

# A Systematic Literature Review and Quality Analysis of Javascript Malware Detection

MD. FAHIMUZZMAN SOHAN<sup>1</sup> AND ANAS BASALAMAH<sup>2</sup>

<sup>1</sup>Department of Software Engineering, Daffodil International University, Dhaka 1207, Bangladesh

<sup>2</sup>Computer Engineering Department, Umm Al-Qura University, Mecca 24231, Saudi Arabia

Corresponding author: Md. Fahimuzzman Sohan (fahimsohan2@gmail.com)

**ABSTRACT** **Context:** JavaScript (JS) is an often-used programming language by millions of web pages and is also affected by thousands of malicious attacks. **Objective:** In this investigation, we provided a general view and a quick understanding of JavaScript Malware Detection (JSMD) research reported in the scientific literature from several perspectives. **Method:** We performed a Systematic Literature Review (SLR) and quality analysis of published research articles on the topic. We investigated 32 articles published between the year 2009 to the year 2019. **Results:** Selected 32 papers explained in this article reflect the outline of what was published so far. One of our key findings is the performance of Machine Learning (ML) based detection models were relatively higher than others. We also found that only a few papers were able to achieve high scores according to the quality assessment criteria. **Conclusion:** In this SLR, we summarized and synthesized the existing JSMD studies to identify the previous research practices and also to shed light on future guidelines in the malware detection space. This study will guide and help future researchers to investigate the previous literature efficiently and effectively.


**INDEX TERMS** Cybersecurity, systematic literature review, malicious code detection, javascript attacks, javascript malware detection.

## I. INTRODUCTION

The risk and vulnerabilities of websites and web applications are constantly increasing. Consequently, injecting malicious codes to websites, PDF documents, browsers, applications are becoming handy and essential channels [27]. According to the report [35], a total of 1062.25 million malicious URLs were found till now; besides, the report says over 0.35 million new malicious and unwanted applications are created every day. In this circumstance, JS is a dynamic and lightweight scripting language, and it has broad participation in website and web application services; JS is used in web pages interface design, creating cookies, mobile apps, games, and so on. The extensive and miscellaneous use of the web, makes JS be an active gateway for various web attacks (e.g., Cross-Site Scripting (XSS), drive-by downloads [31]) to spread malware and malicious components into users platforms [32]. Malware refers to software that is written with malicious intent; viruses, worms, ransomware, spyware, etc. are the parts of malware [36]. The attackers use different vulnerabilities (e.g.

code vulnerability, browser vulnerabilities, PDF document vulnerabilities) to inject malicious JS code into the users' environment. The goal of malware detection techniques is to identify and prevent malicious activities to protect the computer system from any damage. Most of the researchers have worked with two major malware detection techniques: signature and behavior-based detection [37]. The signature-based technique mainly works for known attacks and uses the database of the signature by matching the behavior of the malware to detect them. Behavior-based detection analyses user behavior and the statistics of a process in a normal situation [55]. An essential activity of malware detection technique is malware analysis, which helps to identify the characteristic of malware [38]. Generally, antivirus software is used to detect JS malware and vulnerability. Most of the antivirus software programs use the signature or dictionary-based detection techniques, but they can't detect unknown malicious content properly [26].

JS malware detection is an active research area, therefore, various detection models were proposed due to the limited detection capability of existing antivirus software. Most existing JS malware detection research articles have

The associate editor coordinating the review of this manuscript and approving it for publication was Zhitao Guan .

used two methods: dynamic [11], [16], [21], [29] and static [12], [15], [18] analysis. The static approach can detect only known malware, but the dynamic approach can also make decisions about new malware by analyzing the behavior of the malware. Nowadays researchers are using various machine learning techniques to detect JS malware [6], [13], [17], [25]. ML techniques include feature selection, using various ML classifiers, and performance measures.

This article aims to investigate the trend and current scenario of JSMD models reported in the scientific literature in a systematic way. To the best of our knowledge, this study is the first initiative to conduct an SLR in JSMD research. In this review investigation, we studied 32 JSMD articles published from the year 2009 to 2019. Additionally, we categorized the paper according to the publication information and investigated it from several perspectives. It is the summarized article of JS malware detection research in the past years. Our contributions in this article are:

- Identifying the primary studies (PS) related to JSMD research and extracting several properties from the studies.
- Perform a quality analysis questionnaire on the PS
- Presenting the current scenario of JSMD research including publication information, datasets, detection methods, performance evaluation criteria, and detection performances.
- Summarising the findings and providing future guidelines to the researcher to improve the problems in the future.

The rest of the article is organized as follows: Section II describes the method we used to perform the SLR. Section III is the key part, which includes the results and discussion of the investigation. Section IV validates the threats of our study. In Section V the limitations of this study are presented. Finally, Section VI concluded this literature review study with future guidelines.

## II. REVIEW METHOD

We performed an SLR to summarize the current scenario of JSMD research. The steps of this SLR were prepared followed by Kitchenham's guidelines [40], [41]. The review design, questions, and figures presented in this section are mostly motivated by [42]–[44]. From Kitchenham's guidelines, we conducted this review in three phases: planning the review, conducting the review, and reporting the review results, respectively. Figure 1 represents three steps in detail. The first step of the planning phase is to identify the need for a systematic review. We already described the objective of this SLR in the previous section (I). We developed the review protocol for the validity of this study and to avoid research bias. Two researchers were participated in several meetings and discussions to establish the review protocol of this SLR. The second phase of Figure 1 describes the main steps to perform this review study. In the first step (step 4), we described the research questions to answer the

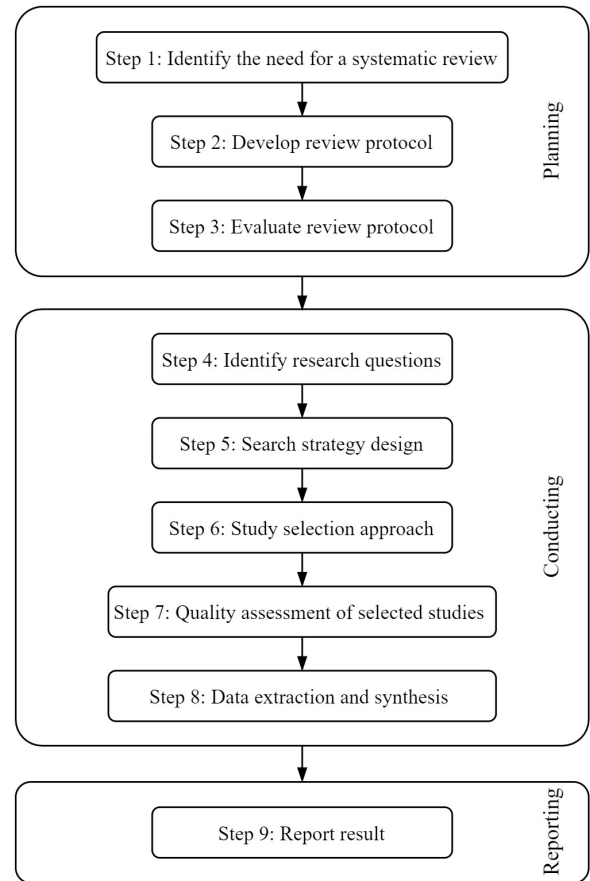


FIGURE 1. Systematic review process.

issues in this SLR. Then the search strategy was designed and implemented in the digital library to collect the PS. The next step is the study selection process, where we performed the inclusion-exclusion process. After that, the quality assessment questionnaire was considered to analyze and assess the quality of each study. Finally, we extracted the required data from each study and devise methods for data synthesis. The following subsections are describing the steps followed while conducting the SLR.

### A. RESEARCH QUESTIONS

We specified the research questions (RQ1-RQ11) of this SLR to determine the insights obtained from the JSMD articles. Table 1 represents the review questions that were prepared systematically. RQ1 analyses the research trend and information of the studies. RQ2, RQ3, and RQ4 give the analytical report about the data sets and data analysis method used in the articles. In the RQ5 and RQ6, we assessed the different detection techniques considered in previous studies. In the next question, we investigated the most commonly used performance metrics in the field of JSMD. RQ8 investigates the performance of the JSMD models proposed by the researchers, where the values of performance metrics were included. RQ9 specifies the performance of ML techniques

TABLE 1. Research questions.

RQ No.	Question	Motivation
RQ1	What is the trend and demographics of the studies on JSMD?	Review the bibliometric studies in the JSMD area.
RQ1.1	What is the annual number of the studies?	Estimate the published article per year on JSMD.
RQ2	Which types of data sets are used for performing the detection?	Identify the datasets that are used in the prediction models.
RQ3	What are the size of the data sets?	Investigate the studies' external validity.
RQ4	What data analysis methods are used to build JSMD models?	Identification of the data analysis methods used to build JSMD models.
RQ5	What detection techniques are used to build JSMD models?	Identification of the detection techniques used to build JSMD models.
RQ6	How many JSMD studies use ML technique?	Identify the usage of ML technique in JSMD.
RQ7	What are the different performance measures used for JSMD studies?	Identify the performance parameters that are used to evaluate the JSMD models.
RQ8	What is the performance of proposed JSMD models?	Investigate the performance of proposed JSMD models.
RQ9	How perform ML based models compared with others?	Investigate the performance of the ML techniques for JSMD.
RQ10	What is the research focus of the articles?	Identify the targeted scope of the research articles.
RQ11	What are the limitations and challenges of the malware detection as reported in the studies?	Identify the limitations and challenges reported in the PS.

used in the studies and compares the result with non-ML techniques based models. RQ10 explores the research focus considered in primary studies. The last question identifies the limitations and challenges reported by the researchers in the selected PS.

**B. SEARCH STRATEGY**

Searching is most important when it needs to include all relevant articles on a specific topic. We have considered Keyword searches, a common method to identify and collect articles from electronic databases [39]. The first part of Figure 2 shows the steps taken for conducting the search strategy. For the first step, the digital libraries were selected by analyzing the previous SLR in the field of software engineering [42]–[45], we considered their most frequently-used databases for our study. After that, we identified the keywords extracted from relevant papers' titles, abstracts, and keywords including alternative terms and synonyms; then applied them using Boolean “AND” and “OR” expressions. Our final search string with keywords was:

- JavaScript AND (“Malware” OR “Vulnerability”) AND Detection
- (“Obfuscated” OR “Malicious” OR “Suspicious”) AND JavaScript AND Detection

The search string was subsequently adapted to suit the specific requirements of each database. We searched each database by titles, abstracts, and keywords. The search of this study was not imitated by the year of publication. Journal papers and conference proceedings were included that are written in English.

**C. STUDY SELECTION**

As Figure 2 shows, our search strategy obtained an initial set of 55 PS. But still, the list may contain some studies that either do not add value to the SLR or do not fall within the purview

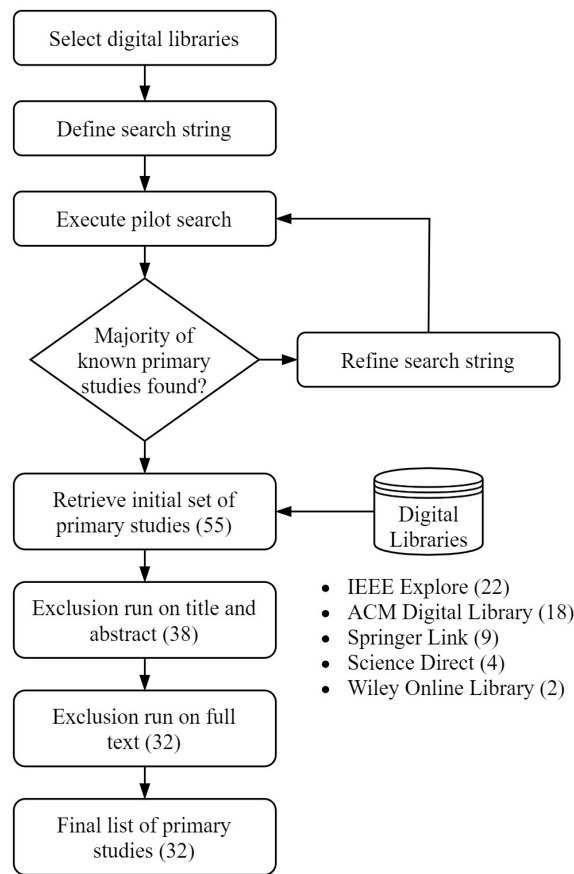


FIGURE 2. Search and study selection process.

of what the review aims to achieve. From this perspective, we necessarily designed inclusion-exclusion criteria below to the initial studies to avoid the studies that do not match the objectives of the SLR.

**TABLE 2. Quality analysis questions.**

No.	Questions	Yes	Partly	No
QQ01	Is the objective of the study clear?			
QQ02	Is the dataset size sufficient for this type of studies?			
QQ03	Is the data collection procedure clearly defined?			
QQ04	Does the author provide sufficient detail about the experiment?			
QQ05	Are the threats to validity given?			
QQ06	Are the limitations of the study given?			
QQ07	Does the study clearly define the performance parameters used?			
QQ08	Are the learning techniques clearly defined?			
QQ09	Are the results clearly stated?			
QQ10	Is there a comparison among techniques?			
QQ11	Does the study add value to the existing literature?			
QQ12	Does the study provide any tool or source code online?			

**Inclusion criteria:**

- Studies that focus on 'JavaScript' malware detection.
- The study outcome must contain detection related information, including test-train samples, Detection Rate.

**Exclusion criteria:**

- Studies that considered the JSMD as a co-topic
- Studies not written in English
- Short papers, less than five pages
- Studies without empirical analysis and clear information of experimental results

In Figure 2, the exclusion process demonstrates two steps: the exclusion of studies by title and abstract and full text of the studies. 55 articles were analyzed based on the title and abstracts of each study; it makes a list of 38 articles. Then, the full text of these 38 articles was analyzed. The full-text analysis process minimizes the list again, it concludes a final list of 32 PS. Finally, the quality assessment criteria are given in the following section were used to obtain the final studies.

**D. QUALITY ASSESSMENT CRITERIA**

We performed a quality analysis questionnaire for assessing the quality of selected PS. This challenging task was developed into considering the suggestions given by Wen *et al.* [46]. Table 2 represents a total of 12 quality assessment questions and that was imposed on 32 articles. Each question can be answered as 'Yes' (1 point), which means the researcher fully agreed with the question for the article; 'Partly' (0.5 points), which means the researcher partially agreed; and 'No' (0 point), which means the researcher fully disagreed with the question for the article. The final score is obtained by 384 (32 × 12) question-answer settings. For each article maximum score could have 12 points and minimum 0 points.

**E. DATA EXTRACTION**

This step is important to extract meaningful information from each study such that the research questions can be answered.

**TABLE 3. Extraction card.**

No.	Attribute Name	Research Question
01	Year of Publication	RQ1.1
02	Detection Technique	RQ5
03	Data Analysis Method	RQ4
04	Performance Measures	RQ7
05	Dataset(s) used	RQ2, RQ3
06	Target Scope	RQ10
07	Detection Rate and Performance	RQ8, RQ9
08	Machine Learning	RQ6

We filled a form by extracted data from the PS. We assigned necessary data attributes in the form needed to answer the questions. Table 3 is the extraction card, which represents eight data attribute names assigned for each study. The extraction card covers publication information, datasets, detection techniques, data analysis techniques, performance metrics, the performance of detection models, targeted scopes, and using ML in the articles. The extracted data was stored in spreadsheets to use in the data synthesis process.

**F. DATA SYNTHESIS**

Data synthesis accumulates and combines facts and figures from the selected studies to build responses to the research questions [42]. We collected both quantitative and qualitative data that include the performance of different detection techniques and publication information, datasets information, detection techniques, targeted scopes, and using ML, respectively. We considered various techniques for our collected articles to synthesize data. We used different visualization tools including bar charts, pie charts, line graphs, tables to answer the questions. Our visualization techniques present and summarize the results of our investigation with a clear view and understanding.

**III. RESULTS AND DISCUSSION**

This section represents the details about selected PS and results obtained from the studies based on research questions. We presented an overview of the studies along with their references. After that, we presented the answer with the necessary discussion and interpretation against each question.

**A. DESCRIPTION OF PRIMARY STUDIES (PS)**

To the best of our knowledge, this is the first SLR investigation in the field of JSMD. We have considered a list of 32 PS (PS01-PS32) after applying various selection criteria. Table 4 represents the list of studies with a unique identifier and its reference number of each study. These 32 studies are completely focused on JSMD research, partially focused studies were avoided. In this following paragraphs, we presented the short description including research techniques, findings, and effectiveness of the detection models of the PS:

Choi *et al.* [1] presented a string pattern analysis technique to detect obfuscation JS attacks on the web pages. They have used three metrics rules, that are N-gram, Entropy, and Word

Size. Finally, using these metrics they have constructed a tool that can detect obfuscated JS malware from web pages effectively.

Cova *et al.* [2] presented a novel approach to detect malicious JS code, where anomaly detection and emulation were combined. Their system used various features and machine learning techniques to catch the behavior of clean JS code. Hence, the behavior of the clean JS code was used to determine the malicious code from the total code. This system is also effective for signature-based obfuscated code detection and publicly available for analysts.

Likarish *et al.* [3] represented a classification based technique to detect malicious JS. They have proposed features that are related to the malicious behavior of JS. Their classification models show high detection and low false alarm rate.

Laskov and Srindic [4] worked for JS-bearing malicious PDF document detection technique where code analysis (static analysis) was conducted. They have collected real-life large-scale datasets from the VirusTotal portal to evaluate their study.

Curtsinger *et al.* [5] described ZOZZLE; an in-browser JS malware detection and prevention approach. Here, abstract syntax tree-based hierarchical features were used to conduct Bayesian classification. The authors reported that their approach achieved a very low false positive Detection Rate and was able to detect malware very quickly.

Schütt *et al.* [6] proposed a method to detect malicious JS behavior at an early stage. They have optimized accuracy and detection time using machine learning techniques. Support Vector Machine (SVM) classifier was used as a learning method. They have used JS code from 0.1 million web pages to conduct their investigation.

Schwenk *et al.* [9] built a system that can automatically collect, analyze, and detect malicious JS code. They have prepared large scale datasets, where 3.4 million were clean and 8,282 malicious web pages. This study is an experimental operation. The proposed automated approach was successful to detect 93% malicious code.

Schmitt *et al.* [7] presented a tool that can detect malicious PDF documents. They have used static and dynamic techniques to identify the vulnerabilities. Their results show that the proposed approach can identify known and unknown both malicious documents with a low False Positive Rate (FPR).

AL-Taharwa *et al.* [8] investigated the existence of readable obfuscated JS code, they also focused on static analysis to prevent suspicious scripts. They developed a JS obfuscated detector called RedJsod. They used an abstract syntax tree (AST) based variable-length context-based feature extraction (VCLFE) scheme. In addition, they collected three datasets from real-world web-pages and then applied them to the RedJsod detector. Finally, study results show RedJsod detector can achieve the highest detection accuracy with a less false positive and negative rate.

Krishnaveni *et al.* [10] presented a strength analysis system for detecting malicious JS in websites. The investigators have extracted features from JS code, such as the presence of suspi-

cious URLs, density, frequency, number of redirections, presence of whitespaces, average line length, enigmatic variable names, and keywords to words ratio; moreover, these features have used to measure the strength of the JS obfuscation. Their developed system is able to achieve satisfactory false positives and negatives detection rates.

Xu *et al.* [11] used the JStill static approach, which identifies the characteristics of vulnerable JS code. Their technique can detect and prevent attacks by obfuscated malicious JS code.

Gorji and Abadi [12] used dynamic analysis for detecting obfuscated JS malware. They have collected a sequence of predictive function calls from a bunch of malicious web pages. They categorized the sequences and applied them into the same cluster based on the Normalized Levenshtein Distance (NLD) metric to create a signature for each cluster. These behavioral signatures were used to detect obfuscated JS malware.

Corona *et al.* [13] constructed their detection technique based on JS code analysis; specifically functions, constants, objects, methods, keywords of JS code. These attributes have been applied to characterize malicious code using machine learning techniques. The detection technique was applied to JS code in PDF documents.

Liu *et al.* [14] conducted their study based on malicious JS in PDF documents using statically extracted features. Their approach demonstrates that a context monitoring code was injected in each PDF document and this code cooperates with their run-time monitor to detect infections in JS execution. They have used 18623 clean and 7370 malicious data to conduct their study and that shows effective results.

AL-Taharwa *et al.* [16] presented JSOD, a static analysis of obfuscated JS detection. They have compared JSOD with the state-of-the-art approaches to detect obfuscated benign and malicious scripts, namely Zozzle and Nofus. Finally, their experimental results have shown their approach can detect obfuscated scripts and their sophisticated variations.

Wang *et al.* [17] presented a machine learning approach for JS malware detection. Firstly they have considered predictive features from the textual information, program structures, and risky function calls. Their proposed technique is able to detect pre-define malware; also can explain newly arrived malware and vulnerabilities.

Xue *et al.* [18] summarized the common behaviors of vulnerable JS samples using Deterministic Finite Automaton (DFA). Their technique uses dynamic execution traces of JS malware to learn DFA. The authors used 10000 clean and 276 vulnerable JS samples to conduct their detection and classification.

Jodavi *et al.* [19] worked on a novel classifier ensemble approach called JSObfusDetector to identify vulnerable JS code. An ensemble of a one-class SVM classifier was used with a binary particle swarm optimization algorithm. Their study shows that JSObfusDetector is able to achieve high precision, recall, and F-measure.

Cosovan *et al.* [20] investigated various detection approaches to JS-based malware attacks. In the study, multiple versions of linear classification, Hidden Markov Models, and ensemble model techniques have been used. A dataset of one million JS files was used to conduct the study with the detection models. Indeed, the investigated results showed a low false-positive rate with the ability to train large datasets.

Takamori *et al.* [33] presented a detection approach for unreadable JS malware. They have combined two probability variables, one is state transition probability of first-order Markov source and another is character appearance frequency. The study results indicate using combined probability variables is better than the individual variable.

Su *et al.* [22] proposed JS code analysis techniques as well as natural language processing. The authors have used tokenizing for JS text and information-theoretic measures to detect obfuscated malicious code. They have claimed that their new approach is competitively more time effective.

Wang *et al.* [23] presented an approach for detecting malicious JS code using a deep learning framework. They have extracted features from JS code by using stacked denoising auto-encoders. Here Logistic Regression classifier was used to identify benign and malicious JS code. Their implemented framework can achieve the highest detection accuracy with less FPR.

Kim *et al.* [24] proposed a forced execution engine to identify the malicious behavior of JS. They have evaluated 50 exploits of popular exploit kits and more than 12000 Chrome extensions. They have tested their technique on 100 real-world JS samples and it was successful over 95% of samples.

Dabral *et al.* [25] presented JS malicious PDF file detection using a machine learning approach. To detect malicious PDF files they extract features from PDF file structure. Besides, they have developed a features extraction tool that is the parser that leverages on Origam tool, it extracts features from JS code that are often used for PDF file attacks. These features were used to develop detection models using machine learning techniques and high accuracy to other existing methods.

Morishige *et al.* [26] presented malicious JS code detection techniques based on the divided URL. They have considered variables which are the segments of the URL. Then dictionary type objects and connection parts of the strings are extracted. The vulnerable JS has identified after comparing and reconstructing these two parts. They concluded that the proposed scheme is more effective than the traditional approach.

Fang *et al.* [27] proposed a Long Short-Term Memory based malicious JS detection technique. They have extracted features from the semantic level of bytecode and used the word vector method. Their proposed method shows high detection accuracy and F1-score.

Ndichu *et al.* [28] studied the limitation on regular signature and heuristic-based malicious JS detection techniques. A neural network classification model was used to learn features. They have deployed SVM and neural networks to create classification models and are used to detect malicious

**TABLE 4. Selected PS with references.**

Study No.	Paper	Ref No.	Study No.	Paper	Ref No.
PS01	Choi 2009	[1]	PS17	Xue 2015	[18]
PS02	Cova 2010	[2]	PS18	Jodavi 2015	[19]
PS03	Likarish 2010	[3]	PS19	Cosovan 2015	[20]
PS04	Laskov 2011	[4]	PS20	Takamori 2015	[33]
PS05	Curtsinger 2011	[5]	PS21	Su 2016	[22]
PS06	Schütt 2012	[6]	PS22	Wang 2016	[23]
PS07	Schwenk 2012	[9]	PS23	Kim 2017	[24]
PS08	Schmitt 2012	[7]	PS24	Dabral 2017	[25]
PS09	Al-Taharwa 2012	[8]	PS25	Morishige 2018	[26]
PS10	Krishnaveni 2012	[10]	PS26	Fang 2018	[27]
PS11	Xu 2013	[11]	PS27	Ndichu 2018	[28]
PS12	Gorji 2014	[12]	PS28	Wu 2018	[34]
PS13	Corona 2014	[13]	PS29	Hou 2018	[29]
PS14	Lui 2014	[14]	PS30	Fass 2018	[30]
PS15	Al-Taharwa 2014	[16]	PS31	Ndichu 2019	[31]
PS16	Wang 2015	[17]	PS32	He 2019	[32]

JS. This study used well known D3M datasets for malicious and JSUPACK for clean samples. Lastly, they have compared their study outputs with other feature learning techniques and found that proposed methods can detect malicious JS more quickly and correctly.

Wu and Qin [34] presented a collaborative training model for detecting obfuscated malicious code, where obfuscation features and malicious features were used which can identify obfuscation and malicious JS code respectively. They build a cooperative relationship between the two characteristics of the collaborative training model. Moreover, the Random Forest classifier was used to classify the model, and the classification report shows high detection accuracy compared to existing models.

Hou *et al.* [29] presented JSPRE, which is an approach to explore web pages more efficiently that is probably malicious. JSPRE exposes a malicious page collection algorithm, based on guided crawling, that initially started with URLs of known malicious web pages. JSPRE technique has been used for static analysis. JSPRE can detect malicious web pages effectively.

Fass *et al.* [30] proposed a static based analysis to detect obfuscated malicious JS. They extracted features from abstract syntax trees and extracted n-grams features to create a JS dataset that is up to date and balanced. They conducted a frequency analysis of specific patterns and applied the Random Forest classifier. Results indicate that their approach has a high detection rate.

Ndichu *et al.* [31] considered Abstract Syntax Tree (AST) and machine learning techniques for their malicious JavaScript detection models. In this study, a neural network model was used to learn context information of texts. The study was conducted with the D3M malware dataset and the top 100 Alexa websites as clean codes. Their study shows that AST features and machine learning are better performers with fast classification ability.

TABLE 5. Results of the quality questionnaire.

No.	Questions	Yes	Partly	No
QQ01	Is the objective of the study clear?	32(100.0%)	00(00.00%)	00(00.00%)
QQ02	Is the dataset size sufficient for this type of studies?	19(59.38%)	05(15.63%)	08(25.00%)
QQ03	Is the data collection procedure clearly defined?	24(75.00%)	07(21.88%)	01(03.13%)
QQ04	Does the author provide sufficient detail about the experiment?	29(90.63%)	03(09.38%)	00(00.00%)
QQ05	Are the threats to validity given?	01(03.13%)	01(03.13%)	30(93.75%)
QQ06	Are the limitations of the study given?	05(15.63%)	01(03.13%)	26(81.25%)
QQ07	Does the study clearly define the performance parameters used?	21(65.63%)	08(25.00%)	03(09.38%)
QQ08	Are the learning techniques clearly defined?	30(93.75%)	02(06.25%)	00(00.00%)
QQ09	Are the results clearly stated?	24(75.00%)	08(25.00%)	00(00.00%)
QQ10	Is there a comparison among techniques?	23(71.88%)	04(12.50%)	05(15.63%)
QQ11	Does the study add value to the existing literature?	22(68.75%)	10(31.25%)	00(00.00%)
QQ12	Does the study provide any tool or source code online?	02(06.25%)	02(06.25%)	27(84.38%)

TABLE 6. Scores assigned to quality questions.

Score	No. of PS	Percentage
Very high (10 and above)	5	15.63%
High (8.5 to 9.5)	13	40.63%
Average (6.5 to 8)	10	31.25%
Low (6 and below)	4	12.50%

He *et al.* [32] proposed a combined technique of static and dynamic analysis to detect malicious JavaScript code. After extracting the features they used classification based models to detect the malicious code. They have collected 1500 web pages and four static key features. Additionally, they have used the Random Forest classifier to create the malicious JavaScript code detection model.

1) QUALITY ASSESSMENT

We already represented the quality questions (QQ) in the previous section, here Table 5 shows the results of the questionnaire. The table indicates that most of QQ were answered positively. QQ01 is the only question where the answer was 100% positive, that means the objective of every primary study was clear. The results of QQ05, QQ06, and QQ12 were mostly negative. Most of the studies have not given any threats to validity and limitation of their work. The answer of QQ11 shows most of the articles are valuable to the existing literature and some of them are partially valuable.

In table 6, we divided the obtained scores from quality analysis into four categories: very high (10 and above), high (8.5 to 9.5), average (6.5 to 8), and low (6 and below). Also, the table summarizes the percentage of PS and the number of studies in each of the four categories. Table 7 provides the list of PS and their quality scores which obtained quality scores in ‘very high’ and ‘high’ categories. These two categories cover 14 studies that obtained 8.5 and above points in the quality analysis process. The full list of quality analysis scores of the studies given in Table ???. Besides, we made available our full length quality analysis scores here: <https://figshare.com/s/3681c20b75f638f51d80>.

TABLE 7. Quality scores of PS under ‘very high’ and ‘high’ categories.

Primary Study	Quality Score	Primary Study	Quality Score
Laskov 2011	11.5	Schütt 2012	9
Corona 2014	10	Cova 2010	9
Xue 2015	10	Ndichu 2018	9
Hou 2018	10	Wang 2016	9
Wang 2015	9.5	Curtsinger 2011	9
Xu 2013	9.5	Su 2016	8.5
Fass 2018	9.5	Lui 2014	8.5

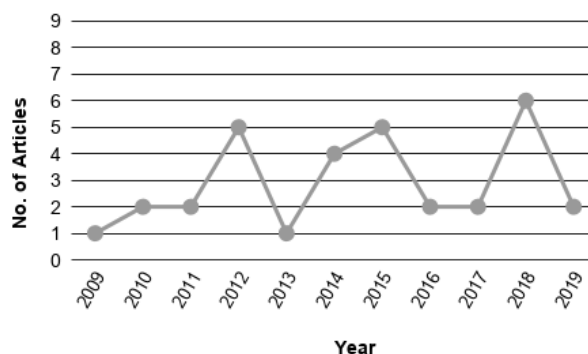


FIGURE 3. Year-wise distribution of studies.

B. RQ1: WHAT IS THE TREND AND DEMOGRAPHICS OF THE STUDIES ON JSMD?

The motive of this research question was to review the bibliometric studies in the JSMD research area. Answers of the following sub-questions reflect the publication information of the PS.

1) RQ1.1: WHAT IS THE ANNUAL NUMBER OF THE STUDIES?

Figure 3 is the year-by-year presentation of selected studies. The year started in 2009 and ended in 2019, we have shown a total of 11 years of data of 32 articles. The figure shows the disparate distribution of articles according to the years. In 2009, we found the first and only one research article on JSMD. Since then several research articles were published on the topic. For the years 2010 and 2011, two papers

**TABLE 8.** Category of datasets based on number of samples.

Category	No. of Samples in Used Dataset	No. of Primary Study(s)	Percentage
Small	0 to 3000	7	21.88%
Medium	3001 to 10000	5	15.63%
Large	10001 and above	20	62.50%

were published each year. After that, the publication rate increased in 2012 as five papers were published in one year. In 2013, again the number of published articles decreased. The figure also shows most of the articles published in the years 2014, 2015, and 2018; where numbers of articles were 4, 5, and 6 respectively. So the overall scenario says published articles are unequally distributed, we haven't found any sequential distribution pattern with the years.

### C. RQ2: WHICH TYPES OF DATA SETS ARE USED FOR PERFORMING THE DETECTION?

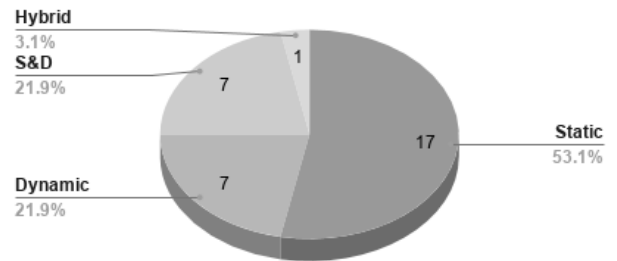
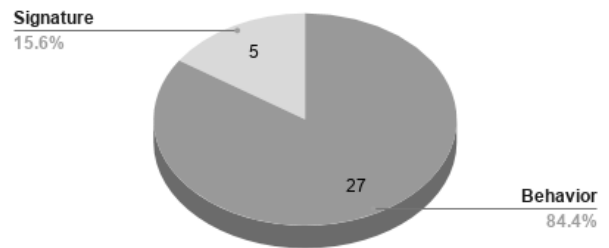
We divided the studies into two categories according to the dataset type, one is public, and another is private. Public dataset refers to openly accessible for everyone and they are distributed freely; private dataset is collected and used by individuals and they are not distributed as public datasets. We found D3M [48] is the only public dataset that was used in the studies. Only 4 (12.5%) studies used public datasets. An important problem is the usage of a private dataset for detection or prediction based learning models [49]. Usually, access to a private dataset is restricted, consequently comparing the results of various learning models become impossible.

### D. RQ3: WHAT ARE THE SIZE OF THE DATA SETS?

The goal of investigating the size of datasets is to determine the external validity of the studies. A large dataset ensures higher external validity than a small dataset. Moreover, dataset size can affect the results of detection models. A detection model that was prepared by large scale training data covers a large learning area and has a greater possibility of achieving more positive results. We divided the studies into three categories based on the size of the datasets used in the studies. We found the required information about the size of used datasets in all of 32 articles. We considered 'large' for the study that used more than 10000 samples, 'medium' that used 3001 to 10000 samples, and 'small' for 0 to 3000 samples in their studies. Table 8 shows: the number of samples among the used datasets, the number of studies, and the percentage according to the three categories.

### E. RQ4: WHAT DATA ANALYSIS METHODS ARE USED TO BUILD JSMD MODELS?

An important part of malware detection techniques is malware analysis. It determines the purpose and functionality of malware so that the detection can be built. We found two major data analysis methods in the malware detection literature: static analysis and dynamic analysis [50], [51]. Another hybrid analysis method also exists in the literature.

**FIGURE 4.** Data analysis methods used in the studies.**FIGURE 5.** Statistic of detection techniques in the studies.

Based on the above categories we divided the primary studied into four different parts. As Figure 4 shows more than half of the studies (53.1%) were used static and 21.9% studies used dynamic data analysis methods. Some studies used both, static and dynamic (S&D) analysis and one study used hybrid analysis.

### F. RQ5: WHAT DETECTION TECHNIQUES ARE USED TO BUILD JSMD MODELS?

Malware detection techniques are also divided into two major categories: behavior-based and signature-based [37]. Malware detection techniques are used to detect and prevent malware in the computer system. Figure 5 shows the distribution of using detection techniques in PS. Most of the studies (84.4%) used behavior-based malware detection techniques and few of them (15.6%) used signature-based techniques.

### G. RQ6: HOW MANY JSMD STUDIES USE ML TECHNIQUE?

Nowadays, ML is a popular technique for detection and prediction based models. Malware detection using ML strategies is also an active research topic. ML approach has some common protocols to detect malware and it starts with generating a set of features. These features belong to the behavior and characteristics of malware. Further, the number of selected features is divided into training and test feature sets. Then ML classifiers (e.g., Support Vector Machine, Random Forest, Naive Bayes, Logistic Regression) are applied on the training sets to generate a malware detection model and the test set is used to validate the model. Finally, the performance of the detection models is evaluated by various performance evaluation metrics (e.g. FPR, Accuracy, Detection Rate, Precision, Recall, F-score) [38]. The first ML-based malware detection method was introduced by Schultz *et al.* [52], their



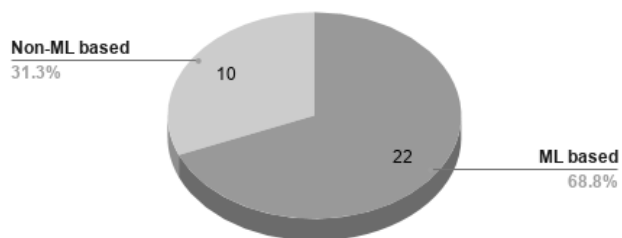


FIGURE 6. Studies using ML approach.

model was able to detect new and unknown malware [50]. Under this investigation, we found the first JSMD article by Cova *et al.* [2], which was based on ML approaches. They used a Bayesian classifier and False Negative Rate (FNR) to evaluate the classification. After that various ML-based JSMD approaches proposed by the researchers. As Figure 6 shows most of the primary articles (68.8%) used ML-based techniques to detect JS malware.

**H. RQ7: WHAT ARE THE DIFFERENT PERFORMANCE MEASURES USED FOR JSMD STUDIES?**

Performance measures are used to evaluate the performance of the detection models. In general, there are various performance measures available for evaluation [42]. These performance measures are also used in the JSMD research field to evaluate and compare the results found using various detection techniques. Table 9 is the list of the used performance measures in the selected 32 articles. The major performance measures used and their description are as follows:

- **FPR:** It is the ratio of all benign websites which are incorrectly detected as malware websites. FPR was the most often used performance measure in JSMD, considered by 14 studies. Lower FPR indicates better detection performance. The average FPR for the 14 studies is 1.51%.
- **Accuracy:** Accuracy is the ratio of the number of correctly detected benign and malware websites to the total number of websites. Accuracy was the second commonly used performance measure which was used by 13 studies. The average Accuracy of the 13 studies is 97.1%.
- **Detection Rate (DR):** Detection rate is the ratio of correctly detected malicious and benign websites to the total number of malicious and benign websites respectively [10]. 12 studies used this performance measure and their average Detection Rate is 91.8%.
- **Recall:** It is the ratio of correctly detected malicious websites to the total number of actual malicious websites. Recall is used in 11 of the selected studies and their average is 92.01%.
- **F-score:** It is the harmonic mean of Precision and Recall. It is used by nine out of 32 studies and the average score is 90.1%.

TABLE 9. Performance measures used around the studies.

Performance Metric	Studies
FPR	PS05, PS06, PS08, PS10, PS11, PS12, PS13, PS14, PS16, PS19, PS22, PS24, PS30, PS32
Accuracy	PS04, PS10, PS16, PS19, PS21, PS22, PS24, PS25, PS26, PS28, PS29, PS30, PS32
Detection Rate (DR)	PS01, PS06, PS07, PS08, PS09, PS10, PS12, PS13, PS16, PS17, PS19, PS23
Recall (True Positive Rate)	PS03, PS09, PS14, PS15, PS18, PS22, PS24, PS26, PS27, PS31, PS32
F-score (F1-score, F2-score, F-measure)	PS03, PS09, PS15, PS18, PS20, PS26, PS27, PS31, PS32
Precision	PS03, PS09, PS15, PS18, PS26, PS27, PS31, PS32
FNR	PS02, PS05, PS10, PS11, PS16, PS30
Other (Error Rate, Area under the ROC Curve (AUC))	PS19, PS27

- **Precision:** Precision is the ratio of correctly detected malicious websites to the total detected malicious websites. It is used in eight studies and their average is 92.53%.

We may figure out that the two most frequently used performance measures FPR and Accuracy indicate more accurate detection than other performance measures. This shows that FPR and Accuracy have reasonable detection capability than others.

**I. RQ8: WHAT IS THE PERFORMANCE OF PROPOSED JSMD MODELS?**

In this section, we summarised the performance taken from the PS of JSMD. To answer this question we accessed the results of all of 32 studies. It is very difficult to allow any ranking among the studies based on the performance. We found a diversity of using performance measures among them, as a result, it is difficult to compare the value of them. In the previous section, we showed the most often used performance measure was FPR, 14 studies used it out of 32; nevertheless, this figure is less than half of the total studies. From this perspective, we refrained to compare the value in an average manner. In Table 11, we represented the value of most considered and seven performance measures used in the studies. The values of the highest performer are highlighted for each performance measure. The value of various performance measures has two types, some metrics (Detection Rate, Accuracy, Precision, Recall, and F-score) which fall between 0% to 100% with a higher number indicating better detection performance and some metrics (FPR and FNR) with a lower number indicating better detection performance. PS16 achieved the highest Detection Rate and Accuracy which were 99.68% and 99.95% respectively. For Precision (99.55%) and F-score (98.37%), the highest scores were achieved by PS26. The highest Recall was 98.40% and achieved by PS24. On the other hand, the lowest score

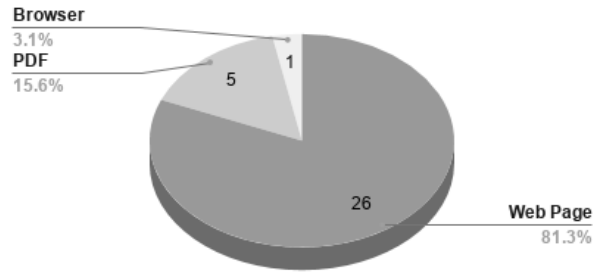
**TABLE 10.** List of top five studies according to the performance measures scores.

Primary Study	Performance	Comment
PS26: Fang 2018	Accuracy 99.51%, Precision 99.55%, Recall 97.21%, F-score 98.37%	This study used four performance metrics. Their precision and f-score of this study were highest among the participants. Besides, the scores of accuracy and recall were also high.
PS16: Wang 2015	DT 99.68%, Accuracy 99.95%, FPR 0.21%, FNR 0.84%	This study is the top scorers in term of DT and precision. The performance of FPR and FNR were also satisfactory.
PS24: Dabral 2017	Accuracy 98.44%, FPR 1.70%, Recall 98.40%	The highest recall was achieved by this study.
PS10: Krishnaveni 2012	DT 98%, Accuracy 97%, FPR 4.35%, FNR 0%	FNR was lowest, also the DT and accuracy were very high.
PS32: He 2019	Accuracy 94.75%, FPR 8.10%, Precision 98.80%, Recall 94.8 %, F-score 94.70%	The scores of accuracy, precision, recall, and f-score were high by their detection model.

for FPR was 0% and achieved by PS05, PS08, and PS14. Similarly, PS10 was the highest performer in terms of FNR. According to the scores of different performance measures, we represented the top scorer studies in Table 10. Among the five articles, PS26 shows the most stable performance in terms of their used performance metrics. After that, we placed PS16, PS24, PS10, and PS32 respectively by analyzing their performance.

**J. RQ9: HOW PERFORM ML BASED MODELS COMPARED WITH OTHERS?**

We compared the studies that were used ML techniques with other studies in terms of identifying the strength and efficiency of ML techniques in JSMD. We already mentioned the activities of using ML techniques in malware detection. We found that most of the PS (22) considered ML-based techniques. The last column of Table 11 represents the information of using ML in the PS. PS01, PS08, PS09, PS10, PS11, PS12, PS14, PS17, PS20, and PS23 are the 10 non-ML based studies. It can be observed that ML-based studies achieved the highest Detection Rate, Accuracy, Precision, Recall, and F-score as they represent the top performer studies in the Table. Only a few non-ML based studies achieved the lowest FPR and FNR. Fang *et al.* [27] used various ML techniques including Random Forest, Support Vector Machine (SVM), Naive Bayes, and Neural Network-based four classifiers. Their results show the highest Precision and F-score compared to other studies; their Accuracy and Recall value was the second-highest scorers. In addition to that, Wang *et al.* [17] used various ML-based classifiers includ-



**FIGURE 7.** Distribution of research focus considered in the PS.

ing Random Forest, Naive Bayes, and Random Tree; their detection model achieved the highest Detection Rate and Accuracy.

We may conclude that the ML-based modeling techniques outperform in terms of scores of various performance measures present in Table 11.

**K. RQ10: WHAT IS THE RESEARCH FOCUS OF THE ARTICLES?**

The target of the researchers was different among the considered articles. We have found three categories based on the research focus of the articles. Figure 7 represents the total distribution of the research focus considered in the articles. The focus of most researchers was web page JSMD, 81.3% of articles are related to this research focus. 15.6% of research studies were related to PDF-based malware detection. Browser and Application based JSMD were minor research topics with only 5.3% and 2.6% coverage respectively. It can be concluded that most of the JSMD researchers have considered websites/web pages as their research focus. The possible reason JS is the commonly used programming language in web development.

**L. RQ11: WHAT ARE THE LIMITATIONS AND CHALLENGES OF THE MALWARE DETECTION AS REPORTED IN THE STUDIES?**

The following sentences represent the discussion about the limitation of the malware detection reported in the PS:

PS11 discussed ‘Discrepancy in Browsers’, a challenge of Web Page malware detection. As they mentioned, the discrepancy in browsers refers to the trouble of using the same web page in different browsers. According to them, the discrepancy in browser implementations will affect the detection effectiveness. Different browsers have their characteristics, it can be challenging for a detection model while a web page is used in different browsers. PS16 addressed a problem related to the training data selection process. They discussed that training data collection from different origins directly affects the accuracy of trained classifiers. Besides, insufficient distribution of the benign and malicious sample in the JSMD dataset is addressed as a vital limitation for effective feature learning of malicious JS contents by PS27. PS04 discussed the imbalance learning capability of their models. They mentioned that their approach has a lack of

**TABLE 11. Results of each primary study according to the various performance measures.**

SL.	Paper	DR	Accuracy	FPR	FNR	Precision	Recall	F-score	ML
PS01	Choi 2009	66.67%							No
PS02	Cova 2010				0.20%				Yes
PS03	Likarish 2010					92%	78.70%	80.60%	Yes
PS04	Laskov 2011		85%						Yes
PS05	Curtsinger 2011			0%	9%				Yes
PS06	Schütt 2012	93.20%		0.01%					Yes
PS07	Schwenk 2012	93%							Yes
PS08	Schmitt 2012	90%		0%					No
PS09	Al-Taharwa 2012	97%				70%	ND	ND	No
PS10	Krishnaveni 2012	98%	97%	4.35%	0%				No
PS11	Xu 2013			1.75%	0.53%				No
PS12	Gorji 2014	85.07%		0.26%					No
PS13	Corona 2014	99.27%		0.05%					Yes
PS14	Lui 2014			0%			97%		No
PS15	Al-Taharwa 2014					98.85%	82.20%	ND	Yes
PS16	Wang 2015	<b>99.68%</b>	<b>99.95%</b>	0.21%	0.84%				Yes
PS17	Xue 2015	95.01%							No
PS18	Jodavi 2015					97%	91%	94%	Yes
PS19	Cosovan 2015	89.75%	99.51%	0.01%					Yes
PS20	Takamori 2015							78%	No
PS21	Su 2016		99.01%						Yes
PS22	Wang 2016		95%	4.20%			93.95%		Yes
PS23	Kim 2017	95%							No
PS24	Dabral 2017		98.44%	1.70%			<b>98.40%</b>		Yes
PS25	Morishige 2018		97.11%						Yes
PS26	Fang 2018		99.51%			<b>99.55%</b>	97.21%	<b>98.37%</b>	Yes
PS27	Ndichu 2018					89%	90%	89%	Yes
PS28	Wu 2018		98.20%						Yes
PS29	Hou 2018		98.70%						Yes
PS30	Fass 2018		99.50%	0.52%	0.54%				Yes
PS31	Ndichu 2019					95%	97%	96%	Yes
PS32	He 2019		94.75%	8.10%		98.80%	94.8%	94.70%	Yes

*ND - No specific data was found*

accurate detection capability and has difficulty with accurate discrimination between malicious and benign samples in their learning models. PS29 has a similar observation, they said preparing train data using one type of large number samples is challenging to detect a new type of samples. Updated data in the training set is necessary to get more accurate results. Some articles emphasized that any error in the API extraction process may negatively affect the accuracy of detection models because Adobe provides an extensive PDF-specific API [13]. Time delay or processing speed was discussed in several articles (e.g., PS04, PS10, PS13, PS22). Reduce the

processing time of the detection models is challenging but effective to draw attention.

The above limitations and challenges to detect malicious JS contents are reported in the studies. Future researchers can consider these points while preparing the detection models.

#### IV. THREATS TO VALIDITY

##### A. PUBLICATION BIAS

Publication bias is the tendency to publish positive results over negative results [47]. It also refers to the distorted

representation of empirical data on a subject. Data is distorted because reviewers of scientific journals tend to accept studies with positive and significant effects rather than those with negative or insignificant results. Studies with negative or insignificant results are often not submitted in the first place, they usually end up in the researcher's desk temporarily or even forever. As a result of publication bias published studies are usually not representative of all research already carried out on a subject, that means the studies' effects are overrated. In this study, one of our motives was representing the performance of different JSMD models. We represented their results only, didn't perform their models to investigate the reflection of the models on found results. We extracted and presented all accessible results of performance measures from the PS to avoid any controversy about presenting the result in this SLR. We carefully avoided delivering any perpetual conclusion about the results of the studies.

### B. STUDY SEARCHING

Our searching approach was conducted to find out the highest quantity of research articles. We deployed a wide range search string, which makes 55 studies in five databases (see section Review process for more). Whatever the figure of articles was sufficient to conduct a literature review study. Our search string performed over five databases individually in order not to miss any relevant article with the topic. We have selected 32 PS for this SLR in total. In addition to that, we used some alternative search strings which increases the chance of getting more articles on the topic. However, it is difficult to declare that we were able to include all relevant studies using the search strategy.

### C. STUDY SELECTION

Our inclusion/exclusion process was based on the title and abstract of the studies. Initially, We collected articles from different databases using the search string. The next step was the final paper selection, we performed a selection approach to list out the most relevant articles from the PS. This exclusion criterion was based on the full text of each article. After completing the selection we checked the list again. Nevertheless, there is a possibility to violate the exclusion process that a study may mistakenly exclude.

### D. DATA EXTRACTION AND QUALITY ASSESSMENT

The data extraction process was similar to the study selection. We extracted the necessary data from the selected articles. Then stored the collected data into a CSV file according to the articles and checked the data to find any miss-classification. Moreover, we discussed a common ground of understanding and resolve regarding the disagreements of any included-excluded data. End of this process the final CSV file was prepared with corrected data. Our double times' data extraction, validation, and correction process are the step toward demonstrating the validity of our study.

### V. LIMITATIONS

The goal of this study was to conduct an SLR on selected PS to evaluate the current scenario of JSMD research. A limitation in this investigation is the representation of the performance of the studies. We represented the results found among the studies but didn't show any comparison among the performance of the studies. It is also difficult because the studies used different types of datasets, modeling techniques, and performance measures; that's why reaching out to a conclusive performance comparison was difficult. We searched most five wealthy digital libraries to collect studies, but still, there are some digital libraries and journal venues that may be left out. Similarly, there still may be a possibility that a suitable study may be left out. We already mentioned this is the very first SLR in the field of JSMD research, we didn't have prior guidelines for conducting SLR in the field. As a result, our research and quality questions may be limited to evaluate and assess the PS.

### VI. CONCLUSION AND FUTURE GUIDELINES

Malicious attacks are a common threat for computer and communication systems to damage devices or steal confidential information. As usual JS-based attacks are also making trouble for the users, therefore, many techniques were proposed to detect and prevent JS malware attacks by the researchers. Based on the existing JSMD research papers, this study represents an SLR and quality analysis. Initially, we collected papers from several digital libraries followed by a search strategy. Then we have considered the most relevant 32 articles as PS, and the remaining papers have excluded that were not written exactly on the topic. We explored 11 different research and 12 quality questions to fulfill the requirement of the SLR and quality analysis respectively. The questions explained different appearances take place in the JSMD research repository which includes publication information, datasets, detection techniques, data analysis techniques, performance metrics, the performance of detection models, targeted scopes, and using ML in the studies. To answer the questionnaire, we investigated the studies from several perspectives. Our investigation began with collecting data from articles, then data pre-processing, summarizing, and visualization were performed. Firstly ten types of data attributes (see Table 3 for details) were collected from the studies where each research question belongs to one data attribute. After that, collected data were summarized and visualized to answer the respective questions. We represented the results of this SLR by answering the questions. The main findings obtained from the selected PS are:

- The demographics statistic indicates the number of journal articles was very low among the articles, it shows a lack of journal work.
- The PS were divided into four categories based on the scores of the quality questionnaire. According to the categorization, few studies achieved notable scores, but scores of most studies were average.

- Results show that most of the researchers used their privately collected datasets for their detection models. Besides, most of the studies were used large dataset, that means the number of samples in the datasets were large.
- There are only five studies that considered signature-based detection techniques to build detection models.
- There are 22 studies that perform ML-based detection techniques among 32.
- The ML techniques have acceptable detection capability for estimating JS malware. Most of the ML-based studies perform outstanding according to the results of performance measures.

Overall, we conclude that although the literature available for JSMD has a lack of high-quality work and in the overall methodology that is used to construct malware detection studies.

The following guidelines are recommended for the future researchers and whoever interested in JSMD based on the findings from our investigation:

- 1) The JS malware datasets should be made available in public and easily accessible so that researchers can be more interested to contribute to the field.
- 2) It is observed that the datasets should be balanced while constructing the training models, balanced datasets refer to equal and sufficient presence of benign and malicious samples in the datasets.
- 3) In different fields researchers are using new techniques (e.g., transfer learning [53], blockchain technology [54]) to obtain more effective and improved research output. From this perspective existing JSMD literature has a lack of using modern techniques. It is important to explore the applicability of the techniques to JS-related malware and vulnerabilities detection.
- 4) Nowadays, detection and prediction based activities become automatic, many tools are being used to automate the detection and prediction in various software engineering fields. In this investigation, the result of QQ12 shows that very few studies were involved in any tool or automation, it is essential that more studies that address easily accessible prediction tool or platform be conducted.
- 5) The results of SLR depict that most of the studies based on ML techniques performed outstandingly. Future studies should consider ML techniques including widely used Neural Networks and Deep Learning to detect JS malware.
- 6) Under this investigation, we found very few journal articles. As well as the conference proceedings, the future researcher should give attention to top-ranked journals.

## REFERENCES

- [1] Y. Choi, T. Kim, S. Choi, and C. Lee, "Automatic detection for Javascript obfuscation attacks in Web pages through string pattern analysis," in *Future Generation Information Technology* (Lecture Notes in Computer Science). Berlin, Germany: Springer, 2009, pp. 160–172.
- [2] M. Cova, C. Kruegel, and G. Vigna, "Detection and analysis of drive-by-download attacks and malicious JavaScript code," in *Proc. 19th Int. Conf. World Wide Web (WWW)*, 2010, pp. 281–290.
- [3] P. Likarish, E. Jung, and I. Jo, "Obfuscated malicious Javascript detection using classification techniques," in *Proc. 4th Int. Conf. Malicious Unwanted Softw. (MALWARE)*, Oct. 2009, pp. 47–54.
- [4] P. Laskov and N. Šrncić, "Static detection of malicious JavaScript-bearing PDF documents," in *Proc. 27th Annu. Comput. Secur. Appl. Conf. (ACSAC)*, 2011, pp. 373–382.
- [5] C. Curtsing, B. Livshits, B. G. Zorn, and C. Seifert, "ZOZZLE: Fast and precise in-browser Javascript malware detection," in *Proc. USENIX Secur. Symp.*, 2011, pp. 33–48.
- [6] K. Schütt, M. Kloft, A. Bikadorov, and K. Rieck, "Early detection of malicious behavior in JavaScript code," in *Proc. 5th ACM Workshop Secur. Artif. Intell. (AISec)*, 2012, pp. 15–24.
- [7] F. Schmitt, J. Gassen, and E. Gerhards-Padilla, "PDF scrutinizer: Detecting JavaScript-based attacks in PDF documents," in *Proc. 10th Annu. Int. Conf. Privacy, Secur. Trust*, Jul. 2012, pp. 104–111.
- [8] I. A. AL-Taharwa, H.-M. Lee, A. B. Jeng, K.-P. Wu, C.-H. Mao, T.-E. Wei, and S.-M. Chen, "RedJsod: A readable JavaScript obfuscation detector using semantic-based analysis," in *Proc. IEEE 11th Int. Conf. Trust, Secur. Privacy Comput. Commun.*, Jun. 2012, pp. 1370–1375.
- [9] G. Schwenk, A. Bikadorov, T. Krueger, and K. Rieck, "Autonomous learning for detection of JavaScript attacks," in *Proc. 5th ACM Workshop Secur. Artif. Intell. (AISec)*, 2012, pp. 93–104.
- [10] R. Krishnaveni, C. Chellappan, and R. Dhanalakshmi, "Hybrid obfuscated Javascript strength analysis system for detection of malicious Websites," in *Network and Parallel Computing* (Lecture Notes in Computer Science). Berlin, Germany: Springer, 2012, pp. 129–137.
- [11] W. Xu, F. Zhang, and S. Zhu, "JStill," in *Proc. 3rd ACM Conf. Data Appl. Secur. Privacy (CODASPY)*, 2013, pp. 117–128.
- [12] A. Gorji and M. Abadi, "Detecting obfuscated JavaScript malware using sequences of internal function calls," in *Proc. ACM Southeast Regional Conf. (ACM SE)*, 2014, pp. 1–6.
- [13] I. Corona, D. Maiorca, D. Ariu, and G. Giacinto, "Lux0R," in *Proc. Workshop Artif. Intell. Secur. Workshop (AISec)*, 2014, pp. 47–57.
- [14] D. Liu, H. Wang, and A. Stavrou, "Detecting malicious Javascript in PDF through document instrumentation," in *Proc. 44th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw.*, Jun. 2014, pp. 100–111.
- [15] B. Sayed, I. Traore, and A. Abdelhalim, "Detection and mitigation of malicious JavaScript using information flow control," in *Proc. 12th Annu. Int. Conf. Privacy, Secur. Trust*, Jul. 2014, pp. 264–273.
- [16] I. A. AL-Taharwa, H.-M. Lee, A. B. Jeng, K.-P. Wu, C.-S. Ho, and S.-M. Chen, "JSOD: JavaScript obfuscation detector," *Secur. Commun. Netw.*, vol. 8, no. 6, pp. 1092–1107, Apr. 2015.
- [17] J. Wang, Y. Xue, Y. Liu, and T. H. Tan, "JSDC," in *Proc. 10th ACM Symp. Inf., Comput. Commun. Secur. (ASIA CCS)*, 2015, pp. 109–120.
- [18] Y. Xue, J. Wang, Y. Liu, H. Xiao, J. Sun, and M. Chandramohan, "Detection and classification of malicious JavaScript via attack behavior modelling," in *Proc. Int. Symp. Softw. Test. Anal. (ISSTA)*, 2015, pp. 48–59.
- [19] M. Jodavi, M. Abadi, and E. Parhizkar, "JSObfusDetector: A binary PSO-based one-class classifier ensemble to detect obfuscated JavaScript code," in *Proc. Int. Symp. Artif. Intell. Signal Process. (AISIP)*, Mar. 2015, pp. 322–327.
- [20] D. Cosovan, R. Benchea, and D. Gavrilut, "A practical guide for detecting the Java script-based malware using hidden Markov models and linear classifiers," in *Proc. 16th Int. Symp. Symbolic Numeric Algorithms Sci. Comput.*, Sep. 2014, pp. 236–243.
- [21] J. Nicolay, V. Spruyt, and C. De Roover, "Static detection of user-specified security vulnerabilities in client-side JavaScript," in *Proc. ACM Workshop Program. Lang. Anal. Secur. (PLAS)*, 2016, pp. 3–13.
- [22] J. Su, K. Yoshioka, J. Shikata, and T. Matsumoto, "An efficient method for detecting obfuscated suspicious JavaScript based on text pattern analysis," in *Proc. ACM Int. Workshop Traffic Meas. Cybersecurity (WTMC)*, 2016, pp. 3–11.
- [23] Y. Wang, W.-D. Cai, and P.-C. Wei, "A deep learning approach for detecting malicious JavaScript code," *Secur. Commun. Netw.*, vol. 9, no. 11, pp. 1520–1534, Jul. 2016.
- [24] K. Kim, I. L. Kim, C. H. Kim, Y. Kwon, Y. Zheng, X. Zhang, and D. Xu, "J-force," in *Proc. 26th Int. Conf. World Wide Web*, 2017, pp. 897–906.
- [25] S. Dabral, A. Agarwal, M. Mahajan, and S. Kumar, "Malicious PDF files detection using structural and Javascript based features," in *Information, Communication and Computing Technology* (Communications in Computer and Information Science). Singapore: Springer, 2017, pp. 137–147.

- [26] S. Morishige, S. Haruta, H. Asahina, and I. Sasase, "Obfuscated malicious Javascript detection scheme using the feature based on divided URL," in *Proc. 23rd Asia-Pacific Conf. Commun. (APCC)*, Dec. 2017, pp. 1–6.
- [27] Y. Fang, C. Huang, L. Liu, and M. Xue, "Research on malicious JavaScript detection technology based on LSTM," *IEEE Access*, vol. 6, pp. 59118–59125, 2018.
- [28] S. Ndichu, S. Ozawa, T. Misu, and K. Okada, "A machine learning approach to malicious JavaScript detection using fixed length vector representation," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2018, pp. 1–8.
- [29] B. Hou, J. Yu, B. Liu, and Z. Cai, "JSPRE: A large-scale detection of malicious Javascript code based on pre-filter," in *Cloud Computing and Security (Lecture Notes in Computer Science)*. Cham, Switzerland: Springer, 2018, pp. 586–599.
- [30] A. Fass, R. P. Krawczyk, M. Backes, and B. Stock, "JaSt: Fully syntactic detection of malicious (obfuscated) JavaScript," in *Detection of Intrusions and Malware, and Vulnerability Assessment (Lecture Notes in Computer Science)*. Cham, Switzerland: Springer, 2018, pp. 303–325.
- [31] S. Ndichu, S. Kim, S. Ozawa, T. Misu, and K. Makishima, "A machine learning approach to detection of JavaScript-based attacks using AST features and paragraph vectors," *Appl. Soft Comput.*, vol. 84, Nov. 2019, Art. no. 105721.
- [32] X. He, L. Xu, and C. Cha, "Malicious JavaScript code detection based on hybrid analysis," in *Proc. 25th Asia-Pacific Softw. Eng. Conf. (APSEC)*, Dec. 2018, pp. 365–374.
- [33] K. Takamori, M. Iwamoto, S. Oshima, and T. Nakashima, "Detection of JavaScript of malware with un-readability using Mahalanobis-distance," in *Proc. 9th Int. Conf. Broadband Wireless Comput., Commun. Appl.*, Nov. 2014, pp. 497–502.
- [34] H. Wu and S. Qin, "Detecting obfuscated suspicious JavaScript based on collaborative training," in *Proc. IEEE 17th Int. Conf. Commun. Technol. (ICCT)*, Oct. 2017, pp. 1962–1966.
- [35] (Mar. 4, 2020). *Malware Statistics & Trends Report: AV-TEST*. Accessed: Jun. 21, 2020. [Online]. Available: <https://www.av-test.org/en/statistics/malware/>
- [36] M. Souppaya and K. Scarfone, "Guide to malware incident prevention and handling for desktops and laptops," NIST Special Publication, Gaithersburg, MD, USA, Tech. Rep., 2013, vol. 800, p. 83.
- [37] A. Souril and R. Hosseini, "A state-of-the-art survey of malware detection approaches using data mining techniques," *Hum.-Centric Comput. Inf. Sci.*, vol. 8, no. 1, p. 3, Dec. 2018.
- [38] J. Landage and M. Wankhade, "Malware and malware detection techniques: A survey," *Int. J. Eng. Res. Technol.*, vol. 2, no. 12, pp. 0181–2278, 2013.
- [39] P. Cronin, F. Ryan, and M. Coughlan, "Undertaking a literature review: A step-by-step approach," *Brit. J. Nursing*, vol. 17, no. 1, pp. 38–43, Jan. 2008.
- [40] B. Kitchenham and S. Charters, "Guidelines for performing systematic literature reviews in software engineering," Tech. Rep., 2007.
- [41] B. Kitchenham, "Procedures for performing systematic reviews," *Keele, U.K., Keele Univ.*, vol. 33, no. 2004, pp. 1–26, 2004.
- [42] R. Malhotra, "A systematic review of machine learning techniques for software fault prediction," *Appl. Soft Comput.*, vol. 27, pp. 504–518, Feb. 2015.
- [43] L. Son, N. Pritam, M. Khari, R. Kumar, P. Phuong, and P. Thong, "Empirical study of software defect prediction: A systematic mapping," *Symmetry*, vol. 11, no. 2, p. 212, Feb. 2019.
- [44] P. J. Taylor, T. Dargahi, A. Dehghantaha, R. M. Parizi, and K.-K.-R. Choo, "A systematic literature review of blockchain cyber security," *Digit. Commun. Netw.*, vol. 6, no. 2, pp. 147–156, May 2020.
- [45] L. Li, T. F. Bissyandé, M. Papadakis, S. Rasthofer, A. Bartel, D. Oceau, J. Klein, and L. Traon, "Static analysis of Android apps: A systematic literature review," *Inf. Softw. Technol.*, vol. 88, pp. 67–95, Aug. 2017.
- [46] J. Wen, S. Li, Z. Lin, Y. Hu, and C. Huang, "Systematic literature review of machine learning based software development effort estimation models," *Inf. Softw. Technol.*, vol. 54, no. 1, pp. 41–59, Jan. 2012.
- [47] D. Radjenović, M. Heričko, R. Torkar, and A. Živković, "Software fault prediction metrics: A systematic literature review," *Inf. Softw. Technol.*, vol. 55, no. 8, pp. 1397–1418, Aug. 2013.
- [48] M. Hatada, M. Akiyama, T. Matsuki, and T. Kasama, "Empowering anti-malware research in japan by sharing the MWS datasets," *J. Inf. Process.*, vol. 23, no. 5, pp. 579–588, 2015.
- [49] R. S. Wahono, "A systematic literature review of software defect prediction," *J. Softw. Eng.*, vol. 1, no. 1, pp. 1–16, 2015.
- [50] E. Gandotra, D. Bansal, and S. Sofat, "Malware analysis and classification: A survey," *J. Inf. Secur.*, vol. 5, no. 2, pp. 56–64, 2014.
- [51] T. Wuchner, A. Cislak, M. Ochoa, and A. Pretschner, "Leveraging compression-based graph mining for behavior-based malware detection," *IEEE Trans. Dependable Secure Comput.*, vol. 16, no. 1, pp. 99–112, Jan. 2019.
- [52] M. G. Schultz, E. Eskin, F. Zadok, and S. J. Stolfo, "Data mining methods for detection of new malicious executables," in *Proc. IEEE Symp. Secur. Privacy (S&P)*, May 2001, pp. 38–49.
- [53] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.
- [54] Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang, "An overview of blockchain technology: Architecture, consensus, and future trends," in *Proc. IEEE Int. Congr. Big Data (BigData Congress)*, Jun. 2017, pp. 557–564.
- [55] Y. Robiah, S. Rahayu, M. Z. Mohd, S. Shahrin, M. A. Faizal, and R. Marliza, "A new generic taxonomy on hybrid malware detection," *Int. J. Comput. Sci. Inf. Secur.*, vol. 5, no. 1, 2009.



**MD. FAHIMUZZMAN SOHAN** received the B.Sc. degree in software engineering from Daffodil International University, Bangladesh, in 2019. He has published several conference proceedings. His research interests include machine learning, data science, natural language processing, and cybersecurity.



**ANAS BASALAMAH** received the M.Sc. and Ph.D. degrees from Waseda University, Tokyo, in 2006 and 2009, respectively. He is currently an Associate Professor with the Computer Engineering Department, Umm Al Qura University. He worked as a Postdoctoral Researcher with The University of Tokyo and the University of Minnesota, in 2010 and 2011, respectively. He is the Co-Founder of the tech startups: Averos and Hazen.ai. He received the Okaz Innovator Prize of 2016. His research interests include embedded networked sensing, smart cities, ubiquitous computing, participatory, and urban sensing.

...