CookieCrumbs User Manual

1. Introduction

CookieCrumbs is a browser extension designed to enhance user privacy and security by decentralizing cookie storage. Traditional cookie storage methods rely on centralized servers, making them vulnerable to XSS attacks, session hijacking, and unauthorized access. CookieCrumbs mitigates these risks by offering a secure alternative for handling cookies

2. Installation Guide (Firefox)

# 2.1. Installing the zipped file in GitHub

1. Enter the URL “<https://github.com/tscodes70/CookieCrumbs>”
2. Click on “< > Code” and click“Download Zip” found in the popup

|  |
| --- |
| Figure 1: GitHub repository page showing the “Code” button for downloading a zipped file. |

1. Download the extension's .zip file and extract it.

|  |
| --- |
| Figure 2: The File Explorer context menu for the downloaded ZIP file, highlighting the “Extract All…” option to unzip the extension. |

# 2.2. Firefox

1. Open your firefox browser and enter the following URL:

about:debugging#/runtime/this-firefox

1. Click on "Load Temporary Add-on"

|  |
| --- |
| Figure 3: The Firefox about:debugging page shows the “Load Temporary Add-on…” button, which is used to temporarily install the extension. |

Select the extracted folder and select “manifest.json”

.

|  |
| --- |
| Figure 4: The extracted extension folder in File Explorer, showing “manifest.json” as the file to select. |

3. Getting Started

Click on the extension icon in the toolbar.

Grant necessary permissions if prompted.

Configure the extension settings as needed.

Start using the features as described below.

4. Features & Usage

## Overview of CookieCrumbs UI:

The Popup Extension Interface in CookieCrumbs provides a quick way to manage and monitor cookies for different domains. From this popup, you can retrieve cookie data, define rules for how cookies are stored, and audit or clear these rules. This interface allows you to control cookie behavior without having to manually inspect or edit browser settings

|  |
| --- |
| Figure 5: The CookieCrumbs extension’s popup interface, showing a text field to retrieve cookie data, a display of cookie rules, and management options (e.g., “View Rules” and “Clear Rules”). |

## 4.1. Retrieve Cookie Data

The **Retrieve Cookie Data** feature allows you to view all cookies stored for a specific domain. When you enter a domain (for example, www.instagram.com), the extension queries the browser for any cookies related to that domain and displays them in a detailed table. This table shows important information such as the cookie name, value, domain, expiration date, and security attributes. By reviewing these details, you can better understand which cookies a site is using and how they are configured

How to use it.

1. **Open the Popup:**

* Click on the CookieCrumbs icon in your browser toolbar to open the popup.

1. **Enter the Domain upon accessing a website:**

* Access a certain website, for example [www.instagram.com](http://www.instagram.com)
* In the “Enter Domain Name” field at the top, type the domain you want to inspect (e.g., [www.instagram.com](http://www.instagram.com)).

1. **Click "Retrieve Cookies":**

* Press the **Retrieve Cookies** button.
* CookieCrumbs will fetch the cookies associated with that domain from your browser.

1. **Review the Results:**

* The retrieved cookie data will appear in a table below.
* You can see fields such as:
  + **name:** The name of the cookie (e.g., csrftoken).
  + **value:** The cookie’s value (often a unique string).
  + **domain:** The domain for which the cookie is valid (e.g., www.instagram.com).
  + **path:** The URL path where the cookie is accessible.
  + **expires:** The expiration date and time of the cookie.
  + **httpOnly:** Indicates if the cookie is accessible only via HTTP/HTTPS (not JavaScript).
  + **secure:** Specifies if the cookie should only be sent over secure (HTTPS) connections.
  + **sameSite:** Defines the cross-site request context for the cookie (e.g., None, Lax, or Strict).
  + **chunkCIDs or totalChunks:** These fields show how the cookie data is broken into chunks and stored.

|  |
| --- |
| Figure 6: The CookieCrumbs extension retrieving cookie data for instagram.com, displaying detailed cookie attributes (e.g., name, value, domain) in a structured table. |

## 

## 4.2. Cookie Rules Management

The Cookie Rules Management feature lets you define how cookies are handled on a per-domain basis. By specifying rules in JSON format, you can decide whether to store cookies for certain domains and for how long. For example, you might choose to block cookies from www.facebook.com but store cookies from www.instagram.com for a specific number of seconds. These rules are saved within the extension, allowing CookieCrumbs to consistently apply your preferences across browsing sessions.

**How to use it**

1. **Enter Your Rules:**
   1. In the text area labeled **“Cookie Rules,”** write a valid JSON object.
   2. Each key should be a domain (e.g., "www.facebook.com"), and the value should be an object defining the rules, such as:

*{ "store": true, "storageDuration": 86400 }*

**store:** A boolean indicating whether to store cookies (true) or block them (false).

**storageDuration:** An integer representing how long (in **seconds**) the cookies should be retained. For instance, 86400 seconds is equivalent to 24 hours.

1. **Save Rules:**
2. Click **“Save Rules”** to store your JSON object in the extension’s storage.
3. Ensure the JSON is valid before saving, or the extension will show an error.
4. **ViewRules:**
5. Click **“View Rules”** to display the rules you previously saved.
6. This helps you confirm that your rules are correctly set and for reference what rules have been set for your browsing session
7. **Clear Rules:**
8. Click **“Clear Rules”** to remove all existing domain-based cookie rules.
9. Once cleared, the extension will no longer apply any special handling for cookies.

.

|  |
| --- |
| Figure 7: The CookieCrumbs extension’s cookie rules interface, showing domain-specific rules (e.g., “www.facebook.com” and “www.instagram.com”) and buttons to save, view, or clear these rules. |

## 4.3. Audit Logs

The **Audit Logs** feature provides a chronological record of which cookies have been stored by CookieCrumbs. Each log entry includes details such as a timestamp, the action taken, the cookie name, and the associated domain. By reviewing these logs, you can see precisely which cookies are actively being saved and retrieved, this gives more control to the users by providing insights on what is being stored and retrieved.

**How to use it:**

1. **View Audit Logs:**
   * Click **“View Audit Logs”** to see a list of all cookies that have been stored.
   * A JSON-like output will appear, displaying entries with fields such as:
     + **timestamp:** The time the cookie was stored (in Unix epoch format).
     + **action:** The action performed (e.g., "stored").
     + **name:** The name of the cookie.
     + **domain:** The domain (or sub-domain) associated with the cookie.
2. **Clear Audit Logs:**
   * Click **“Clear Audit Logs”** to delete all existing log entries.
   * A confirmation message (e.g., **“Audit logs cleared successfully!”**) will appear, indicating that the logs have been removed..

|  |
| --- |
| Figure 8: The CookieCrumbs extension’s “Audit Cookie Data” panel, displaying JSON-formatted logs (timestamp, action, domain) and buttons to view or clear the stored audit logs after storing cookies from accessing www.google.com |

## 4.4. Import/Export Cookie Data

The **Import/Export Cookie Data** feature allows you to back up and restore your cookie data using a structured JSON format. When exporting, CookieCrumbs automatically filters out any cookies that are marked by the Cookie Rules Management as "not to be stored." (refer to [4.2](#_318st56k4590)) This ensures that only the cookies permitted by your current rules are backed up, allowing you to safely share or restore data without including unwanted cookies.

**How to use it**

1. **Export Cookies:**
   * **Action:** Click **“Export Cookies”** in the popup interface.
   * **What Happens:** The extension compiles all cookie data that complies with your current storage rules, meaning any cookies for which the rule is set to not store will be excluded from the export.
   * **Outcome:** A JSON file (e.g., exported\_cookies.json) is automatically downloaded to your default download folder. This file contains only the permitted cookie data, ensuring your export reflects your desired cookie storage settings.
2. **Import Cookies:**
   * **Action:** Open your exported JSON file, copy its contents, and paste them into the **“Import Cookie Data”** textarea in the popup interface.
   * **What Happens:** The extension validates and parses the JSON data.
   * **Outcome:** If the JSON is valid, the cookie data is imported successfully and a confirmation message is displayed (e.g., **“Import completed successfully!”**). The imported data now becomes available for retrieval and management via the extension.
3. **Confirm or Manage Your Data:**
   * After importing, you can verify the restored cookies by using the **Retrieve Cookie Data** feature.
   * This process ensures that your backup or transferred data contains only the cookies that were approved for storage, in line with your current cookie management policies.

|  |
| --- |
| Figure 9: The CookieCrumbs extension’s Import/Export Cookie Data interface, featuring a text area to paste JSON data for importing and an “Export Cookies” button to download a filtered JSON file. |

|  |
| --- |
| Figure 10: The browser’s download prompt indicating the “exported\_cookies.json” file has been successfully saved. |

|  |
| --- |
| Figure 11: With cookie rules set in place not to store a domain, for example [www.instagram.com](http://www.instagram.com), only the symmetric key is shown to be available for import |

|  |
| --- |
| Figure 12: A confirmation dialog in the CookieCrumbs extension indicating that the cookie data was imported successfully. |

|  |
| --- |
| Without a specific-domain cookie rules set in place, for example [www.instagram.com](http://www.instagram.com):, the cookies is available to be set in place for import |

## 

## 

|  |
| --- |
| Figure 12: A confirmation dialog in the CookieCrumbs extension indicating that the cookie data was imported successfully with the cookies of www.instagram.com |

## 

## 4.5 Cookie Stealer

Cookie stealer is a locally hosted web application hosted with Node.js. It serves to show the intercepting capabilities of the extension. To run it, ensure Node has been installed beforehand.

**How to use it**

1. **Ensure node has been installed:**
   * Run ‘npm -v’ to test.
2. **Ensure node dependencies have been installed:**
   * Run ‘npm install’ inside the cookie-stealer directory.
3. **Start the web application:**
   * Run ‘npm start’ in the cookie-stealer directory.
4. **The website will be accessible locally:**
   * go to ‘<http://localhost:3000>’ on Firefox.
5. **Test the website functionality is working:**
   * upon clicking login WITHOUT the extension, there should be a cookie made with the details ‘session\_id=abcd1234’. This can be viewed in developer tools.
   * Click on the ‘Test XSS’ button to ensure the cookie details are visible.
   * Upon logout,the cookie should be deleted.
6. **Verify extension functionality:**
   * On login, the cookie should not be within the cookie storage locally, but users can still access the home.html page with no issues.
   * Extension should have intercepted the cookie request and can be viewed through UI.

|  |
| --- |
| Figure 13: The Node.js download page showing instructions for installing Node.js (v22.14.0 LTS), along with commands to verify and manage Node.js via npm and fnm. |

|  |
| --- |
| Figure 14: A terminal session running “npm install” in the cookie-stealer directory, installing the required Node.js dependencies. |

|  |
| --- |
| Figure 15: The terminal output after running “npm start,” confirming the cookie-stealer server is running on http://localhost:3000. |

|  |
| --- |
| Figure 16: The locally hosted web application displaying “Simulated Cookie Setting” at http://localhost:3000, ready for testing. |

|  |
| --- |
| Figure 17: The “Vulnerable Home Page” with Developer Tools open, showing a session cookie (e.g., “session\_id=abcd1234”) stored by the website. |

|  |
| --- |
| Figure 18: A side-by-side view of the “Vulnerable Home Page” and the CookieCrumbs extension UI, illustrating how the extension intercepts and displays the cookie data. |