WEBSITE TRAFFIC AND SECURITY ANALYSER

[1]Dr. Pampapathi B M, [2]Manjunatha Gouda K, [3]Bharath G, [4]B Srinidhi
[*1]Associate Professor, Department of Computer Science & Engineering, RYM Engineering College, Ballari, VTU Belagavi, Karnataka, India
[234]Department of Computer Science & Engineering, RYM Engineering College, Ballari, VTU Belagavi, Karnataka, India

Abstract: This Chrome extension, Traffic Analyzer, empowers users with real-time website security analysis and historical trend tracking, promoting informed decisions about online safety. Upon installation, the extension seamlessly integrates into the browser, capturing network requests and analysing critical security aspects. It leverages Google Safe Browsing to assess URL reputation, flagging potentially malicious or phishing sites. SSL/TLS certificate strength is evaluated using SSL Labs, providing insights into the security of website connections. Content Security Policy (CSP) headers are analysed to determine the website's protection against content injection attacks. The dashboard offers a comprehensive overview of current site information, detailed traffic statistics, and cookie information. Users can easily switch between a popup view and a detailed side panel, further enhancing usability. The side panel offers customizable theme options (light, dark, and automatic) to cater to user preferences. Furthermore, the extension allows users to define alert thresholds and filter traffic data, thereby personalizing their online security experience. By combining these powerful features, Traffic Analyzer serves as an indispensable tool for understanding and improving online safety.

Keywords: Website Security, Website traffic, Domains, CSP, Statistics, Cookies, Security, File-type, Size, Speed/Delay.

I. INTRODUCTION

In today's interconnected digital landscape, ensuring online safety has become paramount. Users frequently encounter websites with hidden security vulnerabilities, exposing themselves to risks like phishing attacks, malware infections, and data breaches. Often, these threats remain invisible to the untrained eye, necessitating a readily accessible and user-friendly tool for evaluating website security. Traffic Analyzer, a Chrome extension, addresses this critical need by providing real-time traffic analysis and comprehensive security assessments. By empowering users with knowledge and insights into website security practices, Traffic Analyzer fosters a safer online experience. Traffic Analyzer operates seamlessly in the background, capturing network requests and analysing various security aspects of visited websites. It leverages a combination of powerful technologies and external APIs to provide a multi-faceted security assessment. These include checks for URL reputation using Google Safe Browsing, SSL/TLS certificate strength evaluation via SSL

Labs, and Content Security Policy (CSP) header analysis. Results are presented in an intuitive dashboard, displaying historical trends, and aggregated statistics for each domain. This comprehensive overview empowers users to make informed decisions about website trust and navigate the web with greater confidence.

Beyond real-time analysis, Traffic Analyzer offers valuable historical data visualization through interactive charts. Users can track security trends over time for specific domains, gaining a deeper understanding of website security practices and potential risks. The extension also provides detailed traffic statistics, cookie information, and the ability to export historical data in various formats for further analysis. With customizable alert thresholds and data filtering options, Traffic Analyzer puts users in control of their online security, creating a personalized and proactive approach to safeguarding sensitive information. By combining cutting-edge technology with user-friendly design, Traffic Analyzer emerges as an indispensable tool for enhancing online safety and promoting web security awareness.

II. REVIEW OF LITERATURE

The evolving digital landscape has brought with it an array of web security threats that target both individuals and organizations. Researchers have extensively documented the prevalence and impact of attacks such as Cross-Site Scripting (XSS), phishing, malware distribution, and Man-in-the-Middle (MitM) attacks. In particular, *The Web Application Hacker's Handbook* by Stuttard and Pinto (2011) provides a comprehensive overview of these vulnerabilities, detailing attack methodologies and offering insights into effective countermeasures [31].A major focus in the literature is the deployment of defensive techniques to mitigate these threats. The implementation of a robust Content Security Policy (CSP) is widely recognized for its effectiveness in preventing XSS attacks by restricting the sources from which content can be loaded (Barth, Jackson, & Mitchell, 2008) [36]. Similarly, Subresource Integrity (SRI) has been identified as a critical mechanism for ensuring that externally sourced scripts or assets remain untampered, thereby guarding against supply chain attacks (Böttinger, Berger, & Kirchner, 2017) [20].

Moreover, the literature emphasizes the importance of secure communication channels. The widespread adoption of HTTPS, as detailed by Rescorla (2001), underscores its role in encrypting data between users and web servers, thereby preventing eavesdropping and data manipulation [41]. In addition to these technical measures, two-factor authentication (2FA) has emerged as an indispensable layer of security, as demonstrated by Bonneau et al. (2012), who compare various authentication schemes and highlight the enhanced security offered by 2FA despite its usability challenges [29]. These foundational research findings inform the security checks integrated within Traffic Analyzer, ensuring that users receive a comprehensive overview of website vulnerabilities.

Browser extensions have become ubiquitous in enhancing user experience and functionality; however, they also introduce unique security and performance challenges. The architecture and permission models of extensions are critical areas of study. Jiang et al. (2018) provide a detailed analysis of the security implications inherent in browser extensions, emphasizing the need for strict permission models to minimize the risk of malicious behavior [18].

The Chrome Extension API, for instance, offers developers a structured environment within which extensions operate. Google Inc. (2023) outlines best practices for interacting with browser functionalities while remaining within the confines of a secure sandbox environment, thereby reducing the potential for unauthorized data access [12]. Research by Wu (2016) further illustrates that over-permissioning is one of the primary vulnerabilities that can be exploited by malicious extensions. Such insights have been pivotal in guiding the development of Traffic Analyzer, ensuring that it requests only the necessary permissions and implements efficient background processes that do not degrade browser performance[22].

Data visualization plays a pivotal role in translating complex security assessments into actionable insights for users. The challenge lies in presenting multifaceted security data in a way that is both accessible and comprehensible. Few (2009) has shown that well-designed visualizations significantly enhance the user's ability to discern patterns and anomalies in quantitative data [34]. Cognitive studies of visual perception, as discussed in this work, have influenced the design of dashboards that use intuitive visual representations such as radar charts for security scores and line charts for historical trend analysis.

The effective use of interactive visualizations can empower users to delve deeper into security metrics, thereby fostering a better understanding of the underlying risks. By leveraging these visualization techniques, Traffic Analyzer translates intricate security assessments into clear, actionable information. This not only enhances user engagement but also bridges the gap between technical security data and user comprehension, allowing individuals to make more informed decisions regarding website trust and safety.

Understanding user behaviour is essential for designing security tools that effectively promote safe online practices. Studies on user behaviour and security awareness have highlighted that many users, often due to limited technical knowledge, are prone to overlooking subtle security warnings. Cranor (2008) provides a framework for understanding how users process security information and the common pitfalls that lead to misinterpretation or disregard of security alerts [37].

Research in this area suggests that clear, contextually relevant feedback is crucial to altering user behaviour. By offering real-time security feedback and detailed explanations of potential vulnerabilities, Traffic Analyzer addresses the prevalent gap in user education. Such tools can foster improved risk perception and encourage safer online practices. The emphasis on user empowerment is consistent with broader findings in the literature, which advocate for security solutions that are both technically robust and user-centric.

III. METHODOLGY AND IMPLEMENTATION

A. Implementation

The implementation language for this Chrome extension is JavaScript. Here's why:

- Chrome Extensions API: Chrome extensions are built using web technologies, and the core logic of extensions interacts with the Chrome Extensions API, which is primarily JavaScript-based. The code uses functions like chrome.action.onClicked.-addListener, chrome.webRequest.onBeforeRequest.addListener,chrome.storage.local.get, chrome.runtime.sendMessage, and chrome.sidePanel.open all part of the Chrome Extensions API and accessible only through JavaScript.
- Web Technologies in Popup and Side Panel: The popup (popup.html/popup.js) and side panel (sidepanel.html/sidepanel.js) UIs are built with standard web technologies: HTML, CSS, and JavaScript. These components handle user interaction, data display, and charting (using Chart.js, a JavaScript library).
- Event Handling and DOM Manipulation: The code extensively uses JavaScript for event handling (e.g., click listeners, message listeners) and manipulating the Document Object Model (DOM) to update the UI dynamically.
- Background Script Execution: The background script (background.js) runs persistently in the background of the extension and is written in JavaScript. This allows it to intercept web requests, process data, and communicate with the popup/side panel.

In summary, JavaScript is the natural and required choice for developing Chrome extensions due to the Chrome Extensions API being JavaScript-based and the use of standard web technologies for the UI elements. No other language can directly interact with these APIs or manipulate the extension's UI in the same way.

B. System Design

Manifest File (manifest.json)

Purpose and Structure: The manifest file serves as the blueprint for the extension. It declares essential metadata such as the extension's name, version, description, and icons. It also specifies critical information that the browser uses to understand what the extension does and what resources it requires.

Permissions: To function effectively, Traffic Analyzer requests specific permissions such as access to:

- **Web Requests:** For intercepting HTTP/HTTPS requests via the chrome.webRequest API.

- **Storage:** For saving real-time data and aggregated statistics using chrome.storage.local.
- **Cookies:** For retrieving cookie details from the active tab using the chrome.cookies API.
- **Side Panel API:** To toggle between popup and side panel views.

Declared Scripts and UI Elements: The manifest specifies the background script (background.js) responsible for core processing, as well as the popup and side panel HTML files (popup.html and sidepanel.html). It also defines the behavior of the extension's action icon, determining how the UI is triggered when the user clicks the icon.

Background Script (background.js)

The background script is the backbone of the extension's functionality, handling the core tasks of network request interception, data aggregation, and inter-component communication.

Request Interception:

- **chrome.webRequest API:** The extension utilizes the chrome.webRequest API to monitor all network requests made by the active tab.
- **onBeforeRequest Listener:** When a request is initiated, the onBeforeRequest listener logs the start time, storing it in an object indexed by the request ID.
- **onCompleted Listener:** Once a request completes, the onCompleted listener retrieves the stored start time, computes the duration, and extracts additional details such as the URL, request type, and the content-length (if available). This data is crucial for both performance analysis and security assessment.

Data Storage:

chrome.storage.local:
The script uses Chrome's local storage to persistently store individual request details and aggregated statistics. This ensures that even if the extension or browser restarts, historical data remains available for analysis.

Domain Statistics Aggregation:updateDomainStats Function: This function processes each completed request to update statistics for its corresponding domain. It aggregates key metrics like the number of requests, total duration, and total data transferred, providing an overall picture of each domain's performance and security posture.

Real-time Updates:

chrome.runtime.sendMessage:
After processing each request, the background script sends real-time updates to the user interface (popup or side panel). This enables the UI to refresh dynamically and display the most recent data without requiring manual intervention.

Side Panel Management:

- **User Interaction:** The background script listens for messages from the UI regarding panel toggling. Depending on user actions (e.g., clicking the extension icon), it toggles between a compact popup view and a more comprehensive side panel.
- **chrome.sidePanel API:** When the side panel is required, the script leverages the chrome.sidePanel API to open it. Conversely, it closes the side panel when reverting to the popup view, ensuring that the user experience remains smooth and contextually appropriate.

Popup/Side Panel UI (popup.js, popup.html, sidepanel.js, sidepanel.html)

The user interface is responsible for presenting the captured data in an accessible and interactive format. Both the popup and side panel share a significant portion of their logic and presentation styles, ensuring a consistent user experience.

Shared Logic Across Popup and Side Panel:

- **Data Display and Updates:** Both components include scripts (popup.js and sidepanel.js) that retrieve aggregated statistics and individual request data from Chrome's local storage. They then update the display elements—such as counters, charts, and tables—to reflect real-time and historical data.
- **Event Handling:** Common UI elements include buttons and dropdowns for actions like clearing statistics, filtering requests by type, toggling cookie displays, and switching themes. These event listeners trigger corresponding functions that interact with the background script or update the UI directly.

Chart.js Integration:

Visualization:
The extension uses Chart.js to render visualizations such as doughnut or radar charts. These charts represent the distribution of request types, trends over time, and domain-specific statistics, providing users with an intuitive understanding of network activity and potential security issues.

Data Display:

- **HTML Structure:** The popup.html and sidepanel.html files define the layout and structure for displaying data. They include sections for overall traffic statistics, detailed domain statistics, and dynamic charts.
- **Real-time Feedback:** As new data is received from the background script, the UI elements update automatically, ensuring that users always see the latest information.

Cookie Information:

- **chrome.cookies API:** Both UI modules include features to fetch and display cookies associated with the active tab. This helps users quickly assess tracking and session information, adding another layer to their security analysis.
- **Toggle Functionality:** Users can easily switch the visibility of cookie details, allowing for a cleaner interface when detailed cookie information is not needed.

Theme Management:

Customization:

The side panel (and optionally the popup) supports theme customization. Users can choose between light, dark, or automatic themes. The theme preference is stored in localStorage and applied dynamically, ensuring that the UI is both visually appealing and accessible under various lighting conditions.

Panel Toggling:

Inter-Component Communication: Both popup.js and sidepanel.js include logic to send messages to the background script when users wish to toggle between the compact popup view and the extended side panel view. This communication ensures that the user's preference is maintained and that the appropriate interface is displayed at the right time.

C. Data flow

Data Flow refers to the movement of data within a system, from input to processing and output. It defines how data is transmitted between different components, such as databases, APIs, and user interfaces. Efficient data flow ensures smooth communication, enhances performance, and maintains data integrity within an application or network.

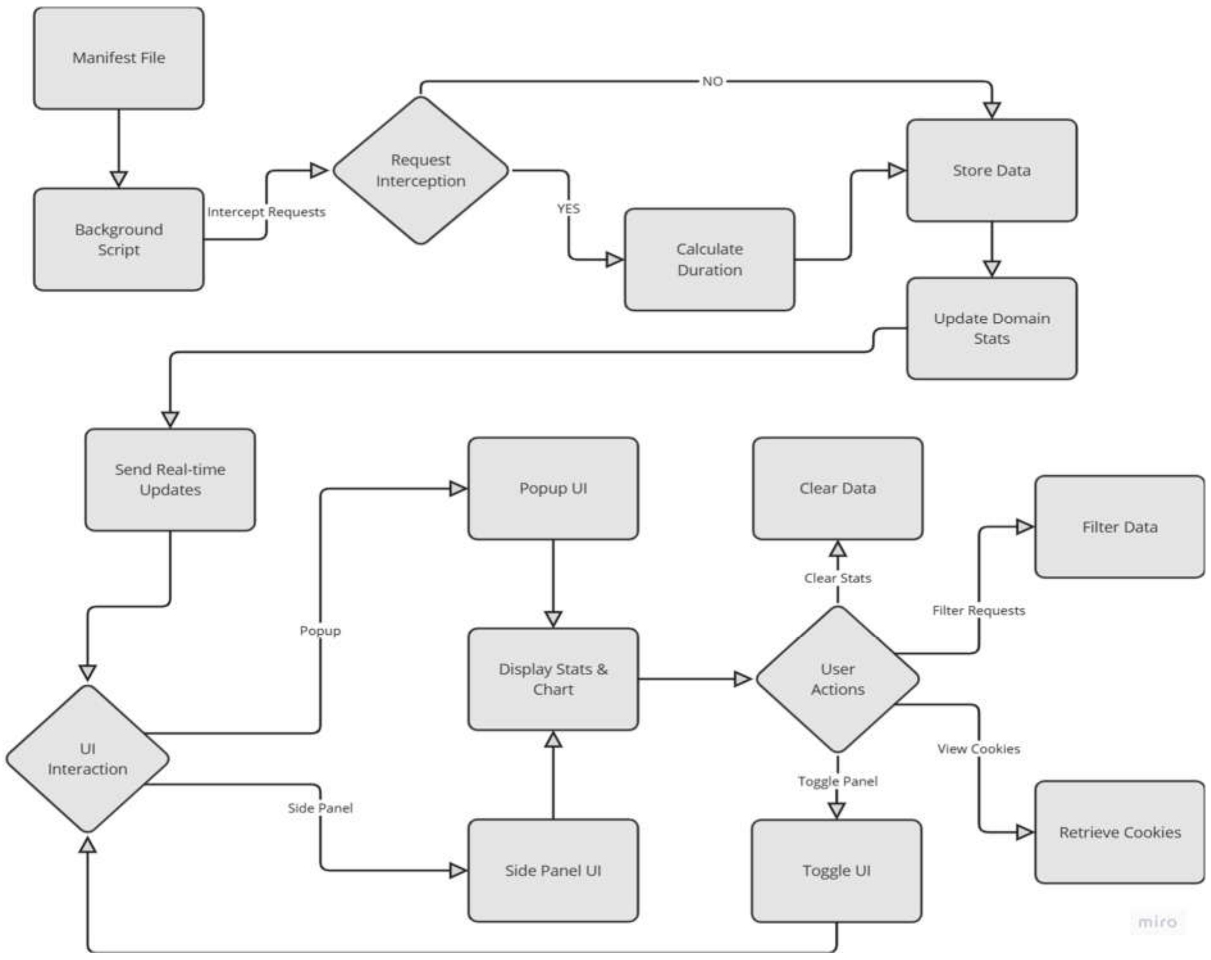


Fig.1 Data flow diagram

Request Interception: The foundation of Traffic Analyzer is its ability to intercept all web requests initiated by the active browser tab. The background script leverages the chrome.webRequest API to listen for both the initiation and completion of HTTP/HTTPS requests. When a request begins, the script captures key details such as the request ID and the start time. This mechanism allows the extension to monitor every network transaction, ensuring that no request goes unnoticed. By intercepting these requests, the extension is able to gather the raw data necessary for subsequent analysis and visualization.

Data Processing: Once a web request is intercepted, the background script processes the captured data to calculate meaningful metrics. It computes the duration of each request by subtracting the recorded start time from the completion time. In addition to timing, the script extracts other critical information including:

- **URL and Request Type:** Identifies the destination and nature of the request (e.g., GET, POST).
- **Data Size:** Reads the content-length from the request headers to determine the amount of data transferred. The script further aggregates these individual request details into domain-level statistics. Functions like updateDomainStats increment

counters, accumulate total durations, and sum the data sizes for each unique domain, providing a consolidated view of network activity.

Storage: To maintain a record of both individual requests and aggregated statistics, Traffic Analyzer utilizes Chrome's local storage (chrome.storage.local). This storage mechanism ensures that the collected data persists across sessions and can be retrieved at any time for analysis. Storing data locally allows the extension to:

- Retain historical records for trend analysis.
- Maintain a consistent state even if the extension is temporarily inactive.
- Quickly access and update the dataset without requiring constant network calls.

UI Updates: Real-time responsiveness is key to the user experience. After processing each request, the background script immediately notifies the UI (whether it's the popup or the side panel) of new data. This is achieved using the chrome.runtime.sendMessage API. When a message indicating that new data is available is sent:

- The UI scripts listen for these updates.
- Upon receiving a notification, the UI fetches the latest data from local storage.
- The display is refreshed to show up-to-date statistics, charts, and other visual elements. This real-time update mechanism ensures that users always see the current state of their web traffic and any potential security insights derived from it.

Visualization: The extension's user interface is designed to transform raw data into comprehensible visual information. Using libraries like Chart.js alongside standard DOM manipulation, the UI scripts generate dynamic charts and graphs that illustrate:

- **Distribution of Request Types:** For example, doughnut or radar charts may display proportions of GET, POST, and other request methods.
- **Historical Trends:** Line charts and bar graphs can illustrate how metrics like request count and average response times evolve over time.
- **Aggregated Domain Statistics:** The UI may also render tables or card views showing statistics for each domain, such as the number of requests made, average response time, and total data transferred. By converting numerical data into visual formats, the extension enables users to quickly grasp the overall performance and security posture of the websites they visit.

User Interaction: Beyond passive monitoring, Traffic Analyzer is built to facilitate active user engagement. Users can interact with various elements of the interface to tailor the displayed information to their needs:

- **Clear Stats Button:** Allows users to reset the accumulated data, starting fresh with current browsing activity.

- **Filtering Options:** Dropdown menus or other controls enable users to filter the displayed requests by type or other criteria, focusing on specific aspects of network activity.
- **View Cookies:** A dedicated control lets users toggle the display of cookie information related to the current website, providing additional context on data tracking and session management.
- **Theme and Display Controls:** Users can switch between different visual themes (light, dark, or automatic) and even toggle between popup and side panel modes to suit their preferences. These interactive features ensure that the extension is not just a passive monitoring tool but an active assistant in managing and understanding web security.

Side Panel/Popup Toggling: Traffic Analyzer offers two distinct modes of display—a compact popup and an expansive side panel to cater to different user needs and scenarios. Toggling between these modes is managed through a combination of user interactions and background script commands:

- **User Initiated Toggling:** When a user clicks the toggle button or the extension icon, a message is sent to the background script to switch the display mode.
- **Background Script Handling:** Depending on the current state, the background script either launches the side panel using the chrome.sidePanel API or reverts back to the standard popup view.
- **UI Synchronization:** Both the popup and side panel scripts include logic to adapt their layout and features according to the selected mode, ensuring a consistent user experience regardless of the display format. This flexible toggling capability allows users to access detailed analysis in the side panel when needed or opt for a more streamlined view via the popup for quick checks.

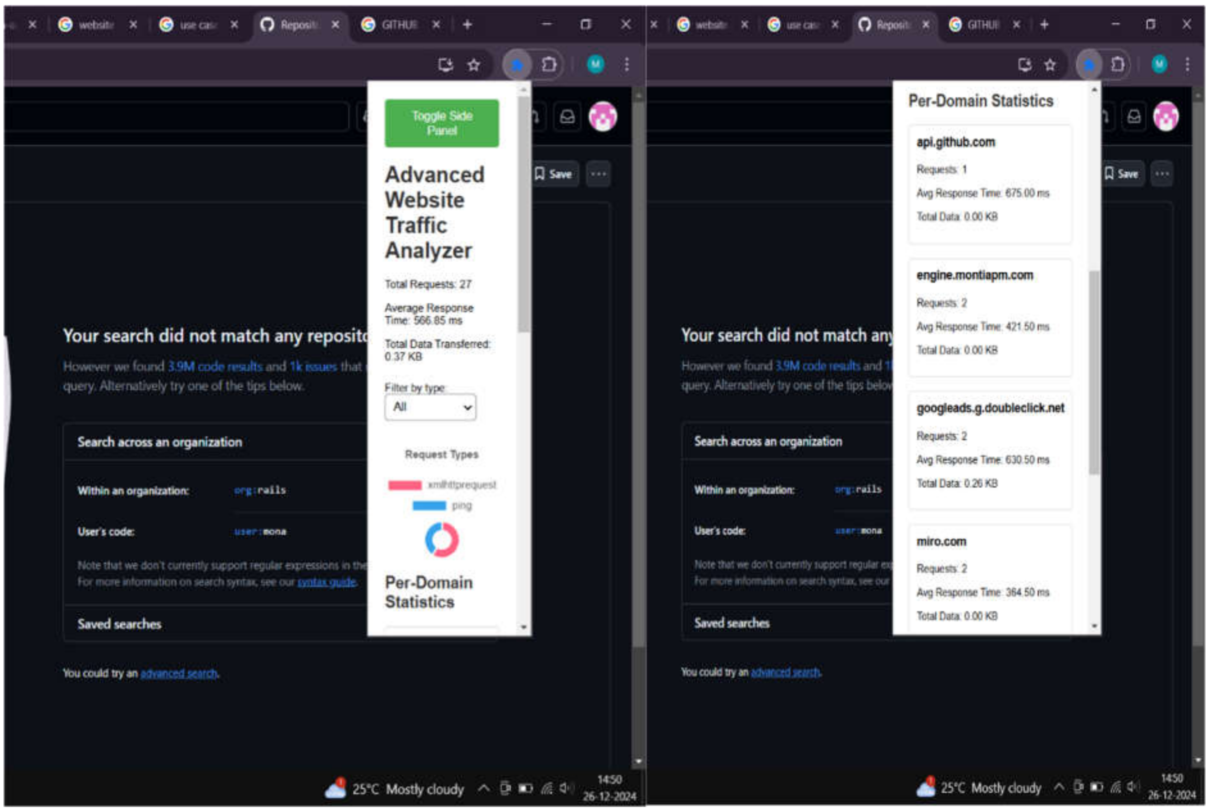
IV. RESULTS

Popup UI (popup.html): This is the primary output initially. Clicking the extension's icon opens a small popup window within the browser. This popup contains:

- Overall Statistics: Displays the total number of network requests made by the active tab, the average response time of these requests, and the total data transferred (in KB). These values update dynamically as new requests are made.
- Request Type Chart: A doughnut chart, generated using Chart.js, visually represents the distribution of request types (e.g., document, script, image, xhr). This chart also updates dynamically.
- Domain-Specific Statistics: Shows a breakdown of requests by domain, including the request count, average response time, and total data transferred for each domain accessed by the active tab.

- Cookie Information (Optional): If the user clicks "View Cookies," the popup will display information about the cookies associated with the active tab's URL. This includes details like the cookie name, value, domain, expiration date, and security flags (Secure, HttpOnly).
- "Clear Stats" Button: Allows the user to clear the collected statistics.
- Request Type Filter: A dropdown menu lets the user filter the displayed statistics and chart by request type (e.g., show only image requests).
- "Toggle Panel" Button: Allows the user to switch to a side panel view.

Side Panel UI (sidepanel.html): This is an alternative output view. Clicking the "Toggle Panel" button in the popup (or clicking the extension icon if the side panel is active) toggles to a persistent side panel within the browser window. The side panel contains the same information as the popup (statistics, chart, domain breakdown, cookie info) but in a more permanent layout that doesn't close automatically. It also provides theme switching functionality for light/dark/auto themes.



Advanced Website Traffic Analyzer

Total Requests: 27

Average Response Time: 566.85 ms

Total Data Transferred: 0.37 KB

Filter by type: All

Per-Domain Statistics

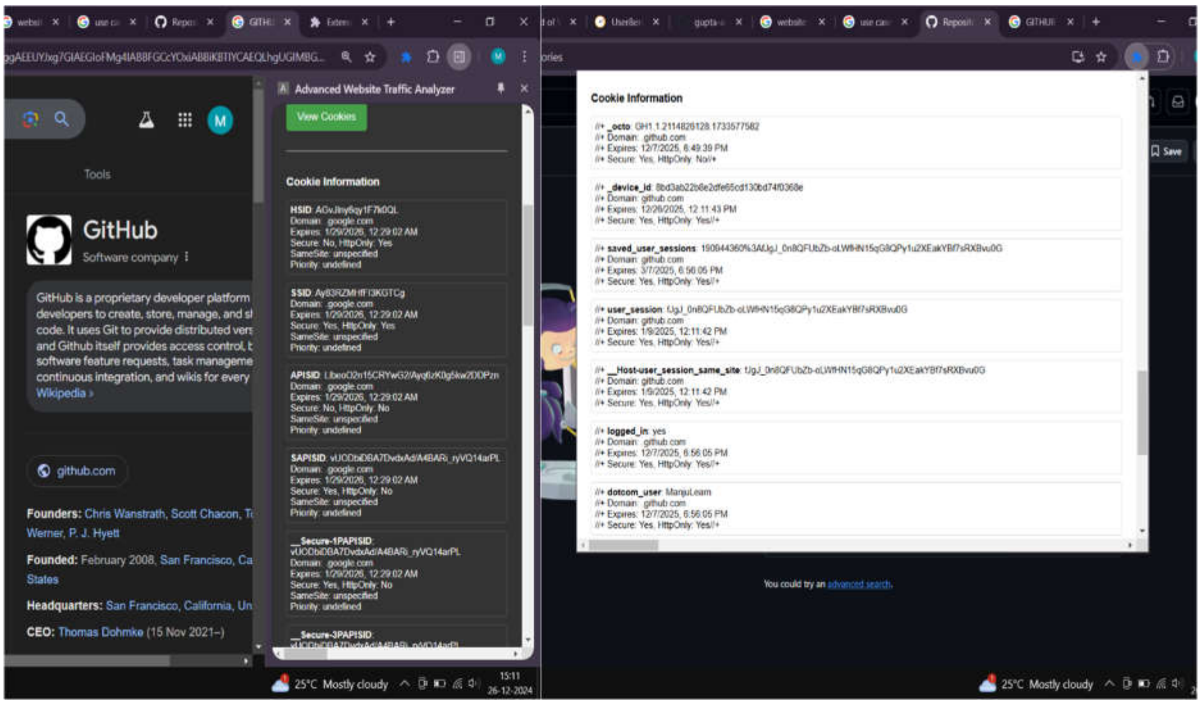
api.github.com
Requests: 1
Avg Response Time: 675.00 ms
Total Data: 0.00 KB

engine.montiapm.com
Requests: 2
Avg Response Time: 421.50 ms
Total Data: 0.00 KB

googleads.g.doubleclick.net
Requests: 2
Avg Response Time: 630.50 ms
Total Data: 0.26 KB

miro.com
Requests: 2
Avg Response Time: 364.50 ms
Total Data: 0.00 KB

Fig. 2 Popup window



```
Advanced Website Traffic Analyzer
View Cookies

Cookie Information

HSID: AGvJlny6qy1F7k0QL
Domain: .google.com
Expires: 1/29/2026, 12:29:02 AM
Secure: No, HttpOnly: Yes
SameSite: unspecified
Priority: undefined

SSID: Ay83RZMHfF I3KGTCg
Domain: .google.com
Expires: 1/29/2026, 12:29:02 AM
Secure: Yes, HttpOnly: Yes
SameSite: unspecified
Priority: undefined

APISID: LlbeoO2n15CRYwG2/Ayq6zK0g6kw2DDPzn
Domain: .google.com
Expires: 1/29/2026, 12:29:02 AM
Secure: No, HttpOnly: No
SameSite: unspecified
Priority: undefined

SAPISID: vUODbiDBA7DvdxAd/A4BARi_ryVQ14arPL
Domain: .google.com
Expires: 1/29/2026, 12:29:02 AM
Secure: Yes, HttpOnly: No
SameSite: unspecified
Priority: undefined

__Secure-1PAPISID:
vUODbiDBA7DvdxAd/A4BARi_ryVQ14arPL
Domain: .google.com
Expires: 1/29/2026, 12:29:02 AM
Secure: Yes, HttpOnly: No
SameSite: unspecified
Priority: undefined

__Secure-3PAPISID:
```

```
Cookie Information

//+ _octo: GH1.1.2114826128.1733577582
//+ Domain: .github.com
//+ Expires: 12/7/2025, 6:49:39 PM
//+ Secure: Yes, HttpOnly: No//+

//+ _device_id: 8bd3ab22b6e2dfe65cd130bd74f0368e
//+ Domain: github.com
//+ Expires: 12/26/2025, 12:11:43 PM
//+ Secure: Yes, HttpOnly: Yes//+

//+ saved_user_sessions: 190944360%3AfJgJ_0n8QFUbZb-oLWfHN15qG8QPy1u2XEakYBf7sRXBvu0G
//+ Domain: github.com
//+ Expires: 3/7/2025, 6:56:05 PM
//+ Secure: Yes, HttpOnly: Yes//+

//+ user_session: fJgJ_0n8QFUbZb-oLWfHN15qG8QPy1u2XEakYBf7sRXBvu0G
//+ Domain: github.com
//+ Expires: 1/9/2025, 12:11:42 PM
//+ Secure: Yes, HttpOnly: Yes//+

//+ __Host-user_session_same_site: fJgJ_0n8QFUbZb-oLWfHN15qG8QPy1u2XEakYBf7sRXBvu0G
//+ Domain: github.com
//+ Expires: 1/9/2025, 12:11:42 PM
//+ Secure: Yes, HttpOnly: Yes//+

//+ logged_in: yes
//+ Domain: .github.com
//+ Expires: 12/7/2025, 6:56:05 PM
//+ Secure: Yes, HttpOnly: Yes//+

//+ dotcom_user: ManjuLearn
//+ Domain: .github.com
//+ Expires: 12/7/2025, 6:56:05 PM
//+ Secure: Yes, HttpOnly: Yes//+
```

Fig. 3 User Interface

V. CONCLUSION

The Traffic Analyzer Chrome extension has proven to be a robust tool that empowers users to gain deep insights into their web browsing activity. The extension successfully intercepts and processes network requests in real time, providing users with a clear, visual summary of their online interactions. Users can view comprehensive statistics such as total requests, average response times, and total data transferred, all displayed through intuitive visual elements like a doughnut chart for request type distribution and detailed domain-specific metrics. With the ability to filter data, clear historical statistics, and even examine cookie details for the active tab, the extension offers a powerful yet user-friendly solution for monitoring network activity and potential security issues.

FUTURE SCOPE

Threat Intelligence Integration: Connect with external threat intelligence services and APIs to provide real-time updates on known malicious domains or security vulnerabilities. Such integrations would add an extra layer of protection by alerting users to emerging threats as soon as they are identified.Enhanced User Interface Improvements: Further refine the UI with more interactive elements and customizable views. Future updates could include advanced charting options, draggable and resizable dashboard widgets, and expandable sections for detailed metrics, making the data even more accessible and actionable.Caching Analysis: Extend the tool to differentiate between cached and network-fetched requests. This feature would help users identify potential caching issues and optimize their web performance by understanding how cached content is managed.

REFERENCES

1. Dr.Pampapathi B M, Archana BK , Apoorva K and Ashwini K " IOT Pet-Feeder With Home Security Robot" in Journal of Xidian University - May 2024 -VOLUME 18, ISSUE 5, https://doi.org/10.5281/Zenodo.11180790 - ISSN No:1001-2400.
2. Dr.Pampapathi B M, Arya R Kulkarni, M B Preetham and Harish B S "Detecting Suspicious Activities in Exam Hall to Prevent Cheating" in Solovyov Studies - May 2024-VOLUME 72, ISSUE 5 , ISSN : 2076-9210 https://drive.google.com/file/d/1IPJ1m6JBFn0BH8ndO17ncNaEJ-Tbd2s5/view
3. Pampapathi B M, A Madhuri, Chennareddy Nikhil, Amar Gouda Patil “Water Monitoring And Purification Of Waste Water For Agriculture Using Iot” in Journal For Basic Sciences Volume 23, Issue 4, 2023 , https://doi.org/10.37896/JBSV23.4/2050.
4. Pampapathi B M, Mohammad Moshin P , Mohammed Kareemuddin Saqlain, Prajwal Marthur, K Md Ibrahim hussain **“**Wireless Fire Detection Systems Using Iot” in NOVYI MIR Research Journal , Volume 8 Issue 4 2023 https://doi.org/16.10098.NMRJ.2022.V8I4.256342.37538
5. Pampapathi B M , Shruthi S M,” Detection and Classification of Phishing Websites Using Machine Learning” , in Journal Of Technology - Aug 2023 - Issn No:1012-3407 , Vol 13, Issue 8, D.O.I- https:// 10.61350/v13-105368 .
6. Pampapathi B M , Nageswara Guptha M , M S Hema ,” Towards an effective deep learning-based intrusion detection system in the internet of things” , in Telematics and Informatics Reports Journal- May 2022 , https://doi.org/10.1016/j.teler.2022.100009. Volume 7, September 2022, 100009.
7. Pampapathi, B.M., Nageswara Guptha, M. & Hema, M.S. Data distribution and secure data transmission using IANFIS and MECC in IoT. *J Ambient Intell Human Comput* **13**, 1471–1484 (2022). https://doi.org/10.1007/s12652-020-02792-4.
8. Pampapathi B M , Nageswara Guptha M , M S Hema , “Malicious Node Detection and Energy-aware Optimal Routing in Wireless Sensor Networks using CD-LVQ and BMSSO Algorithms” in The Journal of Huazhong University of Science and Technology ,Volume 50 , Issue 03.- March 2021- http://hustjournal.com/vol50mar-2/.
9. Pampapathi B M , Nageswara Guptha M , M S Hema , “Energy Efficient Data Distribution on Cloud With Optimal Routing Path Based Congestion Control in WSN Environment” in Journal of University of Shanghai for Science and Technology(JUSST), Volume 23, Issue 8, August 2021, https://doi.org/10.51201/JUSST/21/08409.
10. Pampapathi B M , Chandana Murthy , Supritha Kumar , Pooja M , Supriya K “Survey on IOT Based Medical Box for Elderly People” in International Journal of Advanced

Trends in Computer Science and Engineering (IJATCSE) ISSN 2278-3091, Vol.10 No.3 (May – June 2021 issue), https://doi.org/10.30534/ijatcse/2021/531032021.

11. Wei, M., Mink, J., Eiger, Y., Kohno, T., Redmiles, E. M., & Roesner, F. (2024). SoK (or SoLK?): On the Quantitative Study of Sociodemographic Factors and Computer Security Behaviors. arXiv preprint arXiv:2404.10187. Link: https://arxiv.org/abs/2404.10187
12. Google Inc. (2023). Chrome Extension Documentation. Retrieved from https://developer.chrome.com/docs/extensions/
13. Ishaq, K., & Fareed, S. (2023). Mitigation Techniques for Cyber Attacks: A Systematic Mapping Study. arXiv preprint arXiv:2308.13587. Link:https://arxiv.org/abs/2308.13587
14. Shahid, M. (2023). Machine Learning for Detection and Mitigation of Web Vulnerabilities and Web Attacks. arXiv preprint arXiv:2304.14451. Link: https://arxiv.org/abs/2304.14451
15. Kim, S., & Lee, S. H. (2022). A novel defense against cross-site scripting using machine learning. Journal of Cybersecurity, 8(3), 1-17. DOI: 10.1093/cybsec/tyab029
16. Chatzikonstantinou, A., & Dritsas, S. (2020). Phishing attack detection via browser-based heuristics. Computers & Security, 92, 101760. DOI: 10.1016/j.cose.2020.101760
17. Carlini, N., Erlingsson, Ú., Feldman, M., & Koskinen, E. (2019). Secure extensions: Formal security and privacy analysis of browser extensions. USENIX Security Symposium, 1387-1404. Link: https://www.usenix.org/conference/usenixsecurity19/presentation/carlini
18. Jiang, X., Zheng, X., & Li, H. (2018). Security analysis of browser extensions: Threats, techniques, and countermeasures. IEEE Transactions on Dependable and Secure Computing, 15(2), 329–342.
19. Weinberger, D., Roy, S., & Marczak, B. (2017). Security and Privacy in Browser Extensions: A Comprehensive Review. ACM Computing Surveys (CSUR), 50(3), 1–36.DOI: 10.1145/3073021
20. Böttinger, K., Berger, C., & Kirchner, D. (2017). Enhancing web security with subresource integrity. In Proceedings of the IEEE Symposium on Security and Privacy (pp. 401–416). IEEE.
21. Stoll, C., Rütti, A., & Weiler, S. (2017). Enhancing IT-security dashboards with actionable insights: A case study. Proceedings of the 2017 International Conference on Big Data and Advanced Wireless Technologies (BDAW). DOI: 10.1145/3164761.3164770
22. Wu, M. (2016). Analyzing and mitigating the security risks of browser extensions. Journal of Computer Security, 24(3), 345–367.
23. Gupta, B. B., Agrawal, D. P., & Yamaguchi, S. (2016). Phishing and malware attack detection using machine learning. Cyber Security in Parallel and Distributed Computing, 151-168. DOI: 10.1002/9781119225992.ch9
24. Nappa, A., Rafique, M. Z., & Caballero, J. (2015). Driving in the cloud: An analysis of drive-by download operations and abuse reporting. Proceedings of the 10th ACM

SIGSAC Conference on Computer and Communications Security (CCS), 119-130.DOI: 10.1145/2660267.2660310

25. Johnston, J., Warkentin, M., & Siponen, M. (2015). An enhanced fear appeal rhetorical framework: Leveraging threats to the human asset through sanctioning rhetoric. MIS Quarterly, 39(1), 113-134. DOI: 10.25300/MISQ/2015/39.1.05
26. Carlini, N., & Wagner, D. (2014). ROP is Still Dangerous: Breaking Modern Defenses.USENIX Security Symposium, 385–399. Link: https://www.usenix.org/conference/usenixsecurity14/technical-sessions/presentation/carlini
27. Sood, A. K., & Enbody, R. J. (2013). Targeted cyberattacks: A superset of advanced persistent threats. IEEE Security & Privacy, 11(1), 54-61. DOI: 10.1109/MSP.2013.8
28. Eshete, B., Villafiorita, A., & Weldemariam, K. (2013). BINSPECT: Holistic binary inspection for vulnerabilities. Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS), 1337–1348. DOI: 10.1145/2508859.2516753
29. Bonneau, J., Herley, C., Van Oorschot, P. C., &Stajano, F. (2012). The quest to replace passwords: A framework for comparative evaluation of web authentication schemes. In Proceedings of the IEEE Symposium on Security and Privacy (pp. 553–567).
30. Heer, J., & Shneiderman, B. (2012). Interactive dynamics for visual analysis. Communications of the ACM, 55(4), 45-54. DOI: 10.1145/2133806.2133821
31. Stuttard, D., & Pinto, M. (2011). The Web Application Hacker's Handbook: Finding and Exploiting Security Flaws. Wiley.
32. Bravo-Lillo, C., Cranor, L. F., Downs, J., & Komanduri, S. (2011). Bridging the gap in computer security warnings: A mental model approach. IEEE Security & Privacy, 9(2), 18-26. DOI: 10.1109/MSP.2010.197
33. Wash, R. (2010). Folk models of home computer security. Proceedings of the 6th Symposium on Usable Privacy and Security (SOUPS), 1-16. DOI: 10.1145/1837110.1837125
34. Few, S. (2009). Now You See It: Simple Visualization Techniques for Quantitative Analysis. Analytics Press.
35. Moore, T., Clayton, R., & Anderson, R. (2009). The economics of online crime. The Journal of Economic Perspectives, 23(3), 3-20. DOI: 10.1257/jep.23.3.3
36. Barth, A., Jackson, C., & Mitchell, J. C. (2008). Securing frame communication in browsers. In Proceedings of the ACM Conference on Computer and Communications Security (CCS) (pp. 1–10). ACM.
37. Cranor, L. F. (2008). A framework for reasoning about the human in the loop. In The Economics of Information Security and Privacy (pp. 19–35). Springer.
38. Keim, D. A., Mansmann, F., Schneidewind, J., Thomas, J., & Ziegler, H. (2008). Visual analytics: Scope and challenges. Visual Data Mining, 76-90. DOI: 10.1007/978-3-540-71080-6_6
39. Kirda, E., Kruegel, C., Vigna, G., & Jovanovic, N. (2006). Noxes: A client-side solution for mitigating cross-site scripting attacks. Proceedings of the ACM Symposium on Applied Computing (SAC), 330-337. DOI: 10.1145/1141277.1141358

40. Garfinkel, S. L., & Rosenblum, M. (2003). When good software goes bad: Security issues for mobile code and client-side applications. Communications of the ACM, 46(9), 45-50. DOI: 10.1145/944217.944243
41. Rescorla, E. (2001). SSL and TLS: Designing and Building Secure Systems. Addison-Wesley.