JAAM: Web Browser Security Framework

[*]Note: This is a draft paper, it is in early phase and is not completed yet.

Deepak Singh Tomar
deepaktomar@manit.ac.in

Jishan Shaikh
jishanshaikh9893#

Ankit Chouhan
ankitchoudhan.dws97#

Ankit Chaudhary
ankitchaudhary9838#

Mansoor Sediqi
sedeqym#

***Abstract*—This paper proposes a novel framework for web browser security for technical and non-technical organizations concerning web browser development and usage. It will help organizations assess, evaluate, formulate, and implement a strategy for their web browser security concerns. It is quite adaptive to large scale organizations, also flexible to small and medium scale organizations. Major features of the framework includes easy customization, neutrality towards life-cycle models, open-source nature, and standardization of security practices. Component division of framework makes it more usable and reliable. Core framework is based on four major components: Design sub-components, Tools and Applications, Code and Utilities, and Miscellaneous sub-components. Major design sub-components are Web Browser Security Assurance and Maturity Model (WBSAMM), Security Architecture. WBSAMM is based on modern security practices, business functions, and multiple maturity levels. Quantitative dependency of other parameters is also analyzed. Code utilities includes sanitizer sub-component for web browser languages such as HTML, JavaScript, XML, Python. Issues of coding standards are also proposed under Code and Utilities component.**

A malicious URL detection tool is also developed as a part of tools and application component using data mining of 420K+ URLs giving an accuracy rate of 99.3%. Miscellaneous component includes review and analysis of various security aspects e.g. Browser forensics, Test suites (Static, Dynamic, and Run-time), Testing guides, Risk assessment methods, Threat modeling, Social and Reverse Engineering, Proxy and VPN, Attack tensors, Flash and JavaScript, Global Vulnerability Repository and Vulnerability Query Language. Comparison of modern web browsers (Google Chromium, Mozilla Firefox, Tor) is presented on basis of abstract features, standards, and protocols.

***Index Terms*—Web Browsers, Security, Framework, JAAM**

I. Introduction

II. Related Work and Research Gap

Open Web Application Security Project (OWASP) has presented Software Assurance Maturity Model (SAMM) for general software based on 3 maturity levels. Software Engineering Institute has proposed Software Capability and Maturity Model based on 5 maturity levels. JAAM is web browser specialization of OWASP's SAMM in some aspects. Related work on Browser security presented as white paper by [2] and [3]. Gaps in providing secure browser and research in data breaches is presented in [4]. A rich encyclopedia of XSS attacks is [5]. Various methods of gaining/revoking control, bypassing same origin policy, and attacking through various vectors are presented in [6]. [6] is considered as standard reference for web browser hackers.

III. Components and Working of a web browser

A web browser is a complex application software stack that works as an interface between server and client with utmost security. It helps in accessing information available on world wide web using URL (Uniform Resource Locator). Most common web browsers used today are Chrome, Safari, Firefox, UC, Samsung Internet, Opera [1].

A. Components of a web browser

A common browser usually have utility components including user interface, rendering and browsing engine, UI back-end, JavaScript interpreter, and a database. These components constitute the web browser architecture as shown in Fig. 1.

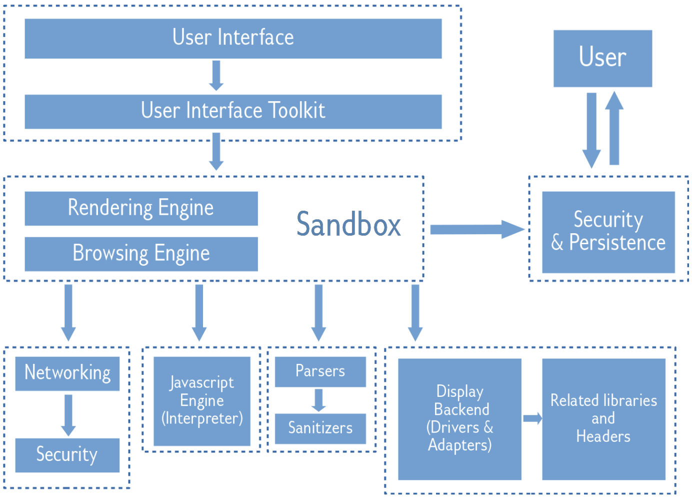

Fig. 1. Architecture of a modern web browser

User Interface: This is the interface with which client communicates with server. This typically comprises of web-pages, web-applications, or simple documents/media according to its usage and application. Usually web-pages are developed with the help of HTML (Hyper Text Markup Language), CSS (Cascading Style Sheets), JavaScript, or other frameworks such

NOTE: This paper is an output of minor project based on Browser Security by authors under supervision of Dr. Deepak Singh Tomar. **# indicates @gmail.com**

as Django, AJAX (It is special related to browser security because of its fundamental *XMLHttpRequest* object), React, and many more. It is usually graphic rich.

Browsing engine: This component acts as connecting component of User interface of client and rendering engine. Rendering engine sends rendered data to browsing engine, it simply formats it to visually good looking and well organized format.

Rendering engine: It renders the JavaScript data, along with processing of CSS file included in user interface. It is also connected with Networking components i.e. all the networking activities seen inside the user interface window passes through rendering engine before, for processing or specifically rendering, hence the name.

Networking: Web browser is a social application, it means it is connected to world wide web using URL (Universal Resource Locator) through internet with the help of networking.

User interface back-end: It comprises of various back-end files written with either PHP, SQL (or any other alternatives of it). It usually have codes for specific purpose for implementation in user interface.

JavaScript virtual machine or JavaScript interpreter: This is the component where JavaScript scripts are interpreted, and are bind-ed with byte code to execute on browser engine. There are multiple flavors of JavaScript available in the industry, each of which having its own advantages and disadvantages.

Data Storage or Database: Each browser has its own data storage/database usually not accessible by end-user / client. It has its preferential applications such as buffer storage, file storage (from server), cookie storage, History and Bookmarks storage, Settings preferences, etc.

B. Working of a web browser

The main work of a web browser is as an interface between client (end-user) and server. But, its not that much easy. It is pretty well concerned with DNS server with website address to IP (Internet Provider) address translation, and fetching data from server with utmost security and encryption. The working cycle is explained in Fig. 2.

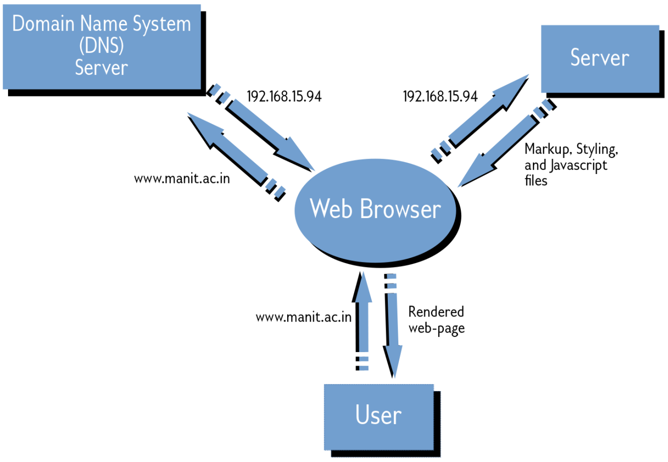

Fig. 2. Working of a web browser

IV. JAAM: Web Browser Security Framework

A. Design components

Web Browser Security Assurance Maturity Model (WB-SAMM): The core framework of WBSAMM is based on business functions, security practices, and maturity levels.

There are 5 business functions proposed. *[Explain each of them]*

1) Planning, Availability and Research
2) Technical Management
3) Development and Construction
4) Evaluation, Validation, and Assessment
5) Maintenance Management

And the proposed 12 security practices are as follows. *[Explain each of them]*

1) Security Strategy
2) Ethics and Policies
3) Training and Evaluations
4) Threat Modeling
5) Privacy requirements
6) Technical Review
7) Secure architecture
8) Management Review
9) Performance metrics and comparison parameters
10) Testing with security considerations
11) Version control management
12) Maintenance best practices

Web Browser Security Architecture based on WB-SAMM: Proper mapping of security practices to business functions to an appropriate maturity level constitute the major architecture for security. *[Write more]*. See Fig. 3.

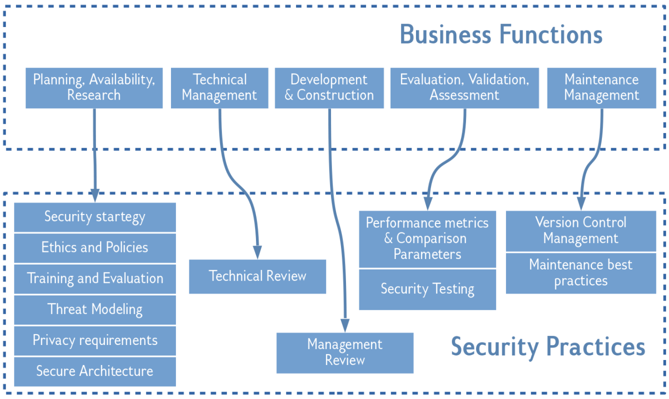

Fig. 3. Proposed architecture of Web Browser Security Assurance Maturity Model

B. Tools and Applications components

Malicious URL detection tool: *[Write more]*.

Methodology for development of Malicious URL detection tool: *[Write more]*. See Fig. 4.

Psuedo-code for Malicious URL detection tool: *[Write more].*

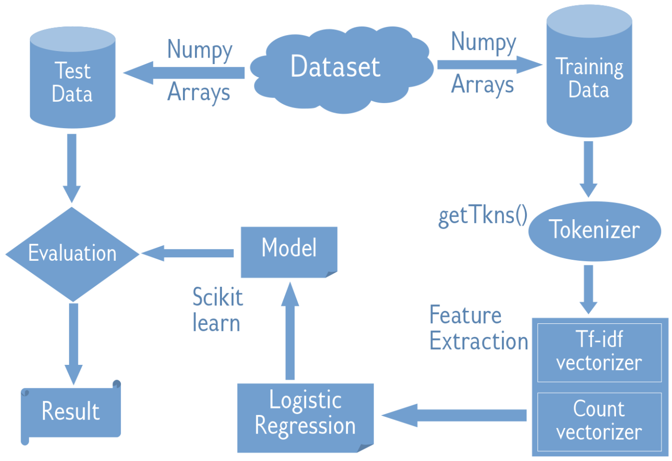

Fig. 4. Proposed method for malicious URL detection tool development

C. Code and Utilities components

Code Sanitizers: *[Write more].*
Sandboxing: *[Write more].*
Fuzzing: *[Write more].*
Electrolysis: *[Write more].*

D. Miscellaneous components

Testing guides *[Write more].*
Test Suites (Static, Dynamic, and Run-time) *[Write more].*
Browser Forensics *[Write more].*
Memory Analysis *[Write more].*
Threat Modeling *[Write more].*
Risk Assessment Methods *[Write more].*
Social Attacks *[Write more].*
Reverse Engineering *[Write more].*
Proxy and VPN *[Write more].*
Attack Tensors *[Write more].*
Flash and JavaScript *[Write more].*
Global Vulnerability Repository *[Write more].*
Vulnerability Query Language (VQL) *[Write more].*

V. Conclusion and Future Work

The component nature of framework allows inclusion of new sub-components to framework in either of four major components.

VI. Screenshots

Acknowledgment

Authors are thankful to their supervisor Dr. Deepak Singh Tomar, Reviewers and Project co-ordinators Dr. Sanyam Shukla and Dr. Rajesh Wadhwani, Head of the department, CSE MANIT Bhopal Dr. Meenu Chawla, Director of MANIT Bhopal Dr. Narendra Raghuwanshi. Authors also appreciates all persons/organizations who contributed directly/indirectly to this project. This project is not funded by any agency/organization.

About the Project

Write about minor project. Supervisors and reviewers.

References

[1] Browser Market Share Worldwide - March 2019. www.statcounter.com
[2] M. Vervier, M. Orrù, B. J. Wever, E. Sesterhenn, "Browser Security White Paper," X41 D-SEC GmbH, Dennewartstr. 25-27 D-52068 Aachen Amtsgericht Aachen: HRB19989, September 2017.
[3] I. M. Heiderich, A. Inführ, F. Fabler, N. Krein, M. Kinugawa, T. C. Hong, D. Weiber, P. Pustulka, "Cure53 Browser Security White Paper," Bielefelder Str. 14, D 10709, Berlin, cure53.de, November 2017.
[4] S. Donaldson, A. Florescu, K. Roemer, M. Zugec. Secure Browsing, Citrix XenApp, Citrix XenServer, Direct Inspect APIs, Bitdefender HVI. Joint white paper, 2016.
[5] J. Grossman, R. Hansen, P. D. Petkov, A. Rager, S. Fogie, "XSS Attacks: Cross Site Scripting Exploits and Defense," Syngress publishing, USA, 2007, ISBN-13: 978-1-59749-154-9.
[6] W. Alcorn, C. Frichot, M. Orrù, "The Browser Hacker's Handbook", Wiley publishing, USA, ISBN: 978-1-118-66209-0 ISBN: 978-1-118-66210-6 (ebk), 2014.
[7] A. Barth, C. Jackson, C. Reis, Google Chrome Team, "The Security Architecture of the Chromium Browser".