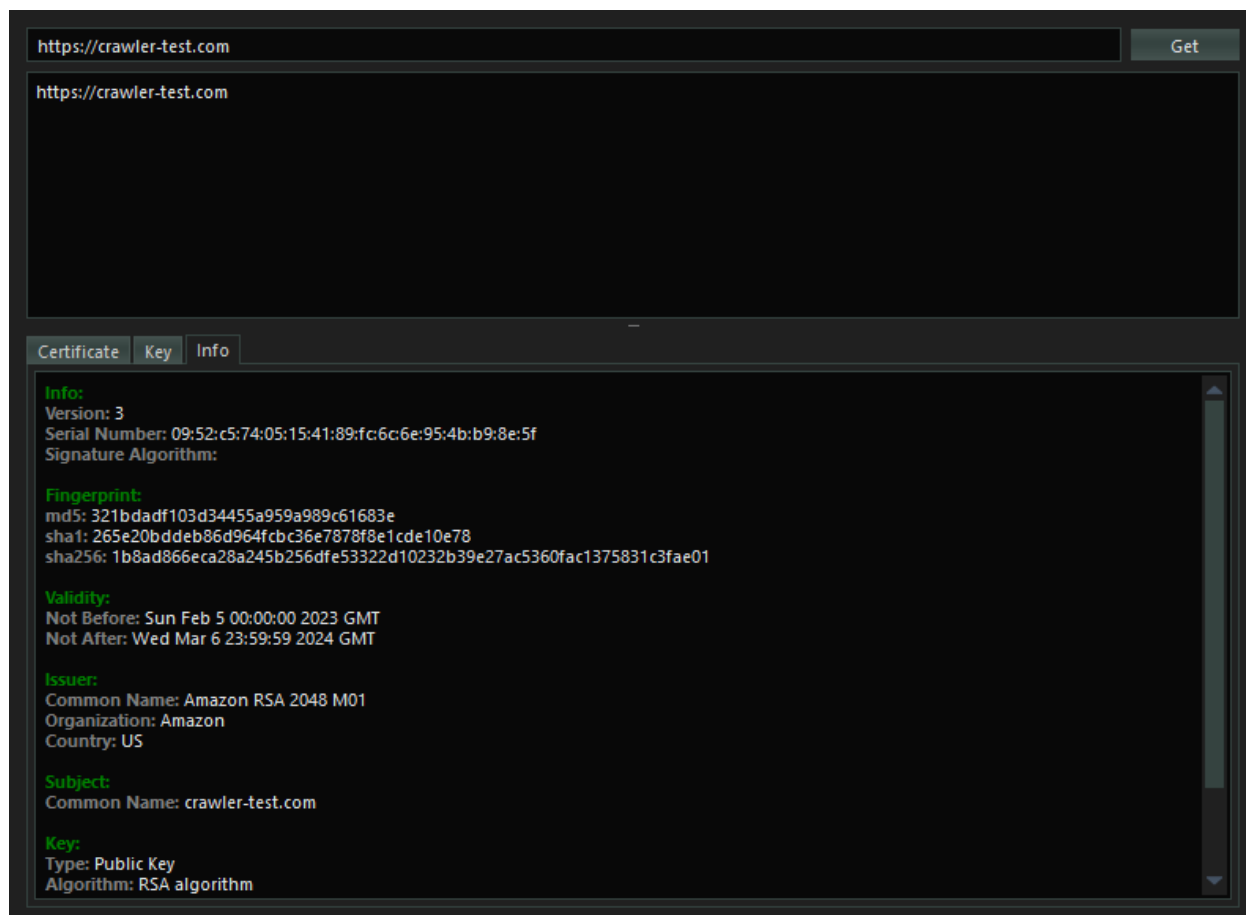
Passive Crawler Tool

Uses OSINT (open source intelligence) sources: **waybackmachine** and **Arquivo** to obtain all publicly available url links of the target. You can use the obtained links as seed links for the next crawl.



SSL Certificates Tool

Fetches SSL certificates of a particular target hostname. It does this by trying to establish a secure connection to the hostname and when done it returns the target hostname SSL certificate and closes the connection.



Decoder Tool

Encodes, Decodes or **Hashes** the input data using the chosen encoding, decoding or hashing algorithm.

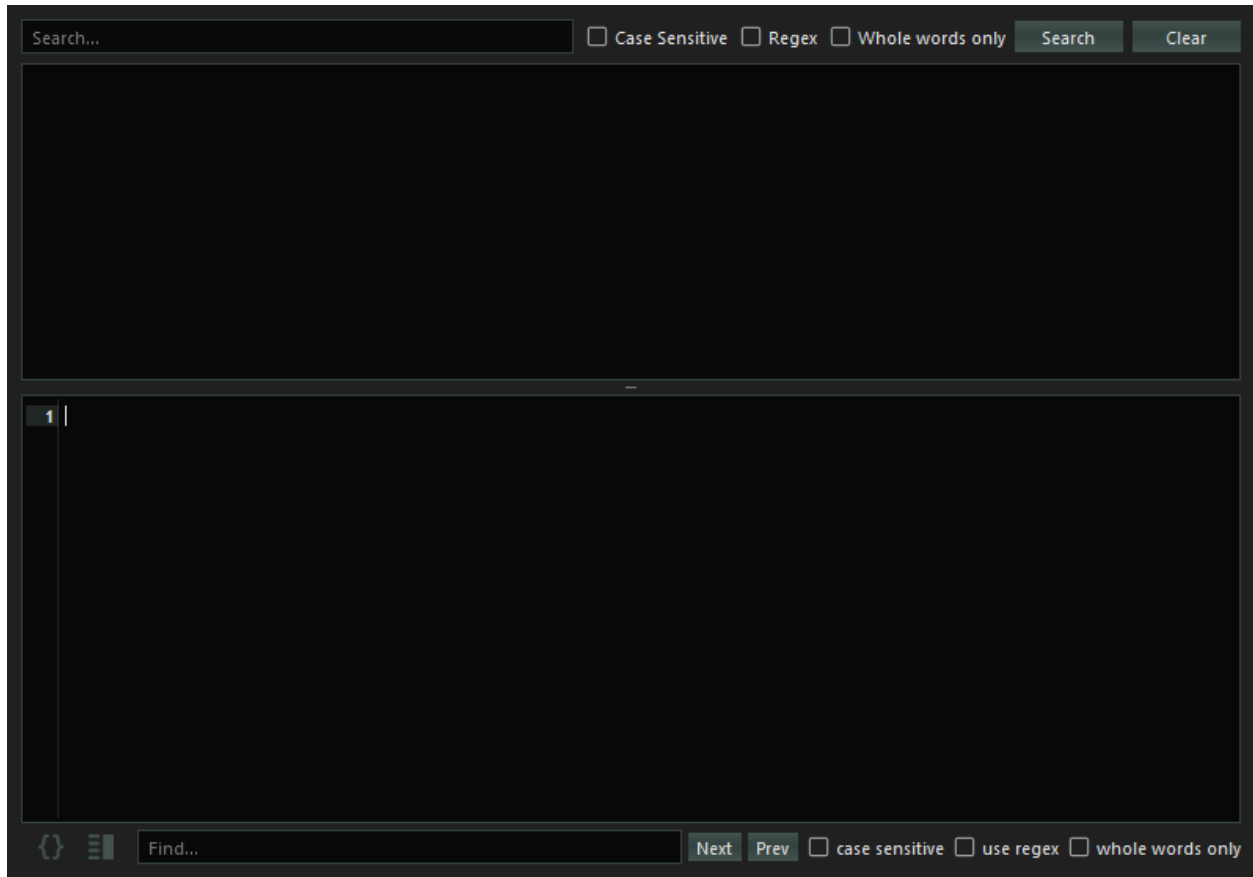
- **Encoding algorithms:** URL, HTML, Base64, Hex, Gzip and Brotli
- **Hashing algorithms:** Md4, Md5, Sha1, Sha224, Sha256, Sha384, Sha512, Sha3_224, Sha3_256, Sha3_384, Sha3_512, Keccak_224, Keccak_256, Keccak_384, Keccak_512

The image shows a web-based interface for a decoder tool. At the top, there are three main sections: 'Decode As:', 'Encode As:', and 'Hash Algorithm:'. Each section has a dropdown menu and a corresponding button. The 'Decode As:' dropdown is set to 'URL' with a 'Decode' button. The 'Encode As:' dropdown is set to 'URL (special ch)' with an 'Encode' button. The 'Hash Algorithm:' dropdown is set to 'Md4' with a 'Hash' button. Below these are two large text input areas. The top area is labeled 'Text to Decode/Encode/Hash' and the bottom area is labeled 'Decoded/Encoded/Hashed Text'. To the right of each input area are radio buttons for 'Text' (selected) and 'Hex', and buttons for 'Clear' and 'Paste' (for the top area) or 'Copy' (for the bottom area).

Many other encoding, decoding and hashing algorithms will be added in the coming versions.

Search Tool

Searches the current project's data in the database and returns all the pages that contains that particular search query. The search can take a long or short period depending on the size of the project.



Compare Tool

Compares two different pages or crawls then highlights the differences and similarities of the two pages or crawls.

- **Compare Pages**

Compares two pages for any differences and similarities.

The screenshot shows the 'Compare Pages' tool interface. At the top, there are two tabs: 'Compare Pages' (selected) and 'Compare Crawls'. Below the tabs, there are two dropdown menus: 'Algorithm:' set to 'Diff by Lines' and 'Views:' set to 'Side by Side'. To the right of these is a checkbox labeled 'Beautify Before Comparing' which is checked. The main area is divided into two sections, 'Page #1' and 'Page #2'. Each section has a table with two columns: 'Link' and 'Source'. The 'Link' column is currently empty. To the right of the 'Page #1' table are three buttons: 'Fetch', 'Load', and 'Clear'. At the bottom right of the 'Page #2' section is a 'Compare' button.

Compare Pages	
Algorithm: Diff by Lines Views: Side by Side <input checked="" type="checkbox"/> Beautify Before Comparing	
Page #1	
Link	Source
Page #2	
Link	Source
Compare	

- **Compare Crawls**

Compares two project crawls for any differences among similar pages.

The screenshot shows the 'Compare Crawls' interface within the Spider Suite. At the top, there are two tabs: 'Compare Pages' and 'Compare Crawls', with the latter being the active tab. Below the tabs, the interface is divided into two main sections for 'Crawl #1' and 'Crawl #2'. Each section contains a large, empty rectangular area for displaying crawl data. To the right of the 'Crawl #1' area, there are two buttons: 'Load' and 'Clear'. At the bottom right of the 'Crawl #2' area, there is a 'Compare' button. The entire interface has a dark, monochromatic theme.

CONTACTS

You can contact us by email, twitter or telegram chat for any issue or inquiry and we will get back to you as soon as possible.

Email:

spid3rsuite@gmail.com

enock.n.michael@gmail.com

Twitter:

https://twitter.com/spider_suite

Telegram chat:

<https://t.me/SpiderSuite>