Volume: 14 Issue: 06 Pages: 5672 - 5677 (2023) ISSN: 0975-0290

Malware Detection in Web Browser Plugins Using API Calls with Permissions

Mohammad SahinurHossen[1]
Email: mohammad.hossen@student.nmt.edu
Rakibul Islam[1]
Email: mdrakibul.islam@student.nmt.edu
Md Nasef Ur Rahman Chowdhury[1]
Email: naseef.chowdhury@student.nmt.edu
Ahshanul Haque[1]
Email: ahshanul.haque@student.nmt.edu
Qudrat E Alahy Ratul[2]
Email: ratul.kuet@gmail.com
[1]Department of Computer Science, New Mexico Institute of Mining and Technology
[2]Department of Computer Science, Boise State University

ABSTRACT

With the exponential growth of internet users, web browsers play an essential role in gathering knowledge, social networking etc. Browser plugin/add-on is a unique feature of modern browsers that allows for adding new gimmicks to the browser functionality. Although this tool is handy, it poses a significant risk as it can collect and store users browsing history, passwords and more. Hence, attackers can try injecting malicious browser add-ons that can utilize security loopholes wherein the attacker may access user-critical data on the host device. The Smart Extension Malware Detector (SEMD), a reliable browser malware detection system that relies on extension development API calls and privileges using outfit machine learning approaches, was suggested and created by us. The research outcomes demonstrate that the SEMD model outperformed peer models while lowering the difficulty of the detection procedure.

Keywords -**Malware Detection, Browser Add-ons, Machine Learning.**

Date of Submission: March 02,2023 Date of Acceptance: April 11,2023

I. INTRODUCTION

The browser is now the preferred application for the Internet and is used for a wide range of tasks like sending and receiving emails, listening to music, and conducting online banking [1]. Because of this, the extensions are integrated into browsers like Chrome, Firefox, Safari, and Internet Explorer: add-ons that let users tweak the fundamental functionality to personalize their browsing experience. Imagine that a user has installed several different search toolbars, a password manager, and ad blockers that, if activated, employ well-defined APIs to intercept web page content and alter each page the user views.
A client can introduce a vast expansion from merchants' official stores such as Chrome Web Store, Firefox Expansions, and Safari Expansion Exhibition conjointly from third-party websites. Chrome takes after a security assurance engineering to decrease the vulnerabilities and noxiousness in expansions to a certain degree [2]. Security engineers anticipate that the Chrome extensions will be devoid of extreme eagerness and that all attacks will come from malicious websites, including malicious javascript codes. If a user does not consistently install extensions from the official Chrome Web Store instead of a variety of third-party extensions, malware is likely to carry out harmful operations within the browser. The same claimcomes from the Firefox, Safari, and Web Pioneer browser destinations. We propose a solution based on tracking the various ex- tension development API requests made by possibly harmful extensions. In order to identify harmful extensions with greater intelligence, we created the Smart Extension Malware Detector (SEMD), which uses several machine learning (ML) classifiers. Extensions were gathered from various extension web stores and third-party websites, and Extension Development API calls and related permissions were extracted from those Extensions. In a subsequent section of this study, we discuss our methodology, experimental strategies, and findings in depth. The primary objective was to use ensemble learning over the four involved ML models to reliably and rapidly detect malware. The following is a list of this paper's main contributions: By leveraging ensemble learning techniques and extension development API calls and permissions, we provided a practical approach for detecting browser malware. Our SEMD model outperformed peer models while lowering the complexity of the detection procedure, according to experimental data.

II. MOTIVATION AND RELATED WORKS

Malware is any harmful program used to disrupt a user's activities, steal sensitive data, or gain access to a system. Its adverse intention goes against the computer user's needs. These are made to get into computer systems and network resources, mess up how computers work, and steal personal information without the owner's permission. They threaten internet access, user data privacy attacks [3], and host security [4, 5].

A. Different Types of Malwares

With numerous types of malicious software and many harmful computer programs within each type, categorizing and recognizing each piece of malware precisely from other programs is crucial. It will be helpful to comprehensively understand how these various types of threats differ and operate if we categorize them.

• **Virus** - A computer virus is a small software designed to affect the operation of a computer without the user's consent or knowledge. It can make copies of itself and run them, and it can put its code in the way of another program [6].

• Worms - Worms are pieces of software that infect computers by taking advantage of security flaws in operating systems [7]. Hostile attackers might use them to launch distributed denial-of-service (DDoS) assaults, steal sensitive data, and conduct ransomware campaigns [8].

• **Trojan Horse -** Trojans is software that, using a remotecontrol interface, controls the other computer in full compliance with a specified program [9]. Once downloaded by unwary users, Trojan horses can establish negative control of the victim's machine [8].

• Spyware and keylogger - Spyware is software that monitors the activities of computer users without their knowledge or permission [10]. It includes passwords, Card PINs, payment details, and unstructured communications. Spyware that observes user behavior is referred to as a keylogger [11].

• **Adware** - Adware tracks the surfing habits of the user to determine which adverts to show and what content to display. Adware is similar to spyware, but unlike spyware, it does not download software on a user's computer [12].

• **Ransomware** - Ransomware prevents a target from accessing its data until a ransom is paid [13].

• **Fileless Malware** - An attack known as fileless malware takes advantage of software pre-installed on mobile de-vices and turns it into a destructive tool for cyber-thieves to use against mobile devices [14, 15].

• **Bots/Botnets** - A software application that, when given instructions, carries out automatic tasks is known as a bot[8, 16].

B. Browser Extension Behavior

In this subsection, we will discuss the evolution of browsers and their extensions. Our primary area of attention is the backdrop of the Chrome and Firefox browsers and their extensions.

1) Chrome Browser and extensions: Chrome implements a multi-process architectural style consisting of a single browser process running in privileged mode [17]. A typical Chrome extension contains content scripts and an extension core. A content script written in JavaScript may be injected into a web page upon page load. The script is then executed within the render process area to access the DOM tree.

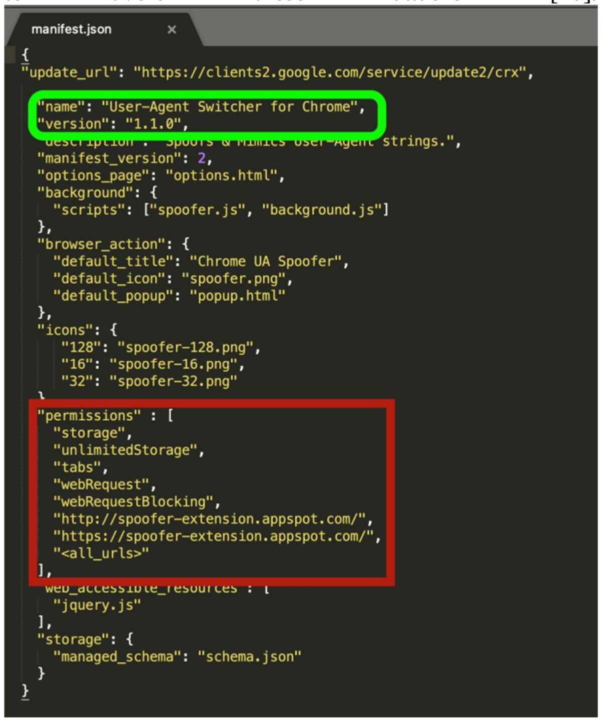
2) Firefox browser and extensions: Mozilla Firefox exten- sions are written in XUL (XML User Interface Language) and use JavaScript and CSS to provide functionality and appearance [18]. Extensions can change the browser's user interface, access the DOM of web pages, and transport data across networks.

C. Extension Source Code

The web store distributes extensions in the form of CRX files, which are ZIP archives with a specific and unique header. All extension files, including the extension's source code (JavaScript/HTML/CSS), local images, and manifest.json are included within the CRX archive file [8]. The manifest.json file is a JSON-formatted metadata record that contains the extension's name, version, description, and permissions. The two most common forms of extension scripts are content script and background script [19].

• **Cross-Site Scripting -** Cross-site scripting (XSS) attacks are caused by direct interaction with untrusted online content. For instance, an attacker may be able to inject a script into an extension that utilizes eval or document. Write without sanitizing the input [5]. In a recent case, a popular RSS aggregation add-on assessed data from an arbitrary website's ¡description¿ element without doing the necessary sanitization [20].

• **Replacing Native APIs -** By replacing regular DOM APIs with methods of its specification, a malicious web page might mislead (and eventually compromise) a browser extension. These fraudulent ways may seemingly imitate native techniques and deceive an extension into committing an offense. Firefox automatically encapsulates references to untrusted data with an XPC Native Wrapper to avoid these attacks [20].



```
{
"update_url": "https://clients2.google.com/service/update2/crx",

  "name": "User-Agent Switcher for Chrome",
  "version": "1.1.0",
  "description": "Spoofs & Mimics User-Agent strings.",
  "manifest_version": 2,
  "options_page": "options.html",
  "background": {
    "scripts": ["spoofer.js", "background.js"]
  },
  "browser_action": {
    "default_title": "Chrome UA Spoofer",
    "default_icon": "spoofer.png",
    "default_popup": "popup.html"
  },
  "icons": {
     "128": "spoofer-128.png",
     "16": "spoofer-16.png",
     "32": "spoofer-32.png"
  },
  "permissions" : [
    "storage",
    "unlimitedStorage",
    "tabs",
    "webRequest",
    "webRequestBlocking",
    "http://spoofer-extension.appspot.com/",
    "https://spoofer-extension.appspot.com/",
    "<all_urls>"
  ],
  "web_accessible_resources" : [
    "jquery.js"
  ],
  "storage": {
    "managed_schema": "schema.json"
  }
}
```

Figure 1 - A manifest that displays the API permission for a url.

• **JavaScript Capability Leaks -** JavaScript capability leaks [21] are an additional means of exploiting extensions. If an extension compromises one of its internal objects to a malicious web page, the attacker can frequently access additional JavaScript objects, including complex extension APIs [20].

D. Installation and permission of Extension

The Chrome Web Store is the official portal for users to discover and install Chrome extensions [22]. Developers develop extensions and publish them to the web store for customers to download, similar to iOS and Android app stores. Additionally, extension authors may push out patches and updates without end-user interaction [22].

Chrome requires extensions to specify the permissions required to use the various extension API components. For example, Figure 1 demonstrates a manifest file section re-questing access to the webRequest and cookies APIs [22]. The webRequest permission enables the extension to view and analyze traffic and intercept, halt, or change in-flight requests by registering HTTP callbacks [22]. The cookies API lets extensions obtain, modify, and monitor cookie modifications. Extension API permissions work with optional host permissions to restrict API access to defined URLs [22].

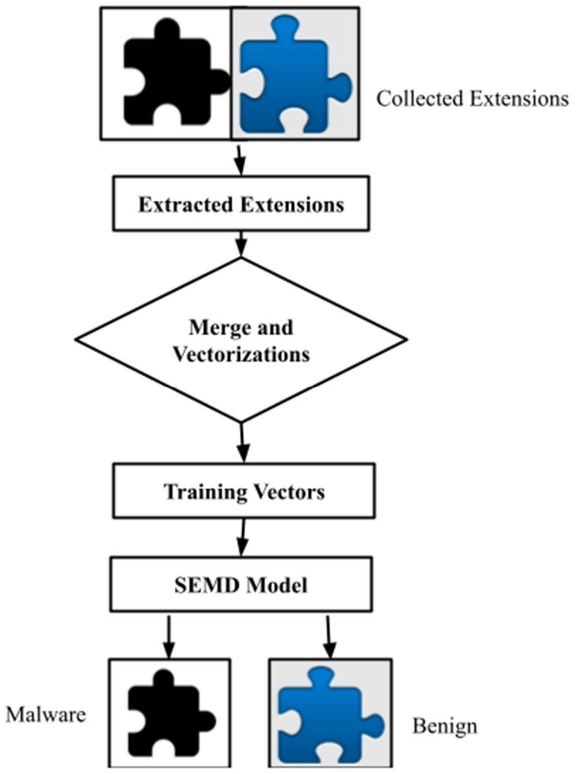


Figure 2 - Work flow of Smart Extension Malware Detector.

III. OUR METHODOLOGY

Due to many challenges in detecting browser extension mal- ware, we proposed an intelligent extension malware detector (SEMD) that can detect malware, utilizing machine learning models and a smart combination of permission and API-based mechanism. Figure 2 shows the workflow structure of the SEMD model. Browser extension files were collected using the data collector module. The Data Vectorization module is used to analyze collected extension files following below

1. Extract all collected extension
2. Merge essential data
3. Data vectorization
4. Prepare training vectors

In our SEMD model, various ML classifiers that will be employed to create vector datasets for training the APIs and permissions are needed. Based on experience, vectors in any acquired training data set are classified as malicious or benign. As shown in figure 2, the following subsections detail the overall process step-by-step.

A. Data Collection

The Extensions are gathered in this phase from various web stores and third-party websites, and the data will be utilized to train ML models. This information is gathered from malware-infected third-party websites during various periods. In order to enable a quick and accurate SEMD, we combined four well-known dataset sources of extensions into a colossally big dataset. Additionally, well-known datasets will allow a fairer comparison of comparable peer work in literature [23]. From the figure 3, First Extension Downloader will the extension file and wait for the completed status. The downloaded file is sent to the JSON parser module to parse the Manifest.json and API file and make a data file.

B. Data Processing and Vectorization

Two sub-phases make up the preparation for training our model. The data file after vectorization is shown in figure 1.

• **Raw Data Extraction -** We create a file parser tool to extract the permission list and extension development API list. We grabbed the manifest file for the permission list and passed it into a parser to acquire the permissions used.

• **Vectorization -** By using binary one-hot encoding, the vectorization procedure is carried out (BOH). Each Chrome extension file has 65 permission fields, and if the API or related permission key is present in the JSON file, the vector's value is set to 1 otherwise. The table was vectorized and now resembles table 1. The 50K extensions were shown here; however, there were 12 extensions.

C. ML Classifier Effectiveness Usage

With the anticipation of varying performance levels, SEMD employs the aforementioned ML classification models of Random Forest (RF), Support Vector Machine (SVM), Decision Tree (DT), and Logistic Regression (LR) [23, 24]. Large data sets are used in modeling, which slows down the training process but yields substantially greater accuracy than small data sets. Due to the frequent application of SEMD as opposed to its single training phase, accuracy is more crucial than training speed [23, 24]. Here, we demonstrate how well these four ML classifiers performed regarding training and test data correctness. The Decision Tree (DT) has the highest accuracy among these four ML classifiers. The accuracy of the Training and Test data for the four classifiers is shown in table 2.

• **Random Forest (RF) -** Random forests (RF) are indeed a mixture of decision trees in which each tree is reliant on the values of a randomly generated vector selected separately and with the same distribution for all trees in the forest [25]. Both classification and regression tasks may be performed with RF.

• **Decision Tree (DT) -** A decision tree topology is a special tree structure that resembles a flowchart and is made up of three main components: decision nodes that represent characteristics, edges or branches that represent the many potential values of those attributes, and decision trees [26]. In DT, data is continually divided according to a particular attribute.

• **Support Vector Machine (SVM) -** Support Vector Machines (SVM) is a reliable and quick classification method that performs better with sparse data than com- peting methods[27]. SVM performs classification and regression analyses on data.

• **Logistic Regression (LR) -** In contrast to linear regression models, which make very similar assumptions about the connection between the dependent and independent variables, logistic regression is conceived of as a specific example of a generalized linear model [28]. It is the approach of choice for issues involving binary categorization.

IV. EXPERIMENT RESULTS

The experimental result is mainly divided into three subcategories. These are given below.

A. Permission uses by extensions

The browser extensions run in the web browser and require permission to read or change on web pages where users visit. The chrome browser has 65 types of permissions and three categories. Extensions can request three types of permissions, and they can be indicated using the corresponding keys in the manifest [29]:

• Elements from such a set of recognized strings are included in privileges (such as ”geolocation”)[29].

• Typical permissions are similar to compulsory permissions however; optional privileges are provided by the ex- tension’s user during runtime instead of beforehand [29].

• One or more match patterns that provide access to one or more hosts are contained in host permissions [29].

From these 65 permissions, the figure 5 shows which permissions is used by the extension. Figure 5 also explain which permission is used more and which is less.

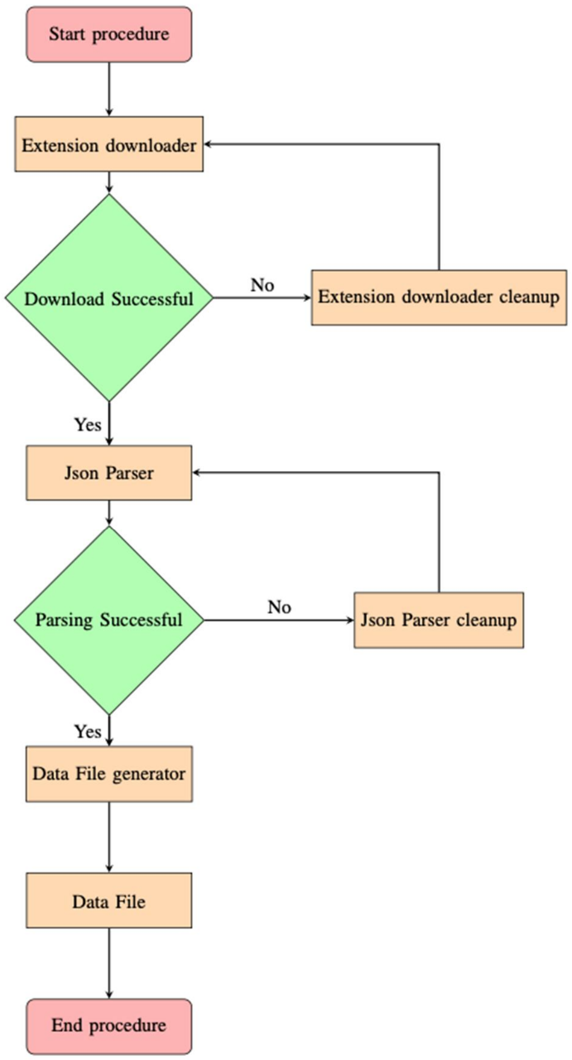


Figure 3 - Data Collection procedure

Name	Permissions																								
	acti	con	stor	webl	tab/t	bacl	coo	all_u	notif	gcm	alar	unli	wel	ider	web	des	idle	tap	ider	tts	decl	hist	boo	top$	URL
Fantasy Targets	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
Giving Assistant Button	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
Good Quotes	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Free Email Signature Generator by cloudHQ	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
Baseliner	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
__MSG_extension_name__	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Boomerang Calendar	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
Hypothesis - Web & PDF Annotation	0	0	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
MySmartPric	0	0	1	0	1	0	1		1	1		1	0	0	0	0	0	0	0	0	0	0	0	0	1
Note Anywhere	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
Mindful Browsing	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1
Polar Tab	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table1 – Vectorization table for the 12 extensions with respect to their uses mission.

The most used permission is storage permission. Among 50K extensions, 34.5Kextensions were using storage permissions.

B. Permission Over Used by the Extensions

When installed extensions use extra permission, a warning may appear that the extensions access the user's sensitive data. The warning indicates that the app has the potential to be hazardous. Accept the warning to give the app or plugin the right toaccess the user's details, or decline to do so[30]. The extensions could ask for authorization, and they might pose risks such,

• **High Alert -** When an application or plugin asks per-mission to view all data on your computer and the websites you visit, it signifies that it can access just about everything. That may be private files or your camera, either within or outside of your browser [30].

• **Medium Alert -** These warnings could ask for access to your data across all the websites you visit, granting you the ability to read, request, or edit information from each page you see (bank account, Facebook). Your information on a list of websites allows for reading, requesting, or editing information on webpages you visit on the list of selected websites [30].

• **Low Alert -** This asks to see the list of installed programs, add-ons, and themes: The app or extension may start programs, extensions, and themes that you've installed and activate, disable and remove them. your saved pages Your bookmarks may be read, modified, added to, and organized by the apps or plugin [30].

In the figure 6 near about 1 percent extension overuse their permission.

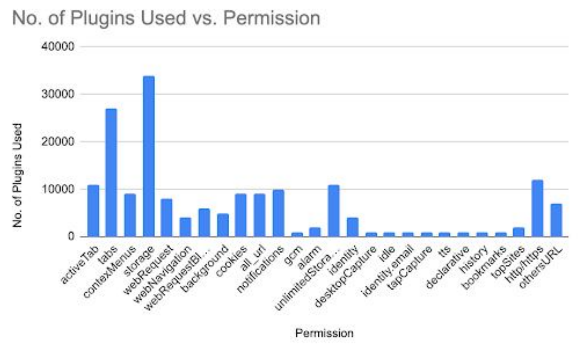


Figure 5 – No of plugins vs permission uses

C. ML classifier accuracy

The accuracy of the classifier is defined as the measure of the number of correct decisions made by the total number of test examples. If you are functioning on a classification task, the best accuracy score is 100 percent; alternatively, the greatest accuracy score for a regression problem is 0.0 error. Calculating accuracy is as simple as dividing the number of correct forecasts by the overall number of guesses. Accuracy can be improved by adding more data, algorithmic tuning, modifying hyperparameters, etc. In the table 2 we use four ML classifier, these are RF, DT, SVM and LR. The classifier is good if the RF classifier accuracy is more than 92.49 in both training and test data. Our experimental results show that the training data accuracy is 98.67 percent and the test data accuracy is 98.23 percent. The accuracy of DT is 67.53 percent then is considered good, and its accuracy can be improved by tuning its hyperparameter. In our experiment, the training accuracy of DT is 98.87 percent, and the test data accuracy is 98.68 percent. For the SVM, if the accuracy is 96.49

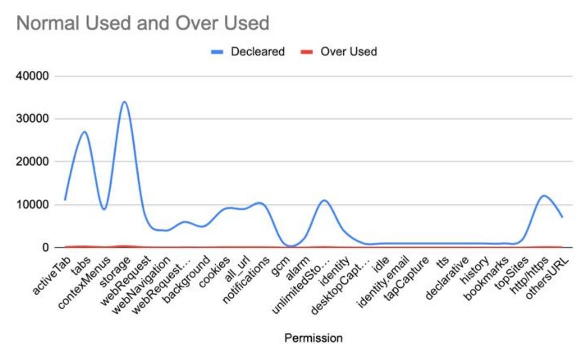


Figure 6 - Permission over use vs normal use

ML Classifier	Train Accuracy(%)	Test Accuracy(%)
Random Forest(**RF**)	98.67	98.23
Decision Tree(**DT**)	98.87	98.68
Support Vector Machine(**SVM**)	95.70	95.11
Logistic Regression(**LR**)	96.72	96.34

Table2 - Machine learning classifier accuracy

percent, it can be considered good. In our experiment, the SVM training data accuracy is 95.70 percent, and the test data accuracy is 95.11 percent. Like the above tree classifier LR, training data accuracy is 96.72 percent, and test data accuracy is 96.34 percent.

From the table 2 we can conclude that the more accurate ML classifier for my experiment is DT and the least accurate classifier is SVM for both training and test data.

V. CONCLUSION

In this study, new machine learning (ML) models were used to enhance the detection of browser extension malware in massive datasets. The new approach is more effective at detecting additional malware types. SEMD successfully separates malicious from benign extensions gathered from various extension web stores and third-party websites. In future work, we will plan to work for an extensive dataset like 10 million extensions. We will add a relational call between API call and permission. In the current procedure, we add four ML classifiers, and in the future, we want to add two more classifiers called Liner Discrimination Analysis (LDA) and K- Nearest Neighbor (KNN). We will also modify the classifiers hyperparameter for better experimental results.

REFERENCES

[1] M. Weissbacher, E. Mariconti, G. Suarez-Tangil, G. Stringh- ini, W. Robertson, and E. Kirda. Ex-ray: detection of history- leaking browser extensions. In Dec. 2017.

[2] Y. Wang, W. Cai, P. Lyu, and W. Shao. A combined static and dynamic analysis approach to detect malicious browser extensions. Security and Communication Networks, 2018, May 2018.

[3] R. Islam, M. S. Hossen, and D. Shin. A mapping study on privacy attacks in big data and iot. In 2022 13th International Conference on Information and Communication Technology Convergence (ICTC), 2022.

[4] S. Talukder and Z. Talukder. A survey on malware detection and analysis tools. International Journal of Network Security Its Applications, 12, Mar. 2020.

[5] A. Dhammi and M. P. Singh. Behavior analysis of malware using machine learning. 2015 Eighth International Conference on Contemporary Computing (IC3), 2015.

[6] A. Damodaran, F. Di Troia, C. A. Visaggio, T. Austin, and M. Stamp. A comparison of static, dynamic, and hybrid analysis for malware detection. Journal of Computer Virology and Hacking Techniques, 13, Feb. 2017.

[7] F. Cohen. A formal definition of computer worms and some related results. Computers Security, 11(7), 1992.

[8] 12 Types of Malware + Examples That You Should Know — crowdstrike.com. https://www.crowdstrike.com/cybersecurity-101/malware/types-of-malware/. [Accessed 24-Aug-2022].

[9] Z. Zhenfang. Study on computer trojan horse virus and its prevention. International Journal of Engineering and Applied Sciences, 2(8), Aug. 2015.

[10] T. Stafford and A. Urbaczewski. Spyware: the ghost in the machine. In volume 14, Jan. 2004.

[11] P. Tuli and P. Sahu. System monitoring and security using keylogger. In 2013.

[12] J. Gao, L. Li, P. Kong, T. Bissyande ́, and J. Klein. Should you consider adware as malware in your study? In Feb. 2019.

[13] P. Okane, S. Sezer, and D. Carlin. Evolution of ransomware. IET Networks, 7, May 2018.

[14] A. Afreen, M. Aslam, and S. Ahmed. Analysis of fileless malware and its evasive behavior. In Oct. 2020.

[15] B. Sanjay, D. Rakshith, R. Akash, and V. V. Hegde. An approach to detect fileless malware and defend its evasive mechanisms. In 2018 3rd International Conference on Computational Systems and Information Technology for Sustain- able Solutions (CSITSS). IEEE, 2018.

[16] S. S. Silva, R. M. Silva, R. C. Pinto, and R. M. Salles. Botnets: a survey. Computer Networks, 57(2), 2013.

[17] L. Liu, X. Zhang, G. Yan, and S. Chen. Chrome extensions: threat analysis and countermeasures. In NDSS, 2012.

[18] H. Shahriar, K. Weldemariam, M. Zulkernine, and T. Lutel- lier. Effective detection of vulnerable and malicious browser extensions. Computers Security, 47, Nov. 2014.

[19] N. Pantelaios, N. Nikiforakis, and A. Kapravelos. You’ve changed: detecting malicious browser extensions through their update deltas. In Oct. 2020.

[20] A. Barth, A. Felt, P. Saxena, and A. Boodman. Protecting browsers from extension vulnerabilities. In Jan. 2010.

[21] Q. E. A. Ratul, N. Chowdhury, H. Soliman, M. S. Chaity, andA. Haque. Android malware detection in large dataset: smart approach. In Feb. 2020.

[22] A. Kapravelos, C. Grier, N. Chachra, C. Kruegel, G. Vigna, and V. Paxson. Hulk: eliciting malicious behavior in browser extensions. In 23rd USENIX Security Symposium (USENIX Security 14), San Diego, CA. USENIX Association, Aug. 2014.

[23] M. N.-U.-R. Chowdhury, Q. E. Alahy, and H. Soliman. Advanced android malware detection utilizing api calls and permissions. In H. Kim and K. J. Kim, editors, IT Conver- gence and Security, Singapore. Springer Singapore, 2021.

[24] Q. E. A. Ratul, N. Chowdhury, H. Soliman, M. S. Chaity, and A. Haque. Android malware detection in large dataset: smart approach. In Feb. 2020.

[25] L. Breiman. Random forests. Machine learning, 45(1), 2001.

[26] I. Jenhani, N. B. Amor, and Z. Elouedi. Decision trees as possibilistic classifiers. International Journal of Approximate Reasoning, 48(3), 2008. Special Section on Choquet Integra- tion in honor of Gustave Choquet (1915–2006) and Special Section on Nonmonotonic and Uncertain Reasoning.

[27] M. Bala, V. Athira, and A. Rajendran. Efficient multi-level lung cancer prediction model using support vector machine classifier. IOP Conference Series: Materials Science and Engineering, 1012, Jan. 2021.

[28] P. Tsangaratos and I. Ilia. Comparison of a logistic regression and naive bayes classifier in landslide susceptibility assessments: the influence of model’s complexity and training dataset size. Catena, 145, 2016.

[29] Declare permissions - Chrome Developers — devel- oper.chrome.com. https : / / developer . chrome . com / docs / extensions / mv3 / declare permissions/. [Accessed 24-Aug- 2022].

[30] Permissions requested by apps and extensions - Chrome Web Store Help — support.google.com. https : / / support . google . com/chrome webstore/answer/186213?hl=en#zippy=\% 2Chigh-alert\%2Cmedium-alert. [Accessed 24-Aug-2022].