International Journal of Engineering & Technology, *7 (4.44) (2018) 242-246*

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

Web Browser Vulnerabilities and Weakness Descriptive Analysis: Is it Chrome Keep Dominant?

Abdullah Fajar[1*], Setiadi Yazid[2*]

[1,2]*Faculty of Computer Science,Universitas Indonesia*
[1]*Faculty Of Engineering, Universitas Widyatama*
**Corresponding author E-mail: setiadi@cs.ui.ac.id;*

Abstract

Web Browser play the important mandatory role in accessing the application through the internet and may carry malicious content to the system hence threatening the system from the attacker. Google Chrome is one of popular browser since released on 2008 as one of product of Chromium Project at Google. Chrome is fourth ranking in Common Vulnerabilities Enumeration website and the first ranking among browser that have most of vulnerabilities reported. This paper describe a Descriptive analysis of weakness and vulnerabilities of Chrome browser. The analysis use comparison approach to other popular browser such as Safari and Firefox. The analysis also use main reference and database from mitre.org which have common weakness enumeration database and scoring system calculation for vulnerability. This work cover responsiveness rate among them regarding weakness and vulnerabilities update duration and severity rate. The validation has performed using Descriptive test regarding weakness and vulnerability behavior. According to Architectural, Development and Research Conceptual weakness reported, the browsers has not significantly indicate the difference except between Chrome and Firefox in research conceptual weakness. The severity of browser vulnerabilities shown by Firefox and the best responsiveness to update browser weakness shown by Chrome, followed by Safari. Using Descriptive analysis, Chrome will keep dominant against the other browser, while Firefox and Safari potentially become unpopular such as Internet Explorer for upcoming time.

Keywords: *Chrome; Vulnerability; Weakness; Attack; Security*

1. Introduction

The most interaction between internet and computer user through World Wide Web. Users can browse the content of internet with a diverse landscape of services and applications. Now, the services and application through the web must keep their user safe from various malicious content. The main target in this challenge mostly occurred on web browsers. Insecure/malicious content sender (attacker) point at Web Browsers because there is a lot of compromising to keep the risk content able to be delivered through it. Reis et.al [1] stated, there three factor must be considered by browsers vendor to keep safe, there are: (1) The Severity of Vulnerability; (2) The windows of Vulnerability; (3) The Frequency of exposure. If these consideration are followed out by mitigation action then the security will improve, multiply benefit and assist user to keep users safe.

Securing browser is the first step that need to be taken in order to assure secure online protection. The threats tend to increase by taking advantage of web browser vulnerability through vulnerable webs application. This condition become worse according to several factor such as:

1. Many computer users are not aware of the click on the web links.
2. Software and third party software packages installed combined increases the number of vulnerabilities
3. Many websites require that users enable features or install more software, third- party software which doesn't get security updates putting the computer at additional risk.
4. Many users do not know how to configure their web browsers securely

If that website design allowing to host malicious code, then some vulnerabilities in a particular browser enabling this malicious code to run processes within the browser application in unintended ways. Web Browser vulnerability can be minimized by means of updating process regularly [2]. That prevention will not affective if the underlying system has already compromised. Some parts of browsers such as scripting, add-ons, and cookies are particularly vulnerable and need to be considered for taking action further.

Google Chrome as one of popular web browser is released at late 2008. This browser is intended to be fast and secure one. Chrome treat each web page as individual process and make it as a sandboxing to confirm the safety. Chrome uses a model to allocate a process for sandboxing tabs. Chrome retrieves updates regularly of two issues (phishing and malware), and also Google warns users if they attempt to visit a potentially harmful which has already warning flag. This service is also made available for use by others via a free public API called "Google Safe Browsing API".

This paper describe an analysis of weakness and vulnerabilities enumeration for Chrome Browser as case study. This work use Descriptive data analysis approach from CVE database (cvedetails.com) to describe vulnerability and weakness occurrences behavior. This work choose Chrome as target to be analyzed and will compare the characteristics to other browser such as Safari,

and Firefox. The comparison using report from 2008 to 2018 for all browser to make sure the analysis has a fair means.
The rest of paper organize as follow, second section describe about Conceptual Review that contain explanation of Common Vulnerability Scoring System, Common Weakness Enumeration, Security Architecture and Browser Security Issue. The third section describe Descriptive Analysis that describe Weakness and Vulnerability Severity and Responsiveness. The fourth section describe a discussion about analysis and potential recommendation for further research. The final section is a conclusion that describe about important point about Weakness and Vulnerability enumeration and future direction of research area of browser security analysis and practice.

2. Conceptual Review

2.1. Common Vulnerability Scoring System

The Common Vulnerability Scoring System (CVSS) is an open framework that addresses this issue[3]. CVSS is composed of three metric groups: Base, Temporal, and Environmental, each consisting of a set of metrics. This System have several benefit, such as: (1) Standardized Vulnerability Scores; (2) Open Framework and (3) Prioritized Risk.

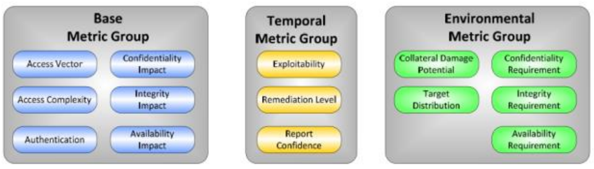


Figure 1 CVSS Metric Group

These metric groups are described as follows:

1. Base: represents the intrinsic and fundamental characteristics of a vulnerability that are constant over time and user environments.
2. Temporal: represents the characteristics of a vulnerability that change over time but not among user environments.
3. Environmental: represents the characteristics of a vulnerability that are relevant and unique to a particular user's environment.

2.2. Common Weakness Enumeration

CWE™ is a community-developed list of common software security weaknesses[4]. It is applied as a common language, a measurement bar for software security tools, and as a baseline for weakness identification, mitigation, and prevention efforts. The Common Weakness Enumeration (CWE™) is a list of software weaknesses types. Each common weakness type is listed by a community initiative to create specific and succinct definitions. The item in the list is hoped adequately described and differentiated by leveraging the widest possible group of interests and talents. Creating the CWE List is a living effort with ongoing work to capture the specific effects, behaviors, exploit mechanisms, and implementation details as well as to review and revise the presentation approaches to provide those that best suit the community using this information.
CWE have three specific points of view, there are: (1) The Research Concepts representation facilitates research into weakness types and organizes items by behaviors;(2) The Development Concepts representation organizes items by concepts that are frequently used or encountered during development;(3) Architectural Concepts.

2.3 Security Architecture

Previously most web browsers implement a monolithic architecture that combines two part of browser usage that are "the user" and "the web" into a single protection domain. The security architecture Google Chrome as part of Chromium project. Chrome has two modules i.e. a browser kernel and a rendering engine[5]. These modules is intended to protect domains separately, a browser kernel, will interacts with the operating system, and a rendering engine, will runs with restricted privileges in a sandbox. This architecture is built for balancing security, compatibility, and performance. The architecture allow an attacker who compromises the rendering engine from attacking other web sites (for example, by reading their cookies). Specifically the architecture is addressed to prevent an attacker from accessing the user's file system, assisting protection to the user from a drive by malware installation.

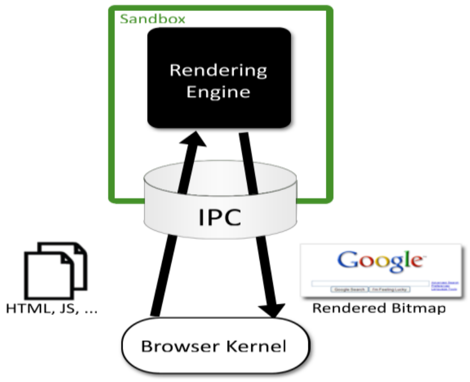
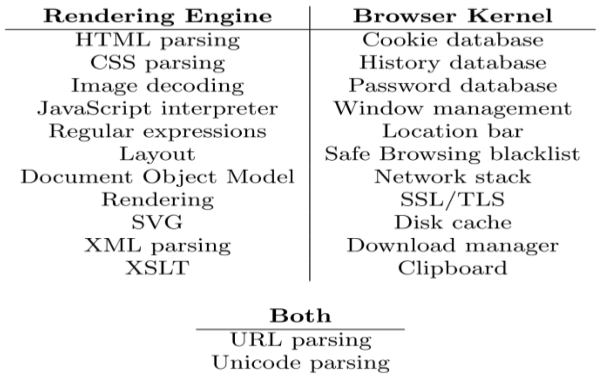


Figure 2 Browser Kernel and Rendering Engine Mechanism for parsing web content and emits bitmaps of the rendered document

Google Chrome uses a modular architecture that places the complex rendering engine in a low- privilege sandbox. Google Chrome has two major components that run in different operating-system processes: a high-privilege browser kernel and a low-privilege rendering engine[1]. The rendering engine interprets and executes web content by providing default behaviors (for example, drawing <input> elements) and by servicing calls to the DOM API and also responsible for enforcing the same-origin policy, which helps prevent malicious web sites from disrupting the user's session with honest web sites. The browser kernel is responsible for managing multiple instances of the rendering engine and for implementing the browser kernel API. The browser ker- nel manages persistent state, such as the user's book- marks, cookies, and saved passwords. It is also responsible for interacting with the network and inter- mediating between the rendering engine and the operating system's native window manager. The browser kernel uses this state to implement a security policy that constrains how a compromised rendering engine can interact with the user's operating system



Rendering Engine	Browser Kernel
HTML parsing	Cookie database
CSS parsing	History database
Image decoding	Password database
JavaScript interpreter	Window management
Regular expressions	Location bar
Layout	Safe Browsing blacklist
Document Object Model	Network stack
Rendering	SSL/TLS
SVG	Disk cache
XML parsing	Download manager
XSLT	Clipboard

Both
URL parsing
Unicode parsing

Figure 3 Roles of Rendering Engine and Browser Kernel

2.4 Browser Security Issues

There are several way that web browsers can be breached as conclude from [6], [7] and [8]such as: (1) In privilege Mode malware able to read or modify memory space of the browser and breach the operating system; (2) In Privilege Mode malware able to read or modify memory space of the browser and a malware running as a background process of an operating system; (3) Main browser executable can be hacked;(4) Browser components may be hacked;(5) Browser plugins can be hacked;(6) browser network communications could be intercepted outside the machine.

Vulnerable web browsers increase potential software attack. The observation to new software vulnerabilities are being exploited and addressed to web browsers by compromised or malicious websites. This problem increase and worse by a number of factors, as stated by Szabo[9] :

1. Many users have an unaware tendency when follow out web links without considering the risks of their actions.
2. An expected site or website address will be accessed.
3. Many web browsers function optimized but the security become vulnerable.
4. Some software potentially carry new security vulnerabilities when it's configured and packaged from the manufacturer.
5. Additional software package for some computer systems and software packages potentially increase the number of vulnerabilities.
6. Security update mechanism for software package may unavailable especially for third-party software.
7. Additional requirement to enable certain features or install more software may lead to additional risk and vulnerabilities..
8. Lack of knowledge to configure web browsers securely often found.
9. Many users disable a feature to secure their web browser.

Web browser vulnerabilities is a popular way to attack computer system. A vulnerability is a hole/gap or a weakness in the application. The weakness can be a design flaw or an implementation bug, that enables an application being attacked to cause harm to the stakeholders of an application. Stakeholders may consist of the application owner, application users, and other entities that rely on the application. The term "vulnerability" is often used very loosely. However, here we need to distinguish threats, attacks, and countermeasures also measurement to identify the potential risk and severity. Measurement database of vulnerabilities is published at cvedetails.com. There are thirteen classification of vulnerabilities type which used by CVSS[10], i.e: (1) Denial of Services;(2) Code of Execution;(3) Overflow;(4) Memory Corruption;(5) SQL Injection; (6) Cross Site Scripting (XSS);(7) Directory Traversal;(8) HTTP Response Splitting;(9) Bypass Something;(10) Gain Information;(11) Gain Privileges;(12) CRF ;and (13) File Inclusion.

4. Descriptive Analysis

3.1. Weakness and Vulnerbilities Severity and Responsiveness

Since 2008 when Chrome Released, there are 1545 occurrences, respectively 1481, 969 for Firefox and Safari. Compared to other browser such as Firefox and Safari, the distribution is shown below:

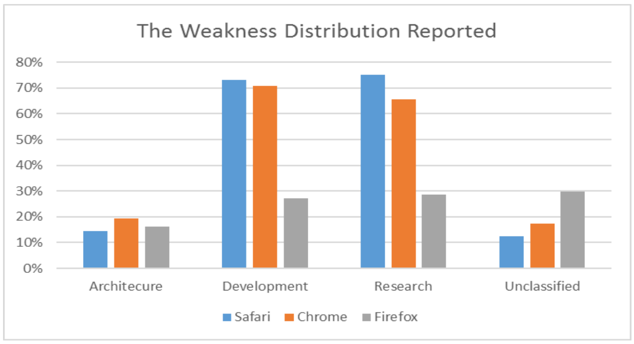


Figure 4 The Weakness Distribution Reported

Firefox shows the severity score for vulnerabilities higher than Chrome and Safari for average since released, respectively 8,31, 6.68 and 6,41. The percentage of severity highest score is 34% that achieved by Firefox, compared to Chrome only 14% and Safari 25%. The trend of severity score have no significantly difference mean for 11 year observation since 2008, but the variances significantly difference one browser to others browsers. Visually, Chrome has lower severity score compared to others browsers.

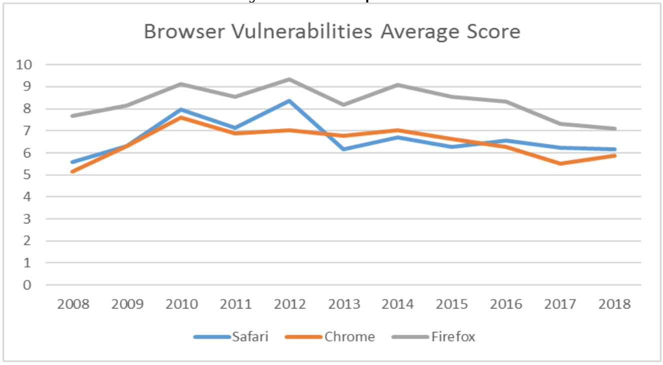


Figure 5 Vulnerabilities Average Score since 2008 to 2018

The responsiveness to update and close the weakness may relate to figure 5 above since Chrome also has good trend in update duration time when weakness reported. Although Safari visually have good trend to respond and update the weakness, there is significant variance between Safari and Chrome, here is the figure 6:

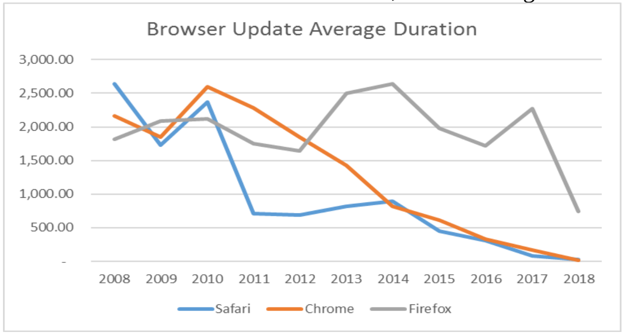


Figure 6 Browser Update Average Duration

3.2. Conceptual Weakness Analysis

MITRE defined three conceptual weakness in application security respectively Architectural, Development and Research. For further reading, please refer to the website. Architectural and development concept is cascading relation for software development practice. Many factors may influence how architectural concept will drive successfully in development practice. This section discuss from Descriptive calculation view to describe relation between them.

3.3.1 Architectural Concept Weakness

Descriptively, the occurrence of architectural concept weakness reported, has strong correlation with development concept weakness reported. The most weakness that occurred by the browsers are Improper Input Validation (CWE-ID 20), Improver Neutralization of Input During Web Page Generation ('Cross-site Scripting (CWE-ID 79), Improper Control of Generation of Code ('Code Injection') (CWE-ID 94) and Improper Access Control (CWE-ID 284).

About 10% to 12.5% of vulnerability score of the browsers is classified to highest level and Improver Input Validation dominate the architectural concept weakness of the browsers (up to 80%) but the severity level is up to medium. Chrome has to take attention of this vulnerability since the percentage of highest vulnerability score is 15%. Firefox and Safari must take serious attention to handle Improper Control of Generation of Code (“Code Injection”) since the percentage respectively 70% and 95% is classified to highest vulnerability score. The behavior of weakness occurrences tend to uniform and slight down for Chrome and Safari compared to Firefox (refer to Figure 7 and 8).

3.3.2 Development Concept Weakness

Related to Architectural Concept Weakness, the most occurrences of development concept weakness classified to Improper Restriction of Operations within the Bounds of a Memory Buffer (CWE-ID 119) and Resource Management Errors (CWE-ID 399). CWE-ID 119 relate to the browser performs operations on a memory buffer, but it can read from or write to a memory location that is outside of the intended boundary of the buffer.

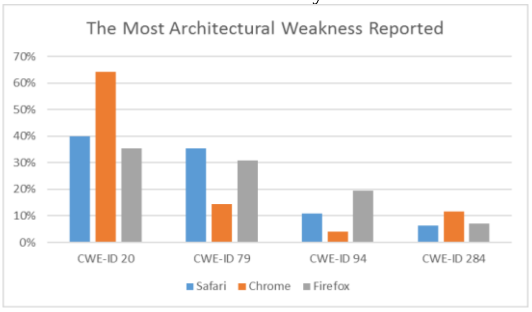


Figure 7 The Most and Trend Architectural Concept Weakness by CWE-ID

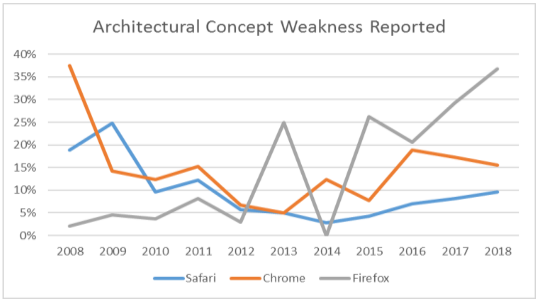


Figure 8. The Most and Trend of Architectural Concept Weakness

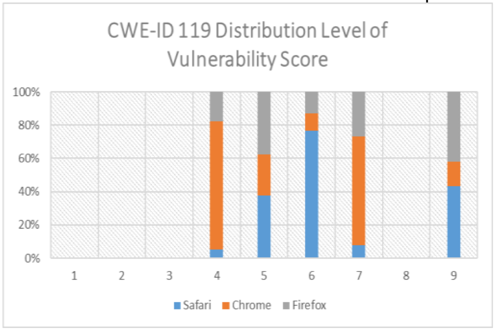


Figure 9 CWE ID 119 Distribution Level of Vulnerability Scores

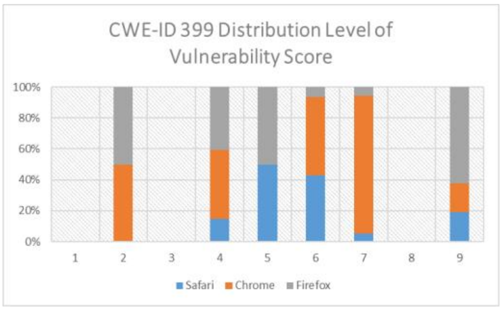


Figure 10 CWE ID 399 Distribution Level of Vulnerability Scores

As a result, an attacker may be able to execute arbitrary code, alter the intended control flow, read sensitive information, or cause the system to crash. Chrome performed better in memory management and slightly different compared to Safari, but Firefox has not good performance. More than a half of CWE-ID 119 of Firefox fall into highest level of vulnerability score. Chrome has the lowest occurrences of Resource Management Errors compared to other browsers. Typical condition of Firefox and Safari have about half of this vulnerability fall into highest level of vulnerability score.

Same condition to architectural concept weakness of the browsers, Chrome and Safari show more constant compared to Firefox on occurrences behavior. As stated above, the conclusion show that Chrome has better development concept, which is indicated to the lowest vulnerability score and weakness occurrences tend to uniform.

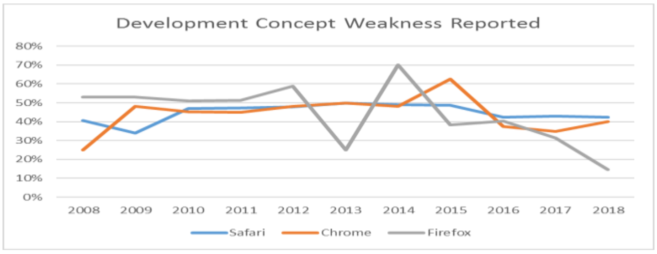


Figure 11 Development Concept Weakness Reported

5. Discussion

Satish and Chavan[11] has studied for securing web browser from attack and the paper describe a systematic study of how to make a browser secure. They conclude that without security patches, web browsers are vulnerable to different types of attack. A web browser is not totally secure because plug-ins are also vulnerable. Browser based attacks originate from malicious websites. The Attacker can easily deliver malicious code to user’s system. The user should block pop-up windows to avoid malicious code to be downloaded on user system. The browser is inherently insecure without multi process and exposes the user to different exploits. Multi process and OS level sandboxing must become standard and mandatory features and eventually each tab must be contained within a separate process. The main requirement to secure the browser mostly have fulfilled especially by Chrome. The adoption Chrome Browser engine by other browser product[12] indicate the superiority to other browser such as Firefox, Internet Explorer and Safari.

On other hand, Descriptively Chrome has the most report from vulnerability and weakness enumeration. This statistics description above show that the frequency of weakness reporting is not indicating the overall weakness. Conceptual Weakness both architectural and development indicate the superiority of Chrome to other browsers. The fact that Chrome have strongest support technically and resources may not be ignored. Market domination by Chrome for last 10 years may not avoided also and increase since Android released.

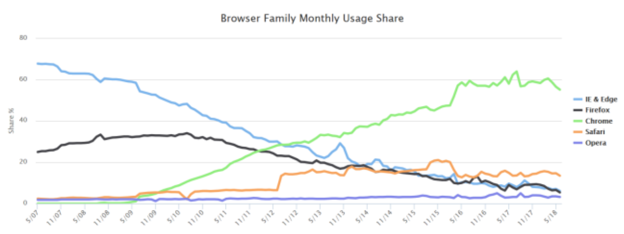


Figure 12 Browser Market Share

Some competitor has become minor share and tend to decrease such as Firefox, Intenet Explore and Edge. These browser typically will be left behind and may vanish from the market like Netscape. The lesson learned can be taken from Opera, the company admit defeat from market and switch the browser engine to Chromium same as Chrome [13]. Although Opera's Market Share only a little, the trend show positively increase. The other fact about new browser such as UC Browser has left Internet Explorer behind. UC Browser adopt Chromium Engine as stated by Hal9000 [12].

6. Conclusion

The conclusion of this study may lead to statement that chrome has strong point in term of architectural and development concept. Although, Chrome has the most frequent report of weakness, the responsiveness to update the report is the best compare to other popular browser. Strong point of support, conceptual and technical is the real facts that Chrome has. Other browsers must defined some strategy to survive or will vanish like Netscape. Several example has been shown such as adoption Chromium Engine for better quality and performance and add some specific feature to make differentiation. Chromium engine domination become threat for browser variance on the market.
Further and depth study may focus to explore the architecture of browser that manage memory and other resources effectively. The result may solve or minimize development conceptual weakness of the browser as discussed above. This recommendation has strong relation to operating system research area..

Acknowledgement

This work is funded by PITTA scheme form University Indonesia. The authors would like to thank for this funding

References

[1] C. Reis, A. Barth, and C. Pizano, "Browser Security: Lessons from Google Chrome," *Queue*, vol. 7, no. 5, p. 3, 2009.
[2] T. Meskauskas, "What Are The Most Secure Web Browsers In 2018?," *PCRisk*, 2018. [Online]. Available: https://www.pcrisk.com/computer-technician-blog/windows/12350-most-secure-browsers-2018.
[3] P. Mell, K. Scarfone, and S. Romanosky, "A Complete Guide to the Common Vulnerability Scoring System Version 2.0," *FIRSTForum Incid. Response Secur. Teams*, pp. 1–23, 2007.
[4] -, "Common Weakness Enumeration," *http://cwe.mitre.org/*, 2018. [Online]. Available: http://cwe.mitre.org/.
[5] A. Barth, C. Jackson, and C. Reis, "The Security Architecture of the Chromium Browser," *Proc. WWW 2009*, 2008.
[6] D. Smith, "The Yontoo Trojan: New Mac OS X Malware Infects Google Chrome, Firefox And Safari Browsers Via Adware," *International Business Time*, 2013.
[7] D. Goodin, "MySQL.com breach leaves visitors exposed to malware," *The Register*, 2011.
[8] -, "On Browser Security Updates, OS and Internet Use," *HoopsOnline*, 2016.
[9] D. Szabo, *FRAUD SMARTS - Fraud Prevention Handbook*, 2017 Editi. eFraud Prevention, LLC, 2012.
[10] -, "Google Chrome Vulnerability Statistics," *CVE Details*, 2018. [Online]. Available: https://www.cvedetails.com/product/15031/Google-Chrome.html?vendor_id=1224.
[11] P. S. Satish, "Web Browser Security : Different Attacks Detection and Prevention Techniques," vol. 170, no. 9, pp. 35–41, 2017.
[12] HAL9000, "7 Chromium Based Browsers With Extra Features," *Raymond.cc*, 2017. [Online]. Available: https://www.raymond.cc/blog/chromium-browser-alternatives-with-extra-features/. [Accessed: 14-Apr-2018].
[13] S. Anthony, "Opera admits defeat, switches to Google's Chromium," *Extrme Tech*, 2013. [Online]. Available: https://www.extremetech.com/computing/148312-opera-drops-presto-switch-to-google-and-apples-webkit-rendering-engine. [Accessed: 15-Jul-2018].