sustainability

MDPI

Article

Prevention and Fighting against Web Attacks through Anomaly Detection Technology. A Systematic Review

Tomás Sureda Riera [1], Juan-Ramón Bermejo Higuera [2], Javier Bermejo Higuera [2], José-Javier Martínez Herraiz [1] and Juan-Antonio Sicilia Montalvo [2,*]

1 Computer Science Department, University of Alcalá, Alcalá de Henares, 28805 Madrid, Spain; tomas.sureda@uah.es (T.S.R.); josej.martinez@uah.es (J.-J.M.H.)

2 Escuela Superior de Ingeniería y Tecnología (ESIT), Universidad Internacional de la Rioja (UNIR), Logroño, 26006 La Rioja, Spain; juanramon.bermejo@unir.net (J.-R.B.H.); javier.bermejo@unir.net (J.B.H.)

* Correspondence: juanantonio.sicilia@unir.net

Received: 25 May 2020; Accepted: 12 June 2020; Published: 17 June 2020

Abstract: Numerous techniques have been developed in order to prevent attacks on web servers. Anomaly detection techniques are based on models of normal user and application behavior, interpreting deviations from the established pattern as indications of malicious activity. In this work, a systematic review of the use of anomaly detection techniques in the prevention and detection of web attacks is undertaken; in particular, we used the standardized method of a systematic review of literature in the field of computer science, proposed by Kitchenham. This method is applied to a set of 88 papers extracted from a total of 8041 reviewed papers, which have been published in notable journals. This paper discusses the process carried out in this systematic review, as well as the results and findings obtained to identify the current state of the art of web anomaly detection.

Keywords: anomaly detection; web attacks; systematic review

1. Introduction & Motivation

Web applications have changed our way of life, allowing daily operations such as making bank transfers, booking a flight, making online purchases, etc. The detection of attacks on web applications, with the purpose of safeguarding their integrity, confidentiality and availability, has become an area of special interest.

According to Liao et al. [1], Intrusion Detection Systems (IDS) can generally be divided into three categories based on the detection principle: Signature-based Detection (SD), Anomaly-based Detection (AD) and Stateful Protocol Analysis (SPA). The characteristics of the methods are as follows:

- Signature-based detection (SD): A signature corresponds to a known pattern of attack. In order to detect possible intrusions, the patterns are compared to the captured data. Alternative names for SD are Knowledge-Based Detection or Misuse Detection.
- Anomaly-based detection (AD): An anomaly is detected when there is a deviation from usual behavior, which is represented by profiles (static or dynamic). Anomaly detection is made by comparing the normal profiles with the observed events in order to detect attacks or intrusions. AD is also called Behavior-Based Detection.
- Stateful protocol analysis (SPA): SPA relies on generic profiles for specific protocols developed by the providers. Generally, SPA network protocol models are typically based on standards of protocols from international organizations. This is also known as specification-based detection.

Current scientific literature abounds with *surveys, comparative studies and reviews* of intrusion detection using anomaly detection techniques, for example:

Jyothsna et al. [2] present an overview of the main anomaly-based technologies for network intrusion detection, along with their operational architectures, and also present a classification based on the type of processing that relates to the behavior model of the target system.

Kakavand et al. [3] provided an overview of data mining methods used by HTTP web services anomaly detection, concluding that most studies do not use public datasets that allow replication of the experiments. Those studies that do use public datasets showed high percentages of accuracy in most of the intrusion detection techniques employed, but these studies were not replicated with a different set of datasets.

Samrin and Vasumathi [4], reviewed the results of applying different anomaly detection techniques on KDD Cup 99 dataset.

None of the existing surveys, comparative studies and reviews address in depth a comprehensive review in which the techniques, results, metrics and datasets used are detailed and compared in an objective and critical manner. As the reader will see in detail in Section 4.3 , one of the biggest problems that have been detected when carrying out this systematic review lies in the fact that most of the studies reviewed do not work on public datasets that allow the validation and replication of the experimental results.

To the best of the authors' knowledge, there is currently no *systematic review* of the existing scientific literature, specifically focused on the detection and prevention of web attacks using anomaly detection techniques. Authors fully agreed with Kitchenham and Charters [5] when they state that: *"(. . .) unless a literature review is thorough and fair, it is of little scientific value. This is the main rationale for undertaking systematic reviews. A systematic review synthesises existing work in a manner that is fair and seen to be fair."*

Considering the lack of systematic reviews in this area, it has been decided to undertake a systematic review on using anomaly detection techniques in web attack detection, adopting a formal and systematic procedure for the conduction of the bibliographic review, with the definition of explicit protocols for obtaining information. This systematic review was done following the guidelines of Kitchenham et al. [5–9].

This paper makes the following contributions:

- An extensive and rigorous review of the existing scientific literature on the use of anomaly detection techniques in web attack detection.
- Identification and classification of the papers reviewed according to the types of datasets, techniques, metrics, results, etc. of each one of them.
- The results and metrics obtained by the different anomaly detection techniques studied in the papers reviewed are detailed.
- Identification of opportunities to improve research aimed at the prevention and detection of web attacks through the use of anomaly detection techniques. These opportunities include: generation of publicly available datasets that allow replication and validation of the experimental work, incorporation of metrics such as F-Score, Area Under the Curve (AUC) and Precision that allow complementing the usual metrics in this type of research, better definition of the attacks analyzed in each study since, as will be seen in Section 5, most of the studies reviewed do not detail the types of attack that are attempted to be detected.

The rest of this paper is structured as follows: Section 2 presents the background: related works and research, definitions, overview and theories on anomaly detection technologies. Section 3 presents the planning of the systematic review, the research questions, as well as the method that has been used to carry out the selection and review of the papers of interest. Section 4 presents the results obtained. In Section 5 the findings are discussed. Finally, Section 6 details the conclusions and makes recommendations for future work.

2. Background

Anomaly detection algorithms have broad applications in business, scientific, and security domains where isolating and acting on the results of outlier detection is critical. Firstly, we provide an overview of related work in the area of study, as well as some definitions and classification of anomaly detection technology:

2.1. Related Work

Patel et al. [10] worked on a systematic review of IDS in cloud environments, focusing mainly on the requirements that an Intrusion Detection and Prevention System (IDPS) should meet in order to be deployed in a cloud computing environment. This paper does not specify the use of any systematic methodology for searching bibliographic sources, nor the definition of specific protocols for obtaining information.

Raghav, Chhikara and Hasteer [11] analyzed in a systematic review the approaches of Intrusion Prevention System (IPS) in a cloud computing environment. Again, this systematic review does not indicate the use of any specific methodology for information gathering and does not pose a set of initial questions to be answered.

In 2007, Patcha and Park [12] conducted a survey of anomaly detection techniques, detailing the existing techniques at that time, but without referring to sections such as the datasets used in the studies reviewed, or the metrics used in the validation of the experiments.

In 2009, Chandola, Banerjee and Kumar [13] conducted a survey of the studies carried out on detection of anomalies in a wide range of knowledge domains. Despite being a great work, it is too general and does not include important aspects in the field of study of web attack prevention by detecting anomalies, such as the datasets used, metrics, etc.

In 2018, Jose et al. [14], provide an overview of various aspects of anomaly based host intrusion detection systems.

Fernandes et al. [15] reviewed the most important aspects pertaining to anomaly detection, covering an overview of a background analysis as well as a core study on the most relevant techniques, methods, and systems within the area. They also discussed the description of an IDS and its types.

Kwon et al. [16], investigated deep learning techniques employed for anomaly-based network intrusion detection; however, a review of the datasets is missing, as they only describe the KDD Cup 1999 and NSL-KDD datasets, the former being heavily criticized in several studies [17–20].

In 2018, Ieracitano et al. [21] propose an innovative statistical analysis driven optimized deep learning system for intrusion detection, extracting optimized and more correlated features using big data visualization and statistical analysis methods, followed by a deep autoencoder (AE) for potential threat detection. Specifically, a preprocessing module eliminates the outliers and converts categorical variables into one-hot-encoded vectors. In 2020, Ieracitano et al. [22] combine traditional data analysis and statistical techniques incorporating advances in Machine Learning (ML). Specifically, Deep Learning (DL) technology is employed in conjuction with statistical analysis. In both studies, the NSL-KDD dataset was used.

Khraisat et al. [23] presented a taxonomy of contemporary IDS, a comprehensive review of recent works and an overview of the datasets commonly used. They also presented an overview of evasion techniques used by attackers.

Ahmed, Naser Mahmood and Hu [24] provide an overview of different network anomaly detection techniques, as well as various alternative datasets to KDD Cup 1999 and NSL-KDD. However, datasets such as CSIC 2010 are not included in the work. The metrics used for the validation of the various studies reviewed are also not listed in this study.

The present work aims to remedy the weaknesses of the above-mentioned studies, through a systematic review of the available literature, strictly following the principles of a methodology widely accepted by the scientific community as proposed by Kitcheham and Charters [5], carrying

out an analysis of publicly available datasets, metrics used for the evaluation of results obtained and discussion of techniques used by the various studies reviewed.

2.2. Anomaly Detection Definition

Kotu and Deshpande [25] define anomaly detection as *"the process of finding outliers in a given dataset"*. Outliers are the data objects that stand out amongst other data objects and do not conform to the expected behavior in a dataset. An outlier is a data object that is markedly different from the other objects in a dataset. Hence, an outlier is always defined in the context of other objects in the dataset.

2.3. Types of Anomaly Detection Algorithms

It is common to divide the anomaly detection algorithms according to their purpose. The main categories are listed below: [26,27]:

- Supervised algorithms: Supervised algorithms model the relationships between input data and prediction. They try to predict the output values that will be obtained when feeding the model with new input data. This prediction is based on the relationships learned from the tagged training data. Examples of supervised algorithms are Nearest Neighbor, Naive Bayes, Decision Trees, Linear Regression, Support Vector Machines (SVM), Neural Networks.
- Unsupervised algorithms: As there is no tagged training data on which the algorithm can perform its learning, these algorithms perform pattern detection on the input data. Examples of unsupervised algorithms are association rules and k-means.
- Semi-supervised algorithms: Semi-supervised algorithms use little tagged data and a lot of untagged data as part of the training set. These algorithms try to explore the structural information contained in the unlabeled data to generate predictive models that work better than those that only use labeled data. Common examples of unsupervised algorithms are: Generative models, Low-density separation and Graph-based methods.
- Reinforcement algorithms: The objective is the development of a system (called agent) that is intended to improve its efficiency by performing a certain task based on the interaction with its environment, receiving rewards that allow it to adapt its behavior. As the agent receives rewards, it must develop the right strategy (called policy) that leads it to obtain positive rewards in all possible situations. Common examples of reinforcement algorithms are: Q-Learning, Temporal Difference (TD) and Deep Adversarial Networks.

There are different auxiliary techniques used in the process of detecting anomalies in a given dataset [28–30].

- Feature extraction: N-grams, Bag of Words, multi-features information entropy prediction model.
- Dimensionality reduction: Principal Component Analysis (PCA), Random Projection, Diffusion Maps.
- Parameter estimation: Limited-memory Broyden—Fletcher—Goldfarb—Shanno (L-BFGS) algorithm.
- Regular expression generator: Simple (SREG) and Complex (CREG) Expression Generator.

2.4. Advantages and Disadvantages of Anomaly Detection Algorithms

According to García-Teodoro et al. [31], signature-based schemes provide very good detection results for specified, well-known attacks, but they are not capable of detecting new intrusions, even if they are built as minimum variants of already known attacks. On the contrary, anomaly detection algorithms are capable of detecting previously unseen intrusion events. However, the rate of false positive (FP, events erroneously classified as attacks) in anomaly-based systems is usually higher than in signature-based ones.

3. Review Methodology

This systematic review has been conducted following the guidelines outlined by Kitchenham et al [5–9]. The methodology includes: development of a review protocol, managing the review, analysis and reporting of results, and discussion of findings.

3.1. Preparing the Review

In the generation of the review protocol, we considered the definition and details of the research questions, the bibliographic databases to be included in the search, as well as the methods used to identify and evaluate the papers susceptible to inclusion in our work. In order to carry out the review, we identified the main studies, applying the inclusion and exclusion criteria to them and synthesizing the results. In order to reduce investigator bias, the review protocol was drafted by one of the authors, reviewed by the other authors and finalized by a discussion among all the authors. The online databases were searched widely and their studies are reported. In total, the initial search returned 8041 articles.

3.2. Research Questions

The main objective of our work was to analyze and classify the available scientific literature focused on the detection of web attacks using anomaly detection techniques. For proper planning of the review, a set of research questions were generated. The method for generating the set of research questions is as follows: first, each of the authors contributes those they consider appropriate for the objective of the paper; then, all the proposals contributed are discussed, and finally, the most relevant questions are chosen by agreement of all the authors. In Table 1 we detail the research questions details.

Table 1. Research questions details.

Research question	Details
(1) **What is the current state of web anomaly detection?**	
What kind of attacks are attempted to be detected or prevented?	An overview of the types of web attacks that most concern the scientific community is achieved.
What web anomaly detection methods are used? How often are they cited in specialized literature?	An overview of the status of the different methods and techniques for detecting anomalies in web applications is achieved. The different studies that evaluate and compare the different anomaly detection techniques are explored.
Advantages and disadvantages of different methods and techniques	
(2) **Systematic reviews of anomaly detection in web applications**	
What studies and research have been done	The different studies are quantified and tabulated by each method or technique
Identification of the strengths and lacks of these works	A critical review of the studies provided is made
(3) **Key areas of interest in the different studies carried out in the detection of anomalies in web applications**	
Featured areas, number of studies carried out in each area and main findings	An objective view of the number of studies is obtained for each subarea that helps identify key areas for future research
Evolution of the number of studies carried out in each area over time	A time-based count shows how the key area has evolved over time

3.3. Information Sources

As recommended by [5–9], a wide search in electronic sources was made to increase the probability of finding relevant articles. In our case, the following databases were searched:

- IEEE Xplore (https://ieeexplore.ieee.org/Xplore/home.jsp)
- ScienceDirect (https://www.sciencedirect.com/)
- Springer (https://link.springer.com/)
- Wiley Interscience (https://onlinelibrary.wiley.com/)
- ACM Digital Library (https://dl.acm.org/dl.cfm)

3.4. Search Criteria

An exhaustive search has been carried out on the different online resources; the search sequence included the keywords *"anomaly detection" and ("web" or "network")*. The search covered the papers published in the period from January 1993 to December 2019. The search terms have been searched in the title and abstract of the publication, whenever possible. This phase returned 8,041 results. Table 2 shows the search strategy in the different online resources.

Table 2. Search queries.

Resource	Search Query	Years	Content Type	# Results
ieeexplore.ieee.org	In title and abstract: "anomaly detection" AND (web OR network)	1993–2019	Conferences, Journals, Early Access Articles, Books	4063
sciencedirect.com	In title and abstract: "anomaly detection" AND (web OR network)	1993–2019	Journals, Books	528
link.springer.com	In title: "anomaly detection" AND (web OR network)	1993–2019	Chapter, Conference Paper, Article	962
onlinelibrary.wiley.com	In title and abstract: "anomaly detection" AND (web OR network)	1993–2019	Journals, Books	31
acm.org	In abstract: "anomaly detection" AND (web OR network)	1993–2019	Articles, Proceedings	2457
			Total Results:	**8041**

3.5. Criteria for Inclusion and Exclusion of Papers

Papers were considered for inclusion in the review if their field of study was web anomaly detection. This systematic review includes only quantitative studies written in English. In a first stage, duplicate results have been detected and eliminated. A publication is considered to be duplicated if its Digital Object Identifier (DOI) is equal to the DOI of another publication. In a second phase, irrelevant documents were manually excluded based on their level of relevance. In that case, the number of irrelevant documents is significant, as research articles on the detection of anomalies in biology, medicine, social networks and study disciplines other than computer security are difficult to distinguish from the detection of web anomalies in an online database search. In a later stage, a selection based on title and abstract was done. Papers that do not contain *"web"* or *"http"* in the title or abstract were rejected. Finally, the selected documents were fully read to select a definitive list of documents according to the inclusion/exclusion criteria.

As shown in Figure 1, our search returned over 8041 total papers, which were narrowed down to 6906 after removing duplicates papers. Subsequently, based on their relevance, 1124 papers were selected, from which 189 papers were further selected based on their titles and abstracts. Then, these 189 papers were read entirely to select a final list of 106 papers based on the inclusion and exclusion criterion.

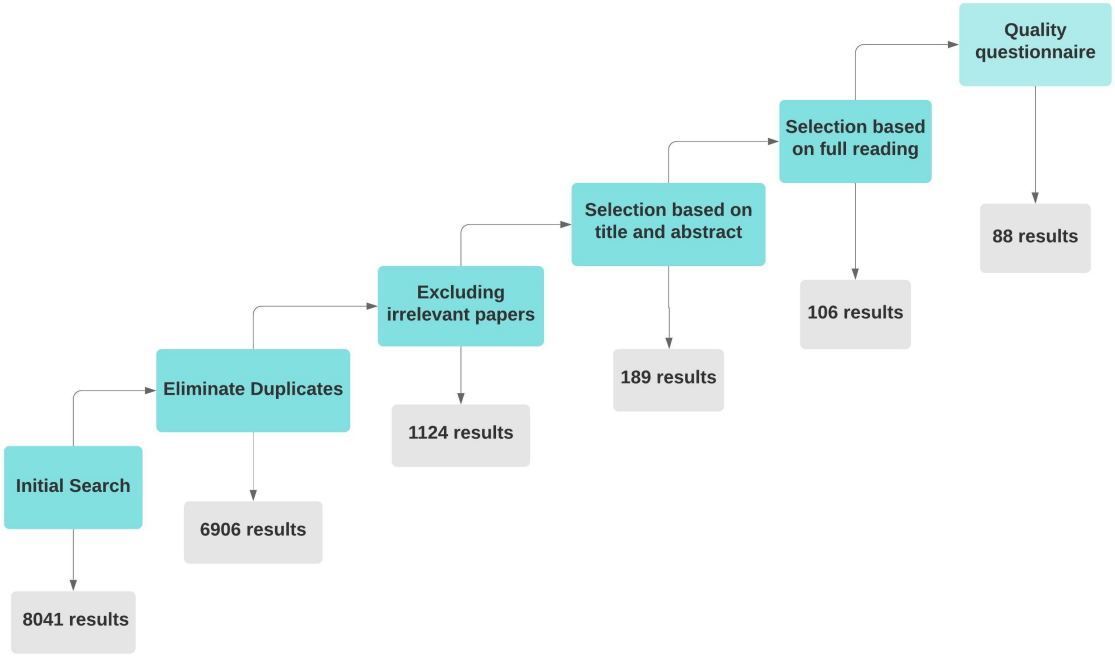

Figure 1. Papers Selection Process.

3.6. Quality Assurance

After applying the inclusion/exclusion criteria to select the relevant papers, an analysis of the quality of the remaining papers was carried out. Following the guidelines cited by [5–9], all studies were evaluated for bias, internal and external validity of results.

A questionnaire Appendix A was made to evaluate the quality of each paper to be included in the systematic review. Each team member evaluated the studies using the quality questionnaire. Only papers that passed the evaluation of the quality questionnaire by unanimity of the five team members were included in the systematic review. After evaluating the papers with the questionnaire, a total of 88 papers conform to this systematic review.

3.7. Quality Evaluation Results

In the last 5 years (2015 to 2019) the interest of the scientific community in anomaly detection techniques applied to web intrusion detection has increased; proof of this is that 67% of the studies analysed (59 out of 88 studies) are concentrated in this period. Figure 2 shows the evolution of the number of papers selected by year. Similarly, Table 3, shows the number of studies selected in each year.

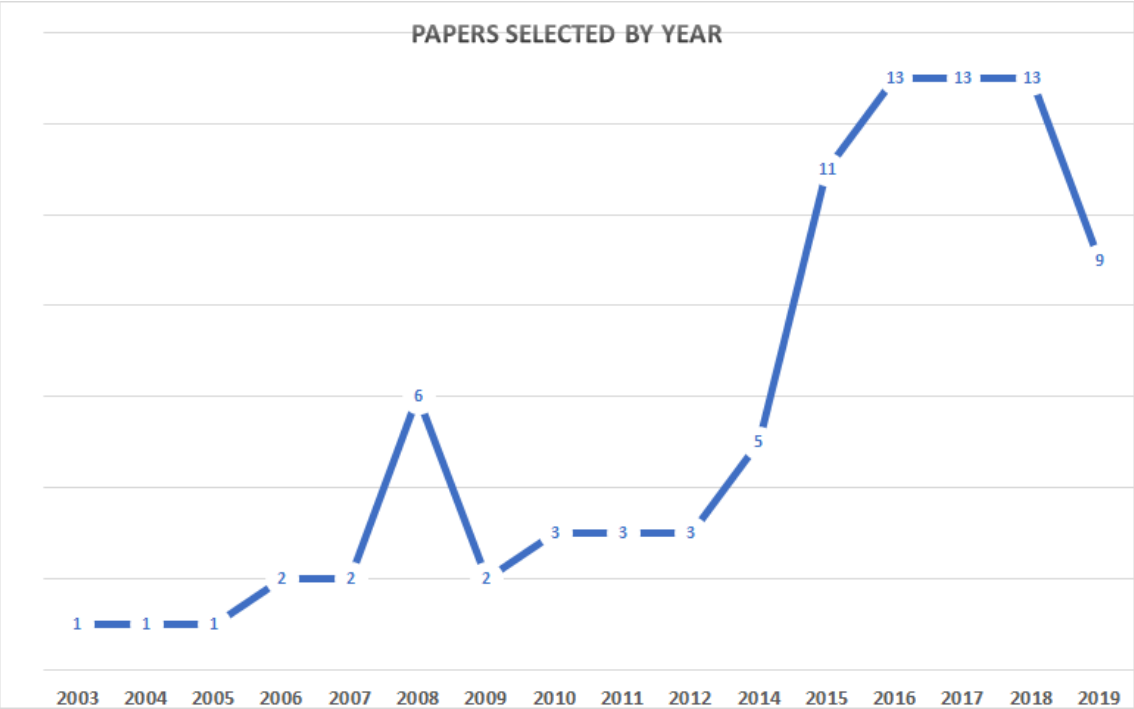

Figure 2. Papers selected by year.

Table 3. Papers selected by year.

Year	Selected by Title and Abstract	Selected after Complete Reading	Selected after Quality Questionnaire
1993	0	0	0
1994	0	0	0
1995	0	0	0
1996	0	0	0
1997	0	0	0
1998	0	0	0
1999	0	0	0
2000	0	0	0
2001	0	0	0
2002	0	0	0
2003	2	1	1
2004	1	1	1
2005	2	2	1
2006	5	4	2
2007	5	3	2
2008	11	6	6
2009	5	3	2
2010	7	4	3
2011	4	4	3
2012	6	4	3
2013	5	1	0
2014	14	5	5
2015	20	13	11
2016	25	14	13
2017	26	16	13
2018	20	14	13
2019	31	11	9
Total	**189**	**106**	**88**

3.8. Information Retrieval

The data retrieval form presented in Appendix B gives guidelines for the retrieval of data from all studies covered in this systematic literature review. It also includes details of the main study itself and the information needed to target the research questions. The entire paper has been read to collect the necessary data and the specific information has been extracted from each document: source, authors, title, year of publication and responses to the research questions.

4. Results

This study aims to investigate the available literature according to the research questions mentioned in Table 1. 29 (32.95%) of the 88 studies included in our systematic review of the literature were published in journals specializing in computer and network security, data science, etc., while 59 (67.05%) were published in leading conferences and workshops in the same or similar areas. Table 4 lists the total of studies selected by year, grouped by conferences and journals, while Figure 3 summarizes the percentage of studies published in journals and conferences.

Table 4. Studies selected by year, grouped by conferences and journals.

Year	Conferences	Journals	Total Studies
2003	1	0	1
2004	0	1	1
2005	0	1	1
2006	2	0	2
2007	2	0	2
2008	5	1	6
2009	2	0	2
2010	3	0	3
2011	1	2	3
2012	3	0	3
2014	4	1	5
2015	7	4	11
2016	10	3	13
2017	11	2	13
2018	8	5	13
2019	0	9	9
Total	**59**	**29**	**88**

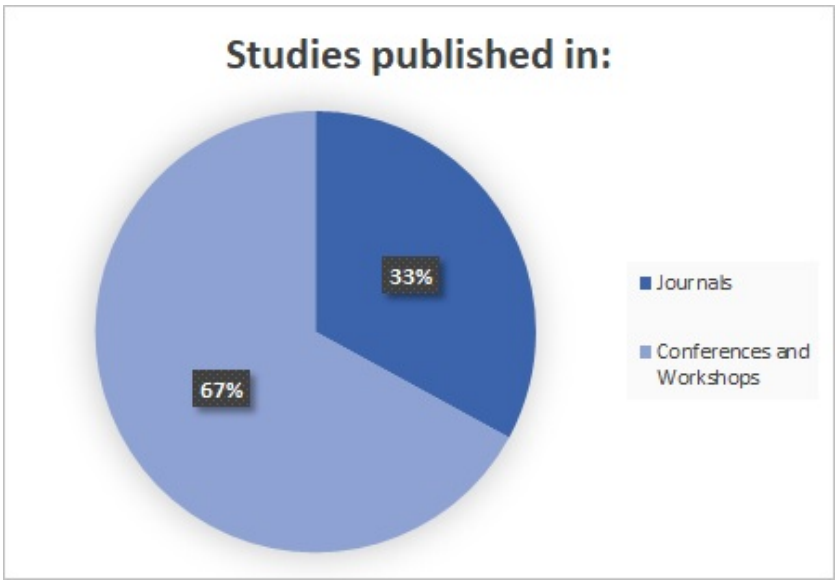

Figure 3. % of studies published in journals and conferences.

4.1. Specific Attack Detection/Prevention

In our research, we have detected that some of the studies reviewed are focused on protecting web servers against specific kinds of attacks, mainly on DDoS and Injection Attacks. A list of the most studied types of attacks is presented below; in addition, Table 5, details the specific attacks, the number of studies dealing with each particular attack, as well as a list of the corresponding citations.

Table 5. Detail of attacks.

Attack	Number of Studies	Citations
DDoS	11	[32–42]
Injection	10	[43–52]
Botnets	2	[53,54]
Defacement	2	[55,56]
Other Attacks	62	[57–118]

4.1.1. DDoS Attacks

Denial of Service (DoS) attacks are a form of attack that seeks to make a network resource unavailable due to overloading the resource or machine with an overwhelming number of packets, thereby crashing or severely slowing the performance of the resource. Distributed Denial of Service (DDoS) is a large scale DoS attack which is distributed in the Internet. In a first phase, the attacker proceeds to identify and exploit vulnerabilities in one or more networks for the installation of malware programs in multiple computers in order to obtain control of them remotely. At a later stage, these compromised computers are exploited for the mass sending of attack packets to the target(s) that will usually be located outside the original network of infected computers. These attacks occur without the knowledge of the compromised hosts [119].

Thang and Nguyen [32] proposed a framework to detect DDoS attacks; this framework was based on using an on-line scanning process to detect certain traits of DDoS attack and building a dynamic blacklist. Tripathi and Hubballi [33] proposed the use of chi-square test in order to detect slow rate denial of service attacks against HTTP/2 protocol. In [34], Najafabadi et al. proposed a detection method for the application layer DDoS attacks that worked extracting instances of user behaviors requesting resources from HTTP web server logs and using Principal Component Analysis (PCA) in order to detect anomalous behavior instances. Zolotukhin and Kokkonen [35] focused on detection of application-layer DoS attacks that utilize encrypted protocols by applying an anomaly-detection-based approach to statistics extracted from network packets headers using the stacked autoencoder algorithm. Shirani, Azgomi and Alrabaee [36] proposed the detection of DDoS attacks on Web Services using time series and applying the ARIMA model. Tripathi, Hubballi and Singh [37] used Hellinger distance between two probability distributions generated in training and testing phases to detect Slow HTTP DoS attacks. Wang et al. [38] proposed a sketch-based anomaly detection scheme for application layer DDoS attacks. The scheme utilizes the divergence of sketches in two consecutive detection cycles to detect the occurrence of an anomaly, designing a variant of Hellinger Distance to measure the divergence for mitigating the impact of network dynamics. Wang et al. [39] proposed multifeatures information entropy prediction model in order to prevent flooding App-DDoS attacks; for asymmetric attacks, a second-order Markov detection model was proposed. Xie and Tang [40] proposed a Web user browsing behavior model to detect DDoS attacks based on Hidden Markov Model. Markov states represent the click-behavior of users while hyperlink among pages is represented by different states. Lin et al. [41] proposed a new statistical model to detect DDoS attacks called Rhythm Matrix (RM), based on the packet size and the interarrival time of consecutive HTTP-request packets in a flow that indicated the users' behaviour when opening and browsing web pages. RM characterized the user access trajectory fragments distribution, including the order of visiting pages and the time spent on each page. Change-rate abnormality in the RM was used to detect DDoS attacks, and further identify the malicious hosts according to their drop points in the RM.

4.1.2. Injection Attacks

Injection flaws allow attackers to relay malicious code through an application to another system. These attacks include calls to the operating system via system calls, the use of external programs via shell commands, as well as calls to backend databases via SQL (i.e., SQL injection) [120]. SQL Injection (SQLI) constitutes an important class of attacks against web applications. By leveraging insufficient input validation, an attacker could obtain direct access to the database underlying an application [121].

Kozik, Choraś and Holubowicz [43] used token extraction of HTTP request, as well as an evolutionary-based token alignment unsupervised algorithm to detect SQLI and Cross Site Scripting (XSS) attacks. Wang et al. [44] proposed a new algorithm called FCERMining (Frequent Closed Episode Rules Mining) for mining frequently closed episode rules, dealing with big data on Spark to find the valid rules quickly. They made some experiments with the SQLMAP map tool in order to test the proposed method against SQLI attacks. Yuan et al. [45] presented a comprehensive three-step approach aimed at detecting and preventing SQLI attacks: Firstly, an ensemble clustering model is applied to separate anomalies from normal samples. In the second phase, word2vec algorithm is used to get the semantical presentations of anomalies. Finally, another multi-clustering approach clusters anomalies into specific types. Kozik, Choraś and Holubowicz [49] proposed an algorithm for SQL injection attack detection using a modified Linear Discriminant Analysis (LDA), including dimensionality reduction using Singular Value Decomposition (SVD), and an adaptation of Simulated Annealing for LDA projection vector computation.

4.1.3. Botnets Attacks

A bot is a compromised computer that can carry out the commands of its master, and bots are networked to form a botnet with a topology chosen by their master [122]. Differences botnet than other types of attacks is the existence of Command and Control (C&C) that work in giving orders from botmaster to bot. Bots always hide while looking for an unattended target, when bot find the target they will report to the botmaster [123].

Yu, Guo and Stojmenovic [53] established a four-parameter semi-Markov model to represent browsing behavior. Based on this model, they found that it was impossible to detect mimicking attacks based on statistics if the number of active bots of the attacking botnet is sufficiently large (though it is hard for botnet owners to satisfy the condition to carry out a mimicking attack most of the time). They concluded that mimicking attacks could be discriminated from genuine flash crowds using second order statistical metrics, defining a new correntropy metric. Sakib and Huang [54] proposed the detection of HTTP-based C&C traffic using statistical features based on client generated HTTP request packets and DNS server generated response packets. They applied three different anomaly detection methods: Chebyshev's Inequality, One-class Support Vector Machines (OCSVM) and Nearest Neighbor based Local Outlier Factor.

4.1.4. Defacement

In the web defacement attack the invader changes the visual appearance of the webpage. The business competitor, insurgent and extremist groups defame the reputation of the organizations and mislead public through these types of attacks, modifying the content of home page. Web defacement can be broadly categorized into Text Defacement and Image Defacement. [124].

Davanzo, Medvet and Bartoli [56] proposed a test framework for a web defacement monitoring service, working with different algorithms aimed at producing an item's (a document downloaded from a specific URL) binary classification. The algorithms evaluated were: kth nearest, Local Outlier Factor, Hotelling's T-Square, Parzen windows, Support Vector Machines and Domain Knowledge aggregator. Best results were obtained with Domain Knowledge, Support vector Machines, Parzen windows and Hotelling's T-Square. Medvet and Bartoli [55] considered the problem of corruption in

the learning data, concerning a Web Site Defacement detection system, presenting a procedure for detecting whether a learning set is corrupted.

4.1.5. Other Attacks

This group includes all those studies in which the type of attack studied is not clearly specified, either because it makes use of non-publicly accessible datasets and does not provide information on the type of attack it is trying to detect, or because it does not try to detect a specific type of attack but any web request which is considered anomalous, etc.

4.2. Current Status of Anomaly Detection

An anomaly detection process implies the use of different strategies and different techniques to achieve the final objective: clustering algorithms, classification, dimensionality reduction, use of auxiliary techniques, etc. The main algorithms and techniques detected in the different studies analyzed are detailed below.

4.2.1. Clustering Algorithms

Clustering is the task of grouping a set of objects in such a way that objects in the same group are more similar to each other than to those in other groups (clusters) [125]. Depending on the comparison of the new data received against the model generated by the clustering algorithm, it is determined whether it is an anomaly point (the way to do this varies depending on the type of clustering algorithm used, and can be based on distance or probabilities). The clustering algorithms most used are:

- **K-Means:** K-means is an unsupervised classification (clustering) algorithm that groups objects into k groups based on their characteristics. Clustering is done by minimizing the sum of distances between each object and the centroid of its group or cluster. The quadratic distance is usually used. The k-means algorithm solves an optimization problem. The function to optimize (minimize) is the sum of the quadratic distances of each object to the centroid of its cluster. [126]

  The objects are represented with real vectors of n dimensions $(x_1, x_2, \ldots, x_n)$ and the k-means algorithm constructs k groups where the sum of distances of the objects is minimized, within each group $S = \{S_1, S_2, \ldots, S_k\}$, to its centroid. The problem can be formulated as follows [126]:

$$\min_{s} E(\mu_i) = \min_{s} \sum_{i=1}^{k} \sum_{x_j \in S_i} \left\| x_j - \mu_i \right\|^2 \tag{1}$$

  where S is the set of data whose elements are the objects x_j represented by vectors, where each of its elements represent a characteristic or attribute. We will have k groups or clusters with their corresponding centroid μ_i [126].

  In each update of the centroids, from the mathematical point of view, we impose the necessary end condition on the function $E(\mu_i)$ which, for the quadratic function (1) is:

$$\frac{\partial E}{\partial \mu_i} = 0 \Rightarrow \mu_i^{(t+1)} = \frac{1}{\left| S_i^{(t)} \right|} \sum_{x_j \in S_i^{(t)}} x_j$$

  and the average of the elements of each group is taken as a new centroid.
- **Gaussian Mixture Model:** Gaussian mixture models are a probabilistic model for representing normally distributed subpopulations within an overall population. Mixture models, in general don't require knowing which subpopulation a data point belongs to, allowing the model to learn the subpopulations automatically. Since subpopulation assignment is not known, this constitutes

a form of unsupervised learning. The Gaussian Mixture function is formed by several Gaussians, individually identified by $k \in \{1, \ldots, K\}$, where K is the number of groups formed in the data set. Each Gaussian k is composed of mean μ (defines its center), Σ covariance (defines its width), a mixture probability π (defines the size of the Gaussian function). The mixing coefficients are probabilistic and must meet this condition:

$$\sum_{k=1}^{K} \pi_k = 1$$

In general, the Gaussian density function is defined by:

$$\mathcal{N}(x|\mu, \Sigma) = \frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}} exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right)$$

where x represents the data points, D is the number of dimensions of each data point. μ is the mean and Σ is the covariance.

- **Mahalanobis Distance:** The Mahalanobis Distance is a multivariate distance metric that measures the distance between a point (vector) and a distribution. The most common use for Mahalanobis Distance is to find multivariate outliers, which indicates unusual combinations of two or more variables. The formal definition is:

$$D^2 = (x-m)^T \cdot C^{-1} \cdot (x-m)$$

where D^2 is the square of the Mahalanobis Distance, x is the vector of observations, m is the vector of mean values of independents variables, C^{-1} is the inverse covariance matrix of independent variables.

- **Affinity Propagation:** Affinity Propagation does not require the number of clusters to be determined before running the algorithm. The data points can be seen as a network where all the data points send messages to all other points [127]. The subject of the messages is the determination of the points being an exemplar. The exemplars are points that explain the other data points "better" and are the most significant of their aggregation. All data points want to determine collectively the data points that are an exemplar to them. These messages are saved in two matrices:

  - *Responsibility Matrix R*. In this matrix, $r(i,k)$ reflects how well point k is adjusted to be an exemplar for point i.
  - *Availability Matrix A*. $a(i,k)$ reflects how accurate it would be for point i to select point k as an exemplar.

  Let $(x_1, x_2, \ldots, x_n)$ be a set of data points, with no internal structure assumptions, and let s be a function that measures the degree of similarity between any two points, such that $s(x_i, x_j) > s(x_i, x_k) \iff x_i$ is more similar to x_j than to x_k.

  Similarity matrix(S) gives us information about the similarity between two data points:

$$s(i,k) = -\|x_i - x_k\|^2$$

that is defined as the negative of the euclidean distance between the two instances. The greater the distance between any two instances, smaller is the similarity between them. The diagonal of s $s(i,i)$ represents the input preference, i.e., the probability that a given input will become an exemplar. When the same value is set for all entries, it controls how many classes the algorithm produces. A value close to the lowest possible similarity produces fewer classes, while a value close to or

greater than the highest possible similarity produces many classes. It is usually initialized with the median similarity of all pairs of entries.

The algorithm proceeds by alternating two message passing stages, to update the Responsibility Matrix and the Availability Matrix. Both matrices are initialized with zeros and can be viewed as log probability tables. These updates are performed on an iterative basis:

First, Responsibility updates are sent:

$$r(i,k) \leftarrow s(i,k) - \max_{k's.t.k' \neq k} \{a(i,k') + s(i,k')\}$$

Then, availability is updated per:

$$a(i,k) \leftarrow \min \left(0, r(k,k) + \sum_{i' \notin \{i,k\}} \max(0, r(i',k)) \right)$$

for $i \neq k$ and

$$a(k,k) \leftarrow \sum_{i' \neq k} \max(0, r(i',k))$$

The iterations are performed until either the cluster boundaries remain unchanged over a number of iterations, or after some predetermined number of iterations. The exemplars whose sum of *responsibility and availability* is positive are obtained: $(r(i,i) + a(i,i)) > 0$

- **DBSCAN:** Density-based spatial clustering of applications with noise (DBSCAN) [128] is a density-based clustering algorithm because it finds a number of groups (clusters) starting with a given density distribution of the corresponding nodes. Clustering happens based on two parameters: *Neighbourhood*, cutoff distance of a point from core point for it to be considered a part of a cluster. Commonly referred to as ε. *Minimum points*, minimum number of points required to form a cluster. Commonly referred to as minPts.

  There are three types of points after the DBSCAN clustering is complete: *Core*, this is a point which has at least minPts points within distance ε from itself. *Border*, this is a point which has at least one Core point at a distance ε. *Noise*, this is a point which is neither a *Core* nor a *Border*. It has less than minPts points within distance ε from itself.

  DBSCAN can be summarized in following steps: The algorithm begins with an arbitrary point that has not been visited. The neighborhood of this point is limited, and if it contains specific points, a cluster starts on it. Otherwise, the point is labeled as noise. Note that the point in question may belong to another neighborhood than the specific one in the corresponding cluster. If a point is included in the dense part of a cluster, its neighborhood is also part of the cluster. Thus, all points in that neighborhood are added to the cluster, as are the neighborhoods of these points that are sufficiently dense. This process continues until a densely connected cluster is completely built. Then, a new point not visited is visited and processed in order to discover another cluster or noise.
- **Nearest Neighbor based Local Outlier Factor:** The Local Outlier Factor (LOF) [129] is based on the concept of a local density, where the locality is given by the k-nearest neighbours. The density is estimated by the distance between close neighbours. If an object's density is compared with the densities of its neighbours, regions with similar density values and points with density values far below the values obtained by its neighbours will be identified. These points are considered outliers. The steps to calculate the LOF are detailed below:

- Calculate distance between the pair of observations.
- Find the kth nearest neighbor observation; calculate the distance between the observation and k-Nearest neighbor.
- Calculate the reachability distance between object p and o:

$$reach\text{-}dist_k(p,o) = max\{k\text{-}distance(o), d(p,o)\}$$

- Calculate Local Reachability Density (LRD): the most optimal distance in any direction from the neighbor to the individual point. The local reachability density of an object p is the inverse of the average reachability distance based on the MinPts (minimum number of objects) nearest neighbors of p.

$$LRD_{MinPts}(x) = \frac{1}{\left(\frac{\sum_{o \in N_{MinPts}(p)} reach\text{-}dist_{MinPts}(p,o)}{|N_{MinPts}(p)|}\right)}$$

- Calculate Local Outlier Factor: It is the average of the ratio of the local reachability density of p and those of $p'sMinPts\text{-}nearestneighbors$; captures the degree to which we call p an outlier.

$$LOF_{MinPts}(p) = \frac{\sum_{o \in N_{MinPts}(p)} \frac{lrd_{MinPts}(o)}{lrd_{MinPts}(p)}}{|N_{MinPts}(p)|}$$

- **Expectation–Maximization:** The Expectation–Maximization (EM) algorithm is a way to find maximum likelihood estimates for model parameters when the data is incomplete, has missing data points, or has unobserved (hidden) latent variables. It is an iterative way to approximate the maximum likelihood function [130,131]. The basics steps for the algorithm are:

  - An initial guess is made for the model's parameters and a probability distribution is created (E-step).
  - Until stability is reached, do:
    * Newly observed data is added to the model.
    * The probability distribution from the E-step is tweaked to include the new data (M-step).

  Formally:

  Given the statistical model which generates a set X of observed data, a set of unobserved latent data or missing values Z, and a vector of unknown parameters θ, along with a likelihood function $L(\theta; X, Z) = p(X, Z|\theta)$, the maximum likelihood estimate (MLE) of the unknown parameters is determined by maximizing the marginal likelihood of the observed data. [132]

$$L(\theta; X) = p(X|\theta) = \int p(X, Z\theta) dZ$$

  The EM algorithm finds the MLE by iteratively calculating the [132]:

  - Expectation Step (E-Step): Define $Q(\theta|\theta^{(t)})$ as the expected value of the log likelihood function of θ, with respect to the current conditional distribution of Z given X and the current estimates of the parameters $\theta^{(t)}$

$$Q(\theta|\theta^{(t)}) = E_{Z|X,\theta^{(t)}}[logL(\theta; X, Z)]$$

– Maximization step (M step): Find the parameters that maximize:

$$\theta^{(t+1)} = arg_\theta \max Q(\theta|\theta^{(t)})$$

4.2.2. Classification Algorithms

The idea behind the classification algorithms is very simple: it is about predicting the target class by analyzing the training dataset. The training dataset is used to get better boundary conditions which could help to determine each target class. When boundary conditions are determined, target class can be predicted. The classification of new data as anomalous or not, depends on the class in which it is classified.

All classification algorithms can be generalized as algorithms that receive a training set and learn a classification function of the form $f : R^n \rightarrow \{+1, -1\}$. This function is applied to news inputs and its value represents the class to which the input is classified [133].

The classification algorithms most used are:

- **One Class Support Vector Machine:** The problem addressed by One Class Support Vector Machine (OCSVM) is novelty detection [134]. The idea of novelty detection is to detect rare events, i.e., events that happen rarely, and hence, with very little samples. The problem is then, that the usual way of training a classifier will not work. Here the idea is to find a function that is positive for regions with high density of points, and negative for small densities.

  Consider a dataset:
  $\Omega = \{(x_1, y_2), (x_2, y_2), \ldots, (x_n, y_n)\}$; $x_i \in R^d$ in a multi-dimensional space where x_i is the i –th input data point and $y_i \in \{-1, 1\}$ is the i –th output pattern, indicating the class membership.

  SVMs can create a non-linear decision boundary by projecting the data through a non-linear function ϕ to a space with a higher dimension. This means that data points which can't be separated by a straight line in their original space I are lifted to a feature space F where there can be a straight hyperplane that separates the data points of one class from another. When that hyperplane would be projected back to the input space I, it would have the form of a non-linear curve.

  OCSVM separates all the data points from the origin (in feature space F) and maximizes the distance from this hyperplane to the origin. This results in a binary function which captures regions in the input space where the probability density of the data lives.
- **Hidden Markov Model:** A Hidden Markov Model (HMM) is a statistical model in which the system to be modeled is assumed to be a Markov process of unknown parameters. The objective is to determine the hidden parameters of that string from the observable parameters. In a normal Markov model, the state is directly visible to the observer, so the transition probabilities between states are the only parameters. In a hidden Markov model, the state is not directly visible, but only the variables influenced by the state are visible. Each state has a probability distribution over the possible output symbols. Consequently, the symbol sequence generated by an HMM provides some information about the state sequence [135].

  Formally, a HMM is a quintuple (S, V, π, A, B), characterized by the following elements [136]:

  – $S = \{S_1, S_2, \cdots, S_N\}$ is the set of states, where N is the number of states. The triplet (S, π, A) represents a Markov chain; the states are hidden and never observable directly.
  – $V = \{V_1, V_2, \cdots, V_M\}$ is the discrete set of possible symbol observations, where M represents the number of observations.

- $\pi : S \rightarrow [0,1] = \{\pi_1, \pi_2, \cdots, \pi_N\}$ is the initial probability distribution on states. It gives the probability of starting in each state. It can be expected that:

$$\sum_{s \in S} \pi(s) = \sum_{i=1}^{N} \pi_1 = 1$$

- $A = (a_{ij})_{i \in S, j \in S}$ is the transition probability of moving from state S_i to state S_j. It can be expected that $a_{ij} \in [0,1]$ for each S_i and S_j, and that $\sum i \in S^{a_{ij}} = 1$ for each S_j.
- $B = (b_{ij})_{i \in V, j \in S}$ is the emission probability that symbol v_i is seen in state S_i.

The model makes two assumptions:

- *The Markov assumption:* represents the memory of the model, so current state depends only on the previous state. Formally:

$$P(q_t | q_1^{t-1}) = P(q_t | q_{t-1})$$

- *The independence assumption:* the output observation at time t is dependent only on the current state and it is independent of previous observations and states. Formally:

$$P(o_t | o_1^{t-1}, q_1^t) = P(o_t | q_t)$$

- **K-Nearest Neighbors:** The K-Nearest Neighbors (KNN) algorithm classifies new objects according to the outcome of the closest object or the outcomes of several closest objects in the feature space of the training set [137]. An object is classified by a majority vote of its neighbors, with the new object being assigned to the class that is most common among its k nearest neighbors (k is a positive integer, and typically small). The neighbors are taken from a set of objects for which the correct classification is known. In the classification phase, k is a user-defined constant, and a new object with given features is classified by assigning to it the label that is most frequent among the k training samples nearest to that new object. With continuous features, Euclidean distance is used as distance metric, while with categorical features the Hamming distance is used. Finally, the input x gets assigned to the class with the largest probability.
- **Naive Bayes:** The Naive Bayesian classifier is based on Bayes' theorem with the independence assumptions between predictors [138–140]. Bayes theorem provides a way of calculating the posterior probability, $P(c|x)$, from $P(c)$, $P(x)$, and $P(x|c)$. Naive Bayes classifier assumes that the effect of the value of a predictor (x) on a given class (c) is independent of the values of other predictors, i.e., *class conditional independence.* Formally:

$$P(C_k|x) = \frac{p(C_k)p(x|C_k)}{p(x)}$$

where:

- $P(C_k|x)$ is the posterior probability of class given predictor.
- $p(C_k)$ is the prior probability of class.
- $p(x|C_k)$ is the likelihood which is the probability of predictor given class.
- $p(x)$ is the prior probability of predictor.

As the denominator does not depend on C and the values of the features x_i are given, the denominator is constant. The numerator is equivalent to the joint probability model $p(C_k, x_1, x_2, \ldots, x_n)$ and, using the chain rule:

$$\begin{aligned}
&p(C_k, x_1, x_2, \cdots, x_n) = \\
&= p(x_1, x_2, \cdots, x_n, C_k) \\
&= p(x_1|x_2, \cdots, x_n, C_k)p(x_2, \cdots, x_n, C_k) \\
&= p(x_1|x_2, \cdots, x_n, C_k)p(x_2|x_3, \cdots, x_n, C_k) \\
&p(x_3, \cdots, x_n, C_k) \\
&= \cdots \\
&= p(x_1|x_2, \cdots, x_n, C_k)p(x_2|x_3, \cdots, x_n, C_k) \cdots \\
&p(x_{n-1}|x_n, C_k)p(x_n|C_k)p(C_k)
\end{aligned}$$

Supposing that all features in x are independent of each other, depending on the category C_k, then:

$$p(x_i|x_{i+1}, \cdots, x_n, C_k) = p(x_i|C_k)$$

Thus, the joint model can be expressed as:

$$\begin{aligned}
&p(C_k|x_1, \cdots, x_n) \alpha p(C_k, x_1, \cdots, x_n) \\
&= p(C_k)p(x_1|C_k)p(x_2|C_k)p(x_3|C_k) \cdots \\
&= p(C_k) \prod_{i=1}^{n} p(x_1|C_k)
\end{aligned}$$

where α denotes proportionality.

4.2.3. Neural Network

A neural network is a network or circuit of neurons, or in a modern sense, an artificial neural network, composed of artificial neurons or nodes [141]. The connections between neurons are modeled as weights. A positive weight reflects an excitatory connection, while a negative weight results in an inhibitory connection. There is a *linear combination* that modifies all inputs applying the corresponding weights and proceeding to the sum of modified inputs. There is also, an activation function that controls the amplitude of the output. When the neural network receives a new anomalous data, it will have difficulty in processing it, as it is trained to process normal data, so it will generate a high mean square error (MSE).

- **Stacked Auto-encoder:** Autoencoder is a kind of unsupervised learning structure that owns three layers: input layer, hidden layer, and output layer. The structure of a stacked auto-encoder consists of several hidden layers of auto-encoders in a neural network. The output of each hidden layer is connected to the input of the next layer. The hidden layers are trained by an unsupervised algorithm and then tuned by a supervised method. Stacked autoencoder mainly consists of three steps [142]:

  - Train the first autoencoder by input data and obtain the learned feature vector.
  - The feature vector of the former layer is used as the input for the next layer, and this procedure is repeated until the training completes.
  - After all the hidden layers are trained, backpropagation algorithm (BP) [143] is used to minimize the cost function and update the weights with labeled training set to achieve fine-tuning.

- **Word2vec:** Word2vec is a two-layer neural net that processes text by transforming words into vectors. Its input is a text corpus and its output is a set of vectors: feature vectors that represent words in that corpus. Word2vec tries to group vectors of similar words together in vectorspace [144,145]. There are two different architectures capable of representing words in a distributed way: *continuous bag of words (CBOW)* and *continuous skip-gram*. Word2vec is capable of using any of them. The word prediction is done in a different way depending on the selected architecture: in the CBOW architecture, the prediction is done based on a window of context words, without being influenced by the order of those contextual words. In the skip-gram architecture, the surrounding context words are predicted from the current word, with the nearby words having more weight in the context than those that are distant.

4.2.4. Feature Selection and Extraction

Features are the specific variables that are used as input to an algorithm. Features can be selections of raw values from input data or can be values derived from that data. Feature extraction starts from an initial set of measured data and builds derived values (features) intended to be informative and non-redundant, facilitating the subsequent learning and generalization steps. In feature selection and extraction models, an anomaly is defined in terms of the redundancy present in the model, i.e., the semantic information is modelled in multiple ways. The most commonly used algorithms and techniques for selecting and extracting features are listed below.

- **N-Grams:** This is a consecutive sequence of n elements that constitute a text sample; based on a probabilistic language model, the next element in the sequence is predicted in the form of an $(n-1)$ order *Markov Model.* An N-gram model predicts x_i based on $x_{i-(n-1)}, \cdots, x_{i-1}$, i.e.,: $P(x_i|x_{i-(n-1)}, \cdots, x_{i-1})$. Formally [146,147]: Given a sequence of tokens $S = (s_1, s_2, \cdots, s_{N+(n-1)})$ over the token alphabet $\mathcal{A}$, where N and n are positive integers, an *n-gram* of the sequence S is any *n-long* subsequence of consecutive tokens. The *ithn-gram* of S is the sequence $s_i, s_{i+1}, \cdots, s_{i+n-1}$.
- **Bag Of Words:** Bag Of Words (BOW) algorithm encodes words of the text (that represent categorical features) into real-valued vectors, making a list of unique words in the text corpus called *vocabulary*. Each sentence or document can be represented as a vector with a value of 1 if the word is present in the vocabulary, or 0 otherwise. Another representation can be done by counting the number of times each word appears in the document, using the *Term Frequency-Inverse Document Frequency (TF-IDF)* technique [148,149].

  - Term Frequency (TF): $TF = TD/ND$ where TD is the number of times term t appears in a document and ND is the number of terms in the document.
  - Inverse Document Frequency (IDF): $IDF = log(N/n)$,
    where N is the number of documents and n is the number of documents a term t has appeared in. The IDF of a rare word is high, whereas the IDF of a frequent word is likely to be low.
  - Term Frequency-Inverse Document Frequency (TF-IDF): $TF\text{-}IDF = TF \cdot IDF$

4.2.5. Attribute Character Distribution

The attribute character distribution model captures the concept of a "normal" or "regular" query parameter by looking at its character distribution. The approach is based on the observation that attributes have a regular structure, are mostly human-readable, and almost always contain only printable characters. In case of attacks that send binary data, a completely different character distribution can be observed. Characters in regulars attributes are drawn using the corresponding ASCII table values [98].

- **Idealized Character Distribution:** In [98] approach, the Idealized Character Distribution (ICD) is obtained during the training phase from normal requests sent to web application. The IDC is calculated as the mean value of all character distributions. During the detection phase, the probability that the character distribution of a sequence is an actual sample drawn from its ICD is evaluated. For that purpose Chi-Square metric is used. Let $D_{chisq}(Q)$ be the Chi-Square metric for a sequence Q where N indicates the length of Q, ICD the distribution established for all the samples, σ the standard deviation from the ICD, and h the distribution of the sequence that is being tested Q, then the value of $D_{chisq}(Q)$ is computed as:

$$D_{chisq}(Q) = \sum_{n=0}^{N} \frac{1}{\sigma^2}[ICD_n - h(Q_n)]^2$$

  In [87], Kozik et al. group the characters for which the decimal value in ASCII table belongs to the followings ranges: $< 0,31 >$, $< 32,47 >$, $< 48,57 >$, $< 58,64 >$, $< 65,90 >$, $< 91,96 >$, $< 97,122 >$, $< 123,127 >$, $< 128,255 >$.

4.2.6. Dimensionality Reduction

Dimensionality reduction is the process of reducing the number of random variables under consideration by obtaining a set of principal variables [150]. In this case, an anomaly will be detected depending on the distance between a new data and the standard deviation of the training data set.

- **Principal Component Analysis:** This is the most commonly used technique of dimensionality reduction; it works by linearly reducing the existing data to a lower dimensionality space, trying to preserve the maximum variance of the data in the lower dimensionality space. [151,152]. Principal Component Analysis (PCA) is mathematically defined as an orthogonal linear transformation that transforms the data to a new coordinate system such that the greatest variance by some scalar projection of the data comes to lie on the first coordinate, the second greatest variance on the second coordinate, and so on [153]. The process of obtaining PCA from a given dataset can be summarized as:

  - Take a whole dataset of $d + 1$ dimension ignoring the label field, so the dataset becomes d *dimensional*.
  - Compute the mean of every dimension of the whole d *dimension* dataset and represent it in a matrix A.
  - Compute the covariance matrix of A. The result would be a square matrix of $d \times d$ *dimensions*.
  - Compute *Eigenvectors* and corresponding *Eigenvalues*.
  - Sort the *Eigenvectors* by decreasing *Eigenvalues* and choose k *Eigenvectors* with the largest *Eigenvalues* to form a $d \times k$ *dimensional matrix* W.
  - Transform the samples onto the new subspace.

- **Linear Discriminant Analysis:** Linear discriminant analysis (LDA) is a generalization of Fisher's linear discriminant to find a linear combination of features that characterizes or separates two or more classes of objects or events. LDA attempts to express one dependent variable as a linear combination of other features or measurements [154–156]. The goal of an LDA is to project a n *dimensional* feature space onto a smaller subspace k where $k \leq n - 1$. The process of obtaining LDA from a given dataset can be summarized as:

  - Compute the *d-dimensional* mean vectors for the different classes from the dataset.
  - Compute the scatter matrices.

  - Compute the eigenvectors $(e_1, e_2, \cdots, e_d)$ and corresponding eigenvalues $(\lambda_1, \lambda_2, \cdots, \lambda_d)$ for the scatter matrices.
  - Sort the eigenvectors by decreasing eigenvalues and choose k eigenvectors with the largest eigenvalues to form a $d \times k$ dimensional matrix W, where every column represents an eigenvector.
  - Use W matrix to transform the samples onto the new subspace.

- **Diffusion Map:** Unlike others popular dimensionality reduction techniques like PCA and LDA, Diffusion Maps are non-linear and focus on discovering the underlying manifold, i.e.,: lower-dimensional constrained "surface" upon which the data is embedded [157–159]. It achieves dimensionality reduction by re-organising data according to parameters of its underlying geometry. A diffusion map embeds data in (transforms data to) a lower-dimensional space, such that the Euclidean distance between points approximates the diffusion distance in the original feature space. The dimension of the diffusion space is determined by the geometric structure underlying the data, and the accuracy by which the diffusion distance is approximated.

4.2.7. Statistical Techniques and Probability Distribution

In addition to the algorithms and techniques described above, there are a number of statistical techniques commonly used in the studies reviewed. These techniques are listed below:

- **Chebyshev's Inequality.** Let K is any positive real number greater than 1. Chebyshev's Inequality says that *at least* $1 - \frac{1}{K^2}$ of data from a sample must fall within K standard deviations from the mean [160]. In a normal distribution, 68% of the data is one standard deviation from the mean, 95% is two standard deviations from the mean, and approximately 99% is three standard deviations from the mean. If data set is not normally distributed then the use of Chebyshev's Inequality provides a way to, knowing only the mean and standard deviation of the sample, estimate the worst scenario in which the data is distributed: i.e.,: for *any distribution, at least* 75% of the data must be between two standard deviations from the mean. For example:

  - If $\sigma = 2$, then $1 - (1/2^2) = 3/4 = 75\%$ of the data values of any distribution must be within two standard deviations of the mean.
  - If $\sigma = 3$, then $1 - (1/3^2) = 8/9 = 89\%$ of the data values of any distribution must be within three standard deviations of the mean.
  - If $\sigma = 4$, then $1 - (1/4^2) = 15/16 = 93.75\%$ of the data values of any distribution must be within four standard deviations of the mean.

- **Pearson's Chi Square Test.** Pearson's chi-squared test χ^2 is a statistical test applied to sets of categorical data to evaluate how likely it is that any observed difference between the sets arose by chance. The chi-square test belongs to the so-called goodness-of-fit or contrast tests, which aim at deciding whether the hypothesis that a given sample comes from a population with a probability distribution fully specified in the null hypothesis can be accepted, allowing the comparison of the goodness of fit, the independence of the variables and the level of homogeneity of a distribution. The comparisons are based on the comparison of observed frequencies (empirical frequencies) in the sample with those that would be expected (theoretical or expected frequencies) if the null hypothesis were true. Thus, the null hypothesis is rejected if there is a significant difference between the observed and expected frequencies. [161]. To calculate the statistic, the procedure is as follows:

  - Calculate the chi-squared test statistic.
  - Determine the degrees of freedom *(df)* of that statistic.

- Select a desired level of confidence.
- Compare χ^2 to the critical value from the chi-squared distribution with df degrees of freedom and the selected confidence level.
- The difference between the observed and expected frequencies of a distribution is evaluated using the χ^2 statistic. If the difference between observed and expected frequencies is large, the null hypothesis H_0 is false and may be rejected, i.e., this distribution does not fit the theoretical distribution. The alternative hypothesis H_1 can be accepted.

- **Bayesian Probability.** Bayesian Probability theory provides a mathematical framework for performing inference, or reasoning, using probability. It is most often used to judge the relative validity of hypotheses in the face of noisy, sparse, or uncertain data, or to adjust the parameters of a specific model [162]. The combined probability of two events, A and B, can be expressed as $P(AB) = P(A|B)P(B) = P(B|A)P(B)$. Assuming that one of the events is the hypothesis H and the other is data D, it is possible to judge the relative certainty of the hypothesis in relation to the data: According to Bayes' rule:

$$P(H|D) = \frac{P(D|H)P(H)}{P(D)}$$

Another way of interpreting Bayes' rule is by taking into account the acquired learning. That is, the transformation from $P(H)$ to $P(H|D)$ reflects the level of learning about the validity of the hypothesis from the data.

On Table 6, the different clustering algorithms found in the studies analyzed are listed. Table 7 lists the main classification algorithms. Finally, in Table 8, we detail the main auxiliary techniques used on different papers analyzed.

Table 6. Detail of clustering algorithms used.

Algorithm	Number of Studies	Citations
K-Means	7	[35,60,67,73,74,88,91]
Gaussian Mixture Model	4	[45,93,94,163]
Mahalanobis Distance	4	[82,93,102,113]
Affinity Propagation	3	[58,91,103]
DBSCAN	2	[32,35]
Convolutional Autoencoder	2	[71,115]
Nearest Neighbor based Local Outlier Factor	2	[54,56]
Expectation Maximization	2	[93,163]
Single-linkage clustering algorithm	1	[35]
Fuzzy C-Means	1	[35]
Self-Organizing Maps (SOM)	1	[35]
Deep Learning Enabled Subspace Spectral Ensemble Clustering (DEP-SSEC)	1	[45]
Subspace Spectral Ensemble Clustering (SSEC)	1	[45]
Min-Max-Tree	1	[69]
Graph-based segmentation	1	[72]
Outlier Gaussian Mixture	1	[79]
Birch	1	[88]
Mean shift	1	[88]
Subspace Weighted Ensemble Clustering	1	[94]
Query based projected clustering	1	[106]
Infinite Bounded Generalized Gaussian mixture model (InBGG)	1	[116]
Session Feature Similarity (SFAD)	1	[42]
Lambda Cut	1	[42]

Table 7. Main classification algorithms used.

Algorithm	Number of Studies	Citations
Markov Models [1]	15	[39,40,48,53,63,66,84,88,98,104,105,107,109,111,116]
One-Class Support Vector Machine (OCSVM)	11	[45,54,56,63,64,77,81,90,94,103,114]
Decision Tree [2]	9	[43,47,65,66,70,85,91,101,103]
K-Nearest Neighbors (K-NN)	5	[56,83,91,103,112]
Neural Network [3]	2	[86,89]
Random Forest	2	[75,117]
Isolation Forest	2	[59,114]
Needleman-Wunsch Algorithm	2	[72,74]

[1] Markov Models include: Hidden Markov Model, Hidden Semi - Markov Model, Semi-Markov Model, Continuous Time Markov Chains, Markov Random Field, Event—driven Hidden Semi Markov Model, Markov Chain Monte Carlo, Second Order Markov. [2] Decision Tree Models include: Reduced Error Pruning Tree (REPTree), Decision Stump, Information Gain, C4.5 decision tree, XGBoost, SPAM tree, Modified DPS Pruning. [3] Neural Network Models include: Extreme Learning Machine, C-LSTM, Convolutional Neural Network, Recurrent Neural Network, Long Short Term Memory, Deep Neural Network, Softmax Classifier.

Table 8. Detail of auxiliary techniques used.

Technique	Type	Number of Studies	Citations
N-Grams	Feature Selection	12	[57,59–61,69,77,82,90,93,102,113,118]
Bag-Of-Words (BOW)	Feature Selection	2	[48,115]
Filter Based subset Evaluation (FBSE)	Feature Selection	1	[75]
Uniformed Conditional Dynamic Mutual Information (UCDMIFS)	Feature Selection	1	[78]
Multifeatures Information Entropy Prediction Model	Feature Selection	1	[39]
Genetic Algorithm (GA)	Optimization & Searching	4	[43,81,85,97]
Principal Component Analysis (PCA)	Dimensionality Reduction	6	[34,57,60,91,113,118]
Random Projection (RP)	Dimensionality Reduction	1	[57]
Diffusion Map (DM)	Dimensionality Reduction	1	[57]
Sample Entropy	Dimensionality Reduction	1	[60]

4.3. Datasets

One of the biggest problems in conducting experiments to evaluate web anomaly detection techniques is the lack of a suitable framework to ensure the reproducibility of the experiments and the validity of the conclusions reached; one of the main components of such a framework should be one or more datasets with updated normal and attack records. One of the attempts to establish an adequate framework was the DARPA/MIT Lincoln Lab framework in 1998 and 1999 [164] and the KDD Cup dataset [165] derived from them. However, these frameworks suffer from significant limitations and have been repeatedly criticized [166]. The lack of publicly available data is explained by the data's sensitive nature: the inspection of network traffic can reveal highly sensitive information of an organization. Due to the lack of public data, researchers are forced to assemble their own datasets, generally with no access to appropriately sized networks: activity found in a small laboratory network cannot be generalized to larger scale network [167].

The public datasets that have been used in the different studies to carry out the proposed experiments are detailed below:

- **DARPA:** Created in 1998 by Lincoln Laboratory, Massachusetts Institute of Technology (MIT), promoted by DARPA and the Air Force Research Laboratory. After some time, due to the quickly changing of technology, another version of the dataset was created in 1999 (including novel attacks and a Windows NT target) [168]. DARPA98 and DARPA99 consist of raw tcpdump data, allowing testing of 244 labeled instances of 58 different attacks on four operating systems (SunOS, Solaris, Linux, and Windows NT) [164]. These datasets have been widely criticized by several academic papers [18,19], mainly due to the use of artificial data: customized software was used to synthesize typical user behavior and usual network traffic to generate a small, isolated network as if it were part of an Air Force network. According to McHugh [18], the dataset suffers from flaws in traffic data collection as there is no statistical evidence of similarity to typical Air Force network traffic (mainly with respect to the false alarm rate), attack taxonomy and distribution, and evaluation criteria. In the work of Mahoney et al. [19], numerous irregularities were found, including the fact that, due to the way the data had been generated, all malicious packets had a TTL of 126 or 253, while most normal traffic had a TTL of 127 or 254.
- **KDD Cup 99:** This is a transformed version of the DARPA dataset containing 41 features appropriate for machine learning classification algorithms. The data set can be obtained in three different versions: a complete training set, a 10% version of the training set and a test data set. Records duplication on both training and test sets can produce skewed results for more common cases [17]. Based on the works of McHugh [18] and Mahoney and Chan [19], the archival authority of Irvine KDD Archive, University of California, discourage the use of DARPA and KDD Cup 99 data sets [20].
- **NSL-KDD:** Created in 2009 by the Information Security Center of Excellence (ISCX), University of New Brunswick (UNB), in order to solve the problem of duplicate records found in KDD Cup 99 [17]; this duplication of records could cause biased results by the learning algorithms, as well as the lack of learning of infrequent records. After applying the cleanup operations, the recordset of 4,900,000 and 2,000,000 in the KDD Cup 99 training and test data set was reduced to 125,973 and 22,544 in the new NSL-KDD training and test data sets, respectively.
- **UNSW-NB15:** This dataset was created on by the IXIA PerfectStorm tool in the Cyber Range Lab of the Australian Centre for Cyber Security (ACCS) for generating a hybrid of real modern normal activities and synthetic contemporary attack behaviours; has nine types of attacks, namely, Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode and Worms [169,170]. The number of records is 175,341 and 82,332 in the training and testing sets respectively. The simulation period was 16 h on 22 January 2015 and 15 h on 17 February 2015 [171].
- **Kyoto 2006:** Both KDD Cup '99 dataset and NSL-KDD dataset do not reflect real data flow in computer network since they are generated by simulation over the virtual network. The Kyoto 2006+ data set is built on real three year-traffic data from November 2006 to August 2009. This data set is captured using honeypots, darknet sensors, e-mail server and web crawler [172].
- **ISCX:** Shiravi et al. [173] devised a systematic approach to be able to generate datasets to analyse and evaluate intrusion detection systems, mainly through the use of anomaly detection techniques. It is intended that researchers will be able to generate datasets from a set of profiles that can be combined to create a diverse set of dataset. From this work, the ISCX (Information Security Centre of Excellence) dataset emerged. This dataset consists of simulated traffic for one week, each record consists of 11 different features. The dataset is labeled, containing a description of the legitimate network traffic and attacks.
- **CSIC-2010:** The CSIC 2010 dataset contains the generated traffic targeted to an e-commerce Web application developed at Spanish Research National Council (CSIC) . In this web application, users can buy items using a shopping cart and register by providing some personal information. The dataset was generated automatically and contains 36,000 normal requests and more than 25,000 anomalous requests; the requests are labeled as normal or anomalous [174].

- **ECML/PKDD 2007:** As part of the 18th European Conference on Machine Learning (ECML) and the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD)—ECML/PKDD 2007 Discovery Challenge, a dataset was provided containing 35,006 requests classified as normal traffic and 15,110 requests classified as attacks. The dataset was generated by collecting real traffic which was then processed to mask parameter names and values, replacing them with random values [175].

Table 9 summarizes the public datasets that have been used in the analyzed works while Figure 4 provides an overview of the percentage of datasets used in the different studies reviewed.

Table 9. Public datasets used.

Dataset	Number of Studies	Citations
CSIC 2010	13	[43,59,64,65,72,74,80,85–87,93,114,117]
KDD-Cup 99	8	[58,78,80,81,91,97,106,116]
DARPA	4	[77,82,100,102]
NSL-KDD	2	[75,80]
UNSW-NB15	2	[75,115]
Kyoto 2006	2	[80,116]
ISCX	1	[116]
ECML/PKDD 2007	1	[114]

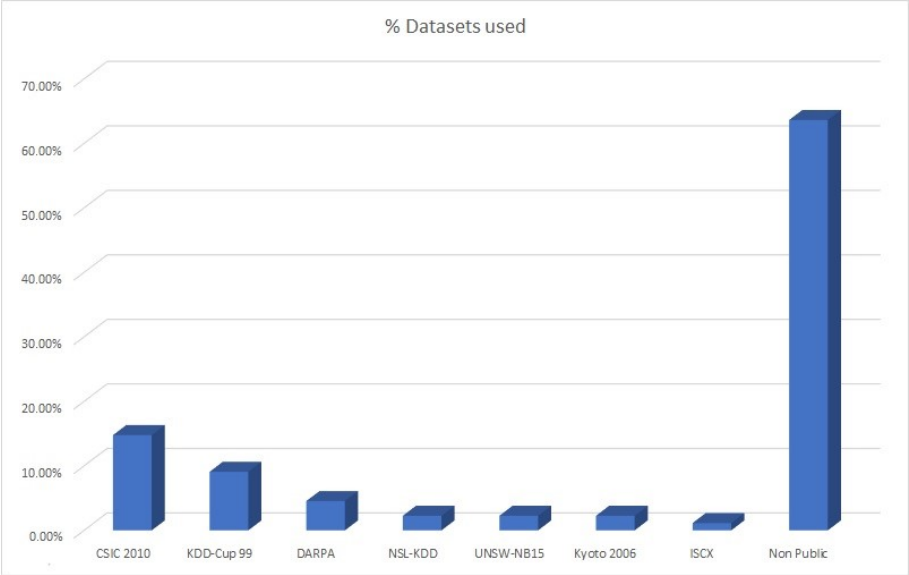

Figure 4. % of datasets used.

4.4. Metrics

This section details the most commonly used metrics to evaluate the different experiments carried out in the works that have been reviewed. A summary is given in Table 10.

- **Accuracy:** Accuracy (ACC) is ratio of payloads correctly identified divided by the total generated payloads.

$$ACC = \frac{TP + TN}{TP + TN + FP + FN}$$

- **False Alarm Rate, False Positive Rate:** False Alarm Rate (FAR) or False Positive Rate (FPR), it's the probability that a false alarm will be raised: that a positive result will be given when the true value is negative.

$$FAR = \frac{FP}{FP + TN}$$

- **True Negative Rate, Specificity:** True Negative Rate (TNR) or Specificity measures the proportion of actual negatives that are correctly identified.

$$TNR = \frac{TN}{FP + TN}$$

- **True Positive Rate, Recall, Sensitivity, Detection Rate:** True Positive Rate (TPR), Recall, Sensitivity or Detection Rate (DR) measures the proportion of actual positives that are correctly identified.

$$TPR = \frac{TP}{TP + FN}$$

- **Precision, Positive Predictive Value:** Precision or Positive Predictive Value (PPV) is the ratio of the number of malicious payloads correctly detected divided by the number of total malicious payloads.

$$PPV = \frac{TP}{TP + FP}$$

- **False Negative Rate:** False Negative Rate (FNR) is the proportion of positives that yield negative test outcomes with the test, i.e., the conditional probability of a negative test result given that the condition being looked for is present.

$$FNR = \frac{FN}{FN + TP}$$

- **F1-Score:** F1 score is a measure of test's accuracy, considering both the precision and recall. The F1-Score, also called F-measure or F-Score, is the weighted harmonic mean of two measures: precision (P) and recall (R) [176–178] .

$$F1 - Score = \frac{1}{\alpha \cdot \frac{1}{P} + (1 - \alpha) \cdot \frac{1}{R}}$$

- **Classification error:** Classification error (CE) depends on the number of samples incorrectly classified (false positives plus false negatives) and is evaluated by the formula:

$$CE = \frac{f}{n} \cdot 100$$

where f is the number of sample cases incorrectly classified, and n is the total number of sample cases.

- **Matthews Correlation Coefficient:** The Matthews Correlation Coefficient (MCC) is used in machine learning as a measure of the quality of binary (two-class) classifications [179]. MCC is a correlation coefficient between the observed and predicted binary classifications; it returns a value between -1 and +1. A coefficient of +1 represents a perfect prediction, 0 no better than

random prediction and −1 indicates total disagreement between prediction and observation.It's equivalent to *phi coefficient*.

$$MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

- **Area Under the Curve:** A Receiver Operating Characteristic Curve or ROC curve is a graph that shows the performance of a classification model at all classification thresholds, representing the true positive rate (TPR) against the false positive rate (FPR). The AUC measures the entire two-dimensional area below the full ROC curve, ranging from 0 (model with 100% incorrect predictions) to 1 (model with 100% correct predictions), representing degrees of separability and thus indicating a model's ability to distinguish between classes. [180].

Table 10. Metrics used to evaluate experiments.

Metric	Number of Studies	Citations
FPR / FAR	61	[32,33,35,36,38–46,48–50,52,55,56,58,60,62–67,69,71,72,74,75,79,81–83,85–87,91,94–102,104,105,107–113,116–118]
TPR/Recall/ Sensitivity/DR	49	[33,35–42,45–50,52,53,60,65–67,69–72,74–76,79–83,85–87,89,91,93,94,97,98,100–102,105–115,117,118]
Accuracy	24	[35,36,41,47,48,50,53,59,60,64,66,75,80,81,88,89,92,99,100,103,105,113,114,116]
F-Score	12	[47,59,62,65,71,76,80,89,113–115,118]
Precision	11	[41,43,59,65,66,76,80,89,113–115]
ROC/AUC	10	[34,50,55,71,72,77,83,90,107,115]
FNR	4	[55,56,95,100]
TNR/Specificity	3	[59,93,114]
CE	1	[163]
MCC	1	[71]

5. Discussion

This section details the different findings after careful review of the different studies evaluated.

- **Attacks:** 11 (12.5%) of the 88 studies reviewed focus on prevention and mitigation of SQLI attacks. The same amount and percentage applies to prevention and mitigation of denial of service (DDoS) attacks. Both types of attacks are included as high risk in various attack classification systems such as OWASP Top Ten Project [181], Common Attack Pattern Enumeration and Classification(CAPEC) [182], Common Weakness Enumeration (CWE) [183], OWASP Automated Threat Handbook Web Applications [184], etc. Only 2 of the studies reviewed target the detection of Botnet attacks (2.27%) and two more focus on the detection of Defacement attacks (2.27%). Also, the type of attack is not clearly specified in the vast majority of studies reviewed: 62 out of 88 (70.45%). Figure 5 presents an overview of the percentage of studies focused on the prevention or detection of specific attacks.

  Due to the wide variety of web attacks that currently exist, it seems necessary to make efforts aimed at specifying clearly and in detail the types of attacks studied in the new papers that will be published from now on. In addition, it is suggested to use recognized resources (e.g., OWASP Top Ten, CAPEC, etc.) to be able to determine the types of attacks with the highest prevalence and thus be able to focus on their study and detection.

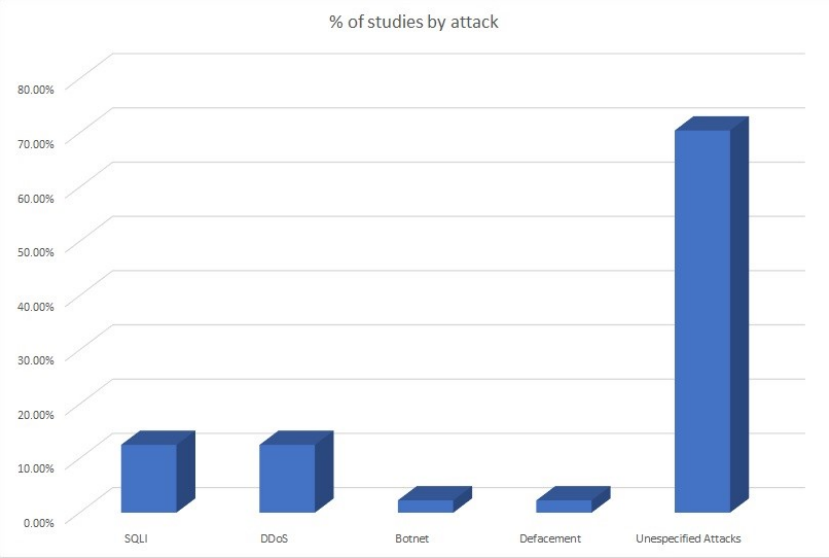

Figure 5. % of studies grouped by attack.

- **Algorithms and techniques used:**
  - CLUSTERING ALGORITHMS: 27 (30.68%) of the 88 studies reviewed used some kind of clustering algorithm. 7 (25.93%) of these 27 studies used more than one type of clustering algorithm: for example, Zolotukhin et al. [35] used 5 clustering algorithms, namely: K-Means, DBSCAN, single-linkage clustering, Self-Organizing Map (SOM), Fuzzy C-Means. The K-Means clustering algorithm is the most used: it is included in 7 of the 27 studies, although in 3 of them it is combined with other clustering algorithms. Gaussian Mixture Model (GMM) is used in 4 different studies, always in combination with other clustering algorithms: Yuan et al. [45] combine GMM with deep learning enabled subspace spectral ensemble clustering (DEP-SSEC) and Subspace Spectral Ensemble Clustering (SSEC). In [163], Kozik, Choraś and Holubowicz combine GMM with Expectation Maximisation (EM). Betarte et al. [93] combine GMM with Mahalanobis Distance and Expectation Maximisation (EM). In [94], Lin et al. combine GMM with Subspace Weighted Ensemble Clustering (SWEC).
  - CLASSIFICATION ALGORITHMS: 47 (53.41%) of the 88 studies reviewed propose some kind of classification algorithm (including those that propose the use of neural networks). Markov-type models represent a 31.91% (15 of 47 studies): Hidden Markov Model (HMM), Semi-Markov Model, Continuous Time Markov Chains, Markov Chain Monte Carlo, Markov Random Field, etc. In 13 (27.66%) of the 47 studies reviewed, the use of decision tree algorithms is proposed, Random Forest and Isolation Forest: Reduced Error Pruning Tree (REPTree), Decision Stump (DS), Isolation Forest, Information Gain (IG), XGBoost, etc. The use of Support Vector Machine (SVM) is proposed in 11 (23.40%) of the 47 studies, although 9 (19.15%) of them are of the One-Class Support Vector Machine (OCSVM) type, while the K-Nearest Neighbors (K-NN) algorithm is proposed in 5 (10.64%) of the 47 studies. Combining different classification algorithms is common: Wang and Zhang [103] introduced Information Gain based attribute selection, and then K-NN and OCSVM were used to detect anomalies. Zhang, Lu and Xu [63] propose a multi-model approach: First, the web request is partitioned into 7 fields: method, web resource, HTTP version, headers and headers inputs values are inspected by a probability distribution model, attribute sequence is inspected by HMM and attribute value is inspected by OCSVM. If one of the algorithms detects the request as anomalous, it is classified as anomalous. Kim and Cho [89] proposed a C-LSTM neural network to model the spatial and temporal information contained in traffic data. More complex characteristics are extracted from the data, combining Convolutional

Neural Network (CNN) and Recurrent Neural Network (RNN)—used to reduce frequency variation in the spatial information—, Long Short Term Memory (LSTM)—reduces temporal variation—and a Deep Neural Network (DNN)—used as a classifier, mapping the data in a more defined space. The detection of anomalies is done through the use of a SOFTMAX function.

- FEATURE SELECTION AND EXTRACTION: 17 of the 88 studies reviewed propose some kind of classification Feature Selection algorithm. 12 (70.59%) of these 17 studies are based on the N-grams algorithm for selection and feature extraction. 2 (11.76%) are based on the Bag of Words (BOW) algorithm. Vartouni, Kashi and Teshnehlab [59] propose an anomaly detection model in which the characteristics of web requests are extracted using an n-grams model based on character distribution. Subsequently, a dimensionality reduction is performed by means of SAE and finally, a classification task is performed by means of Isolation Forest. Zolotukhin et al. [60] propose an anomaly detection model in which n-grams are applied to extract the characteristics of web requests. Sample Entropy is used to capture the degree of dispersion or concentration of the web request parameter distribution in a given time interval. Unsupervised models are then used to generate a normal pattern. Using PCA, a dimensionality reduction is performed. Once PCA is applied, SVDD is applied to the transformed vectors of the training set, to build a hypersphere containing all the data in one category. To detect anomalies in the parameters of a web request, k-means is used. The detection of intrusions inside the user agent is carried out by DBSCAN. Asselin, Aguilar-Melchor and Jakllari [61] propose the use of n-grams to extract all the bi-grams from the URLs of a web server and generate a harmonic average of the probabilities that a given bi-gram is part of a normal request. New requests are classified as normal or abnormal depending on the probability of their bi-grams. Ren et al. [48] propose a model in which by means of BOW the extraction of characteristics of the web requests is carried out. Then, the detection of anomalies is done by HMM.
- DIMENSIONALITY REDUCTION: Only in 7 (7.95%) of the studies reviewed is some kind of dimensionality reduction technique applied. Principal Component Analysis (PCA) is applied in 6 of these 7 studies although in one of them it is combined with Random Projection (RP) and Diffusion Map (DM) and, in another one, it is combined with Sample Entropy. The remaining study applies Linear Discriminant Analysis (LDA) as a dimensionality reduction technique. Juvonen, Sipola and Hämäläinen [57] present the results from three methods that can be used for web log dimensionality reduction in order in order to allow the analysis and detection of anomalies: random projection, principal component analysis and diffusion maps. Based on the experimental results, they propose that RP and DM should be used together. RP methodology is efficient for daily analysis of huge amounts of traffic, while DM produces better visualizations and more accurate analysis of smaller amounts of data when needed. PCA falls in between of the other methods, but does not seem to offer any major advantages. In [49], Kozik, Choraś and Holubowicz estimated LDA transformation vector with Simulated Annealing approach in order to increase the effectiveness of SQL injection attack detection.

The use of dimensionality reduction techniques is recommended as an area for improvement in new research work. As previously mentioned, less than 8% of the papers reviewed incorporate this type of techniques, which allow the reduction of the number of characteristics of complex data samples, thus generating simpler models that avoid their overfitting to the training data and, therefore, a low performance with real data.

- **Datasets:** Only 26 (29.55%) of the 88 papers include public datasets in their experiments. The rest (70.45%) include private, synthetic or unspecified datasets, making it impossible to replicate the experiments and verify the results. Of the 26 studies that include public datasets, 12 of them

include datasets that belong to the DARPA and KDD families, which have been widely criticized in several studies [17–20].

The creation of public datasets, including new types of attacks, appears to be a major area to which new research efforts should be directed. Of considerable concern is the fact that the results obtained by 70% of the papers reviewed cannot be audited, replicated or validated because there is no access to the data on which these results are based. The availability of public data endorsed by the scientific community would allow further progress in the prevention of web attacks while avoiding problems associated with data privacy.

- **Metrics:** While most of the studies reviewed use FPR (61 of 88), TPR/Recall/Sensitivity/DR (49 of 88) and Accuracy (24 of 88), and which represent 69.32%, 55.68%, and 27.27% respectively, only 12 of them use F-Score (13.64%), 11 use precision (12.5%), and 10 use ROC/AUC (11.36%) as metrics to evaluate the experiments performed with the different algorithms analyzed. This fact is noteworthy because F-Score, accuracy and ROC/AUC are widely used metrics in work related to vulnerability detection, tool comparison, etc. [185].

The comparison of the results of the different techniques applied in the studies reviewed is difficult since there is no single public dataset widely accepted by the scientific community that allows for the replication of results, with most of the studies reviewed being based on non-publicly accessible datasets; because of this, the authors have decided to analyse the most representative results of the studies that are based on publicly accessible datasets.

Nguyen, Nguyen and Vu [77] combine an n-gram model for feature extraction and OCSVM for anomaly detection, based on the DARPA dataset, obtaining an AUC of 0.91425 for generic attacks, an AUC of 0.9912 for Shell-Code attacks and an AUC of 0.9831 for Traditional Polymorphic Attacks (CLET). Jamdagni et al. [102] propose an anomaly detection model based on Geometrical Structure Anomaly Detection (GSAD): this is a pattern recognition technique used in image processing. GSAD analyzes the correlations between various payload characteristics, using Mahalanobis Distance Map (MDM) to calculate the difference between normal and abnormal traffic, based on the DARPA dataset. The results obtained are 100% True Positive Rate (TPR) and 0.087% False Positive rate (FPR). Angiulli, Argento and Furfaro [82] identify anomalous packets in the DARPA dataset, by dividing the payload into segments of equal length, using n-grams to learn the byte sequences that usually appear in each chunk. Using a semi-supervised approach, a model is built that associates the protocol-packet length pair. This model is used to classify incoming traffic. Anomaly detection is carried out using Mahalanobis distance to determine whether a sequence of n-grams is unusual or not. A TPR of 100% and an FPR of 0.588% are obtained, but only for FTP traffic.

Wang et al. [58] use the Affinity Propagation (AP) algorithm by which they learn a subject's behavior through dynamic clustering of the data flow. It automatically tags the data and adapts to normal changes in behavior while detecting anomalies. Its study is based on the KDD-Cup 99 dataset and obtains a TPR of 98.4% and an FPR of 1.01%. Kaur and Bansal [81] implement the genetic algorithm (GA) to form clusters of normal and abnormal traffic in the training data set. The resulting clusters are used to generate normal and abnormal data partitions in the test data set. Finally, a classification of attack types is performed using SVM. The study is based on the KDD-Cup 99 dataset and obtains a TPR of 99.46%, an FPR of 1.96% and an Accuracy of 92.09%. Li et al. [97] propose an anomaly detection system based on Transductive Confidence Machines for K-NN (TCM-KNN). GA is used to reduce the size of the training dataset for the TCM-KNN model. Based on the KDD-Cup 99 dataset, it obtains a TPR of 99.38% and an FPR of 3.87%, complementing the TCM-KNN model with CHC (A genetic Algorithm called heterogeneous recombination and cataclysmic mutation used as search strategy).

Kamarudin et al. [75] propose a web anomaly detection model based on two stages: pre-processing and data mining. Pre-processing adopts the Hybrid Feature Selection (HFS) technique. In this phase, the characteristics are extracted. In the data mining phase, the classification is done by the LogitBoost algorithm. This study is based on the NSL-KDD and UNSW-NB15 datasets, obtaining a TPR of 89.75%, a FPR of 8.22% and a 90.33% of Accuracy in the NSL-KDD dataset; in the UNSW-NB15 dataset, the results obtained are as follows TPR: 99.1%, FPR: 0.18%, Accuracy: 99.45%.

Moustafa, Misra and Slay [79] propose a 4-step methodology: 1- Collect attack data by crawling websites and represent this data as feature vectors, 2- Extract features by association rules (ARM), 3- Use the extracted features to simulate web attacks, 4- Use a new Outlier Gaussian Mixture (OGM) algorithm to detect attacks. The data is obtained from the UNSW-NB15 dataset and from a non-publicly accessible dataset; a TPR of 95.56% and an FPR of 4.43% are obtained with the UNSW-NB15 dataset data. The metrics obtained with the data from the private dataset are TPR of 97.28% and FPR of 2.72%.

Alhakami et al. [116] propose a model in which patterns of activities (both normal and abnormal) are learned through Bayesian-based MCMC inference for infinite bounded generalized Gaussian mixture models. Unlike classical clustering methods, this approach does not need to specify the number of clusters, takes into account uncertainty by introducing prior knowledge for the parameters of the model, and allows to solve problems related to over and under evaluation. To obtain better cluster performance, weights of characteristics, model parameters and number of clusters are estimated simultaneously and automatically. The model evaluation data come from the KDD-Cup 99, Kyoto 2006 and ISCX datasets. The following metrics are obtained: Accuracy 83.49%, 87.41%, 90.4% FPR: 16.84%, 14.24%, 9.79% in the KDD-Cup 99, Kyoto 2006 and ISCX datasets respectively.

Alrawashdeh and Purdy [80] propose a rapid activation Adaptive Linear Function (ALF) that increases the speed of convergence and the accuracy of deep learning networks in real-time applications. Accuracy levels are as follows: 96.57%, 98.59%, 99.96% and 98.4% in the CSIC-2010, KDD-Cup 99, NSL-KDD and Kyoto 2006 datasets respectively.

Vartouni, Kashi and Teshnehlab [59] propose a model in which the characteristics of the web requests are extracted through an n-grams model based on character distribution. Subsequently, a dimensionality reduction is performed by means of SAE and finally a classification task is carried out by means of Isolation Forest. The evaluation data come from the dataset CSIC-2010; the metrics obtained are TPR: 88.34%, Precision: 80.29%, Accuracy: 88.32%, Specificity: 88.32%, F-Score: 84.12%.

Parhizkar and Abadi [64] propose a model in which, in the training phase, an initial set of One-Class Support Vector Machine is extracted (OCSVM) on the basis of legitimate requests. This initial set of OCSVM is then pruned using the ABC BeeSnips algorithm, with the aim of finding a "quasi-optimal" subset. In the detection phase, the outputs of the OCSVM present in the subset are combined to classify a new request as normal or abnormal. Based on the CSIC-2010 dataset, the metrics obtained are as follows: TPR: 95.9%, FPR: 2.82%, Accuracy: 96.54%.

Betartte et al. [93] try to improve the detection of ModSecurity through two approaches: the first, by building a One-Class Classification model. The second, by using N-grams for anomaly detection. In the One-Class Classification model, Gaussian Mixture Model (GMM) is used as the probability density distribution of the training data. Expectation Maximisation (EM) is used to estimate GMM parameters and the number of clusters. In the use of n-grams, Mahalanobis distance is used to detect the anomalies. The model evaluation data come from the CSIC-2010 dataset; for N-grams = 3, it obtains a TPR of 96.1% and a specificity of 99.5%.

Moradi Vartouni, Teshnehlab and Sedighian Kashi [114] propose a model that uses stacked auto-encoder (SAE) and deep belief network as feature learning methods in which only normal data is used in the learning phase. Subsequently, OCSVM, Isolation Forest and Elliptic Envelope are used as classifiers. The model is based on the datasets CSIC-2010 and ECML/PKDD 2007; the best values obtained are DR: 89.48%, Specificity: 89.11%, F-Score: 85.35% in the CSIC-2010 dataset and TPR: 89.75%, Specificity: 78.25%, F-Score: 84.93% in the ECML/PKDD 2007 dataset.

Kozik and Choraś [65] propose a model in which, firstly, an extraction of the tokens and the data between them is performed, to later transform them into a characteristics vector by distributing characters in different ranges of the ASCII table, thus achieving a reduction in dimensionality. The classification is carried out by means of a set of One Class Classifiers, specifically Decision Stump (DS) and RepTree. The data come from the dataset CSIC-2010 and the results obtained are TPR: 98.3%, FPR: 1.9%, Precision: 94.6%, F-Score: 96.4%.

Choraś and Kozik [72] propose a model in which they use a graphical approach to build a set of regular expressions to model HTTP requests. An attempt is made to group similar web requests and represent them using a single pattern. The measure of dissimilarity between components is done by the Needleman-Wunsch algorithm. The data come from the dataset CSIC-2010, obtaining a TPR of 94.46% and an FPR of 4.5%.

Kozik et al. [87] present a model in which a character pattern extraction method is used in web requests, based on the ICD (Ideal Character Distribution) proposed by Kruegel [186], but applied to different ranges of the ASCII table in order to reduce the dimensionality. The data come from the dataset CSIC-2010 and the results obtained are TPR: 86.6%, FPR: 1.8% for 1% of the dataset (300 samples), TPR: 95.6%, FPR: 5.8% for 10% of the dataset (3000 samples), TPR: 96.9%, FPR: 6.8% for 20% of the dataset (6000 samples), TPR: 97.7%, FPR: 8.1% for 100% of the dataset (32,400 samples).

Kozik, Choraś and Hołubowicz [43] propose a model in which the genetic algorithm is used to determine the valid subset of extracted tokens and their correct order. The data between tokens is classified by assigning the distribution of characters to different intervals in the ASCII table. RepTree and DS are used as classifiers. The data come from the extended CSIC-2010 dataset *(CSIC-2010+)* using new data samples collected during penetration tests carried out in a web-based Geographic Information System (GIS)-system. The total number of records from the original dataset was increased by 3.6%, adding 2.08% of normal samples (around 1500 requests) as well as 7.9% of new attacks (around 2000 anomalous requests). This results in a Precision level of 98%.

In [74], Kozik, Choraś and Hołubowicz propose the processing of web requests in order to extract vectors of constant length characteristics. Then k-means is applied in order to group web requests of similar characteristics. Finally, through multiple sequence alignment (MSA), an analysis of the request structure is performed. The data come from the dataset CSIC-2010+, obtaining a TPR of 92.7% and a FPR of 6.4%.

Kozik and Choraś [85] propose a model in which a genetic algorithm (GA) is used to realign HTTP payload sequences and extract their structure. The number of characters falling into different ranges of the ASCII table is counted. A hybrid classification technique combining REPTree and AdaBoost is used. The data come from the dataset CSIC-2010+, obtaining a TPR of 91.5% and an FPR of 0.7%.

In [86], Kozik et al. propose a model in which, after extracting the characteristics of web requests by distributing characters that fall within a given range of the ASCII table, Extreme Learning Machine is applied to classify the data as normal or abnormal. The data come from the dataset CSIC-2010+, obtaining a TPR of 94.98% and an FPR of 0.79%.

In [117], Kozik and Choraś propose a modified version of the Random Forest algorithm as a classifier. The study uses data from the dataset CSIC-2010+ and the results obtained are TPR: 93.5% and FPR: 0.5%.

Please note that the results of studies using data from the DARPA and KDD-Cup 99 datasets should be taken with caution, as these datasets have been widely criticized by the scientific community [17–20].

Table 11 provides a summary of the results of the different studies based on public datasets.

The authors recommend selecting the appropriate result validation metrics based on the type of vulnerability scenario to be protected (Business-Critical Applications, Heightened-Critical Applications, Best Effort, Minimum Effort) as recommended in [185].

Table 11. Metrics by study and dataset.

Study	TPR	FPR	Precision	Accuracy	Specificity	F-Score	AUC	Dataset
Nguyen, Nguyen and Vu [77]							91.42% [1] 99.12% [2] 98.31% [3]	DARPA
Angiulli, Argento and Furfaro [82]	100% [4]	0.59% [4]						DARPA
Jamdagni et al. [102]	100%	0.087%						DARPA
Wang et al. [58]	98.4%	1.01%						KDD-Cup 99
Kaur and Bansal [81]	99.46%	1.96%						KDD-Cup 99
Li et al. [97]	99.38%	3.87%						KDD-Cup 99
Kamarudin et al. [75]	89.75% 99.1%	8.22% 0.18%		90.33% 99.45%				NSL-KDD UNSW - NB15
Moustafa, Misra and Slay [79]	95.56% 97.28%	4.43% 2.72%						UNSW - NB15 Private Dataset
Alhakami et al. [116]		16.84% 14.24% 9.79%		83.49% 87.41% 90.4%				KDD-Cup 99 Kyoto 2006 ISCX
Alrawashdeh and Purdy [80]				96.57% 98.59% 99.96% 98.4%				CSIC 2010 KDD-Cup 99 NSL-KDD Kyoto 2006
Vartouni, Kashi and Teshnehlab [59]	88.34%		80.29%	88.32%	88.32%	84.12%		CSIC 2010
Parhizkar and Abadi [64]	95.9%	2.82%		96.54%				CSIC 2010
Betartte et al. [93]	96.1% [5]				99.5% [5]			CSIC 2010
Moradi Vartouni, Teshnehlab and Sedighian Kashi [114]	89.48% 89.75%				89.11% 78.25%	85.35% 84.93%		CSIC 2010 ECML/PKDD 2007
Koziz and Choraś [65]	98.3%	1.9%	94.6%			96.4%		CSIC 2010
Choraś and Kozik [72]	94.46%	4.5%						CSIC 2010
Kozik et al. [87]	86.6% [6] 95.6% [7] 96.9% [8] 97.7% [9]	1.8% [6] 5.8% [7] 6.8% [8] 8.1% [9]						CSIC 2010
Kozik, Choraś and Hołubowicz [43]			98%					CSIC 2010 +
Kozik, Choraś and Hołubowicz [74]	92.7%	6.4%						CSIC 2010 +
Kozik and Choraś [85]	91.5%	0.7%						CSIC 2010 +
Kozik et al. [86]	94.98%	0.79%						CSIC 2010 +
Kozik and Choraś [117]	93.5%	0.5%						CSIC 2010 +

[1] For generic attacks. [2] For Shell Code. [3] For CLET. [4] Only for FTP traffic. [5] For N-grams = 3. [6] 1% of dataset (300 samples). [7] 10% of dataset (3000 samples). [8] 20% of dataset (6000 samples). [9] 100% of dataset (32,400 samples).

6. Conclusions

In this work, a systematic review of the available literature on the detection of web attacks using anomaly detection techniques has been carried out, following the guidelines provided by Kitchenham et al. [5–9]. One of the major drawbacks detected in this systematic review is the unavailability of a standardized, updated and correctly labeled dataset, which allows the verification of the experimental results obtained in the different studies. It is worrying that only 29.55% of the experimental results obtained in the studies reviewed are based on public datasets; of these, approximately 50% are based on datasets that are strongly criticized by the scientific community, so it seems clear that more research efforts are needed to allow for the creation and validation of a public dataset, maintained by the scientific community, that incorporates a sufficient number of normal and abnormal requests, so as to allow for the replication and validation of studies conducted on that dataset.

A small number of studies in which dimensionality reduction techniques are applied have also been detected. Dimensionality reduction allows the analysis of a larger amount of data in a shorter period of time, simplifying the complexity of sample spaces with many dimensions while preserving their information. If PCA is used, the use of robust methods is recommended, as the PCA method is highly sensitive to outliers. If an observation has an anomaly in one of its variables, the variance in this direction will be artificially high. Since PCA tries to identify the directions with the highest variance, the subspace created will have been over-guided in this direction.

Most of the studies reviewed apply grouping algorithms using K-means and GMM classification with Markov and SVM type models. A combination of two or more clustering and/or classification algorithms is common.

A reduced use of classic metrics is detected in works related to vulnerability detection such as F-Score, accuracy and ROC/AUC; however metrics such as FPR, DR/Accuracy and TPR are widely used. Although these last metrics may be valid, the authors believe that more research efforts should be made in this area, in order to establish a concrete methodology that facilitates the choice of particular metrics depending on the type of study being conducted.

In the review of the studies carried out, it was found that most of them do not clearly specify the type of attack they are trying to prevent, although there is a small number that investigate DDoS, injection, botnets and defacement attacks. Further research efforts may be needed to generate studies that investigate the prevention and detection of other types of attacks.

In general, high statistical performances are observed in the different papers that incorporate deep learning techniques, but in those that report on the datasets used, it is detected that the results depend largely on the datasets, with Accuracy's percentages decreasing as the dataset becomes more recent. Therefore, in addition to encouraging the generation and use of public datasets that allow the replication and validation of experiments as indicated above, it should be further analyzed whether deep learning really improves intrusion detection systems.

Author Contributions: Conceptualization, T.S.R., J.-R.B.H., J.B.H. and J.-A.S.M.; methodology, T.S.R., J.-R.B.H., J.B.H. and J.-A.S.M.; investigation, T.S.R.; resources, T.S.R.; writing—original draft preparation, T.S.R.; validation, J.-R.B.H., J.B.H., J.-J.M.H. and J.-A.S.M. writing—review and editing, T.S.R., J.-R.B.H., J.B.H., J.-J.M.H. and J.-A.S.M.; visualization, J.-R.B.H., J.B.H., J.-J.M.H. and J.-A.S.M.; supervision, J.-R.B.H., J.B.H., J.-J.M.H. and J.-A.S.M.; project administration, J.-J.M.H.; funding acquisition, J.-J.M.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Appendix Quality Assurance Form

Appendix A.1. Section 1

Does the study refer to the use of anomaly detection to improve security and prevent attacks in web environments?

- This section includes: intrusion detection, DDoS attacks, IDS/IPS improvement, WAF, RASP, detection and prevention of any attack classified in OWASP Top Ten.
- The study should focus on the detection of anomalies using different techniques: probability distribution models, Markov models, one-class SVM, clustering models, data minig and, in general, any automatic learning technique.

The first section sets out the basis for the initial selection of the study. If the answer to this section is positive, we will move on to the next section. Otherwise, the study will not be included in the systematic review.

Appendix A.2. Section 2

Is the study quantitative?

- Check whether the study provides any kind of empirical measure that allows the objective interpretation of the data obtained.

Only quantitative studies are to be included in the systematic review, so if the answer to this section is yes, we will move on to the next section. Otherwise, the study will not be included in the systematic review.

Appendix A.3. Section 3

Please indicate for each of the following questions the score among the following possible values: (1)—Yes , (−1)—No, (0.5)—Partially, (0)—Not applicable.

1. Are the research objectives clearly defined?
2. Is the choice of techniques used correctly justified?
3. Are the techniques used correctly detailed?
4. Is the measurement of the variables involved correct?
5. Are data collection methods correctly described?
6. Are objective indicators applied for the analysis of the results obtained?
7. Are the objective indicators applied adequate, correctly described and justified?
8. Are all research objectives adequately addressed?
9. Are negative results specified?
10. Are problems of validity or trust of the results obtained adequately discussed?
11. Is there a clearly defined relationship between objectives, data obtained, interpretation and conclusions?

This section details the quality assessment questions for the study itself. Each of the questions has 4 possible answers with an associated score: Yes (1 point), No (−1 point), Partially (0.5 points), Not Applicable (0 points). Each study can obtain from 0 to 11 points (if the score obtained by a study is negative, it will be left at 0); the second quartile (11/2 = 5.5) is taken as the cut-off point, in such a way that those studies whose score is lower than 5.5 points will not be included in the systematic review.

Appendix B. Appendix Data Retrieval Form

Data Item	Description
Study identifier	Unique ID for the study
Date of data retrieval	
Bibliographic data	Title of the study, name(s) of the author(s), publication year
Type of study	Journal, conference, workshop, etc.
Study objective	What is the principal objective of the study, what research areas the study is focused on
Study classification	Case study, experiment, survey, comparative analysis, etc.
Type of anomaly detection technique used	
What statistical indexes are used and its values	Values of the statistical indexes used for the objective validation of the results of the study experiments
Type of dataset	It refers specifically to the type of dataset used for the development of the experiment or the validation of the proposed model. (Publicly available, from private institution, synthetically generated, etc.)
Study findings	Major findings or conclusions from the study

References

1. Liao, H.J.; Richard Lin, C.H.; Lin, Y.C.; Tung, K.Y. Intrusion detection system: A comprehensive review. *J. Netw. Comput. Appl.* **2013**, *36*, 16–24. [CrossRef]
2. Jyothsna, V. A Review of Anomaly based Intrusion Detection Systems. *Int. J. Comput. Appl.* **2011**, *28*, 26–35. [CrossRef]
3. Kakavand, M.; Mustapha, N.; Mustapha, A.; Abdullah, M.T.; Riahi, H. A Survey of Anomaly Detection Using Data Mining Methods for Hypertext Transfer Protocol Web Services. *JCS* **2015**, *11*, 89–97. [CrossRef]
4. Samrin, R.; Vasumathi, D. Review on anomaly based network intrusion detection system. In Proceedings of the 2017 International Conference on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT), Mysuru, India, 15–16 December 2017; pp. 141–147. [CrossRef]
5. Kitchenham, B.; Charters, S. *Guidelines for Performing Systematic Literature Reviews in Software Engineering Version 2.3*; Technical Report; Keele University: Keele, UK; University of Durham: Durham, UK, 2007.
6. Brereton, P.; Kitchenham, B.A.; Budgen, D.; Turner, M.; Khalil, M. Lessons from applying the systematic literature review process within the software engineering domain. *J. Syst. Softw.* **2007**, *80*, 571–583. [CrossRef]
7. Budgen, D.; Brereton, P. Performing Systematic Literature Reviews in Software Engineering. In Proceedings of the 28th International Conference on Software Engineering, Shanghai, China, 20–28 December 2006; Association for Computing Machinery: New York, NY, USA; pp. 1051–1052. [CrossRef]
8. Kitchenham, B.; Pearl Brereton, O.; Budgen, D.; Turner, M.; Bailey, J.; Linkman, S. Systematic literature reviews in software engineering—A systematic literature review; *Inf. Softw. Technol.* **2009**, *51*, 7–15. [CrossRef]
9. Kitchenham, B.; Brereton, P. A Systematic Review of Systematic Review Process Research in Software Engineering. *Manuscr. Publ. Inf. Softw. Technol.* **2013**, *55*, 2049–2075. [CrossRef]
10. Patel, A.; Taghavi, M.; Bakhtiyari, K.; Celestino Júnior, J. An intrusion detection and prevention system in cloud computing: A systematic review. *J. Netw. Comput. Appl.* **2013**, *36*, 25–41. [CrossRef]
11. Raghav, I.; Chhikara, S.; Hasteer, N. Article: Intrusion Detection and Prevention in Cloud Environment: A Systematic Review. *Int. J. Comput. Appl.* **2013**, *68*, 7–11.
12. Patcha, A.; Park, J.M. An overview of anomaly detection techniques: Existing solutions and latest technological trends. *Comput. Netw.* **2007**, *51*, 3448–3470. [CrossRef]
13. Chandola, V.; Banerjee, A.; Kumar, V. Anomaly Detection: A Survey. *ACM Comput. Surv.* **2009**, *41*. [CrossRef]

14. Jose, S.; Malathi, D.; Reddy, B.; Jayaseeli, D. A Survey on Anomaly Based Host Intrusion Detection System. *J. Phys. Conf. Ser.* **2018**. [CrossRef]
15. Fernandes, G.; Rodrigues, J.J.P.C.; Carvalho, L.F.; Al-Muhtadi, J.F.; Proença, M.L. A comprehensive survey on network anomaly detection. *Telecommun. Syst.* **2019**, *70*, 447–489. [CrossRef]
16. Kwon, D.; Kim, H.; Kim, J.; Suh, S.C.; Kim, I.; Kim, K.J. A survey of deep learning-based network anomaly detection. *Clust. Comput.* **2019**, *22*, 949–961. [CrossRef]
17. Tavallaee, M.; Bagheri, E.; Lu, W.; Ghorbani, A.A. A Detailed Analysis of the KDD CUP 99 Data Set. In Proceedings of the Second IEEE International Conference on Computational Intelligence for Security and Defense Applications, Ottawa, ON, Canada, 8–10 July 2009; IEEE Press: Piscataway, NJ, USA, 2009; pp. 53–58.
18. McHugh, J. Testing Intrusion Detection Systems: A Critique of the 1998 and 1999 DARPA Intrusion Detection System Evaluations as Performed by Lincoln Laboratory. *ACM Trans. Inf. Syst. Secur.* **2000**, *3*, 262–294. [CrossRef]
19. Mahoney, M.V.; Chan, P.K. An Analysis of the 1999 DARPA/Lincoln Laboratory Evaluation Data for Network Anomaly Detection BT—Recent Advances in Intrusion Detection. In *Recent Advances in Intrusion Detection*; Vigna, G., Kruegel, C., Jonsson, E., Eds.; Springer: Berlin/Heidelberg, Germany, 2003; pp. 220–237.
20. Brugger, T. KDD Cup '99 dataset (Network Intrusion) considered harmful. *KDnuggets News* **2007**, *7*, 15.
21. Ieracitano, C.; Adeel, A.; Gogate, M.; Dashtipour, K.; Morabito, F.C.; Larijani, H.; Raza, A.; Hussain, A. Statistical Analysis Driven Optimized Deep Learning System for Intrusion Detection BT. In *Advances in Brain Inspired Cognitive Systems*; Ren, J., Hussain, A., Zheng, J., Liu, C.L., Luo, B., Zhao, H., Zhao, X., Eds.; Springer International Publishing: Cham, Switzerland, 2018; pp. 759–769.
22. Ieracitano, C.; Adeel, A.; Morabito, F.C.; Hussain, A. A novel statistical analysis and autoencoder driven intelligent intrusion detection approach. *Neurocomputing* **2020**, *387*, 51–62. [CrossRef]
23. Khraisat, A.; Gondal, I.; Vamplew, P.; Kamruzzaman, J. Survey of intrusion detection systems: Techniques, datasets and challenges. *Cybersecurity* **2019**, *2*, 20. [CrossRef]
24. Ahmed, M.; Naser Mahmood, A.; Hu, J. A survey of network anomaly detection techniques. *J. Netw. Comput. Appl.* **2016**, *60*, 19–31. [CrossRef]
25. Kotu, V.; Deshpande, B. Chapter 13—Anomaly Detection. In *Data Science*, 2nd ed.; Kotu, V., Deshpande, B., Eds.; Morgan Kaufmann: Burlington, MA, USA, 2019; pp. 447–465. [CrossRef]
26. Hodge, V.J.; Austin, J. A Survey of Outlier Detection Methodologies. *Artif. Intell. Rev.* **2004**, *22*, 85–126. [CrossRef]
27. Kaelbling, L.P.; Littman, M.L.; Moore, A.W. Reinforcement learning: A survey. *J. Artif. Intell. Res.* **1996**, *4*, 237–285. [CrossRef]
28. Guyon, I.; Elisseeff, A. An Introduction to Variable and Feature Selection. *J. Mach. Learn. Res.* **2003**, *3*, 1157–1182.
29. Pudil, P.; Novovičová, J. Novel Methods for Feature Subset Selection with Respect to Problem Knowledge BT—Feature Extraction, Construction and Selection: A Data Mining Perspective. In *Feature Extraction, Construction and Selection. The Springer International Series in Engineering and Computer Science*; Liu, H., Motoda, H., Eds.; Springer: Boston, MA, USA, 1998; Volume 453; pp. 101–116._7. [CrossRef]
30. Hu, H.; Zahorian, S.A. Dimensionality reduction methods for HMM phonetic recognition. In Proceedings of the 2010 IEEE International Conference on Acoustics, Speech and Signal Processing, Dallas, TX, USA, 14–19 March 2010; pp. 4854–4857. [CrossRef]
31. García-Teodoro, P.; Díaz-Verdejo, J.; Maciá-Fernández, G.; Vázquez, E. Anomaly-based network intrusion detection: Techniques, systems and challenges. *Comput. Secur.* **2009**, *28*, 18–28. [CrossRef]
32. Thang, T.M.; Nguyen, K.V. FDDA: A Framework For Fast Detecting Source Attack In Web Application DDoS Attack. In Proceedings of the Eighth International Symposium on Information and Communication Technology, Nha Trang, Vietnam, 7–8 December 2017; Association for Computing Machinery: New York, NY, USA, 2017; SoICT 2017; pp. 278–285. [CrossRef]
33. Tripathi, N.; Hubballi, N. Slow Rate Denial of Service Attacks against HTTP/2 and Detection. *Comput. Secur.* **2018**, *72*, 255–272. [CrossRef]
34. Najafabadi, M.M.; Khoshgoftaar, T.M.; Calvert, C.; Kemp, C. User Behavior Anomaly Detection for Application Layer DDoS Attacks. In Proceedings of the 2017 IEEE International Conference on Information Reuse and Integration (IRI), San Diego, CA, USA, 4–6 August 2017; pp. 154–161. [CrossRef]

35. Zolotukhin, M.; Hämäläinen, T.; Kokkonen, T.; Siltanen, J. Increasing web service availability by detecting application-layer DDoS attacks in encrypted traffic. In Proceedings of the 2016 23rd International Conference on Telecommunications (ICT), Thessaloniki, Greece, 16–18 May 2016; pp. 1–6. [CrossRef]
36. Shirani, P.; Azgomi, M.A.; Alrabaee, S. A method for intrusion detection in web services based on time series. In Proceedings of the 2015 IEEE 28th Canadian Conference on Electrical and Computer Engineering (CCECE), Halifax, NS, Canada, 3–6 May 2015; pp. 836–841. [CrossRef]
37. Tripathi, N.; Hubballi, N.; Singh, Y. How Secure are Web Servers? An Empirical Study of Slow HTTP DoS Attacks and Detection. In Proceedings of the 2016 11th International Conference on Availability, Reliability and Security (ARES), Salzburg, Austria, 31 August–2 September 2016; pp. 454–463. [CrossRef]
38. Wang, C.; Miu, T.T.N.; Luo, X.; Wang, J. SkyShield: A Sketch-Based Defense System Against Application Layer DDoS Attacks. *IEEE Trans. Inf. Forensics Secur.* **2018**, *13*, 559–573. [CrossRef]
39. Wang, Y.; Liu, L.; Si, C.; Sun, B. A novel approach for countering application layer DDoS attacks. In Proceedings of the 2017 IEEE 2nd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), Chongqing, China, 25–26 March 2017; pp. 1814–1817. [CrossRef]
40. Xie, Y.; Tang, S. Online Anomaly Detection Based on Web Usage Mining. In Proceedings of the 2012 IEEE 26th International Parallel and Distributed Processing Symposium Workshops PhD Forum, Shanghai, China, 21–25 May 2012; pp. 1177–1182. [CrossRef]
41. Lin, H.; Cao, S.; Wu, J.; Cao, Z.; Wang, F. Identifying Application-Layer DDoS Attacks Based on Request Rhythm Matrices. *IEEE Access* **2019**, *7*, 164480–164491. [CrossRef]
42. Xiao, R.; Su, J.; Du, X.; Jiang, J.; Lin, X.; Lin, L. SFAD: Toward effective anomaly detection based on session feature similarity. *Knowl.-Based Syst.* **2019**, *165*, 149–156. [CrossRef]
43. Kozik, R.; Choraś, M.; Hołubowicz, W. Evolutionary-based packets classification for anomaly detection in web layer. *Secur. Commun. Netw.* **2016**, *9*, 2901–2910. [CrossRef]
44. Wang, L.; Cao, S.; Wan, L.; Wang, F. Web Anomaly Detection Based on Frequent Closed Episode Rules. In Proceedings of the 2017 IEEE Trustcom/BigDataSE/ICESS, Sydney, NSW, Australia, 1–4 August 2017; pp. 967–972. [CrossRef]
45. Yuan, G.; Li, B.; Yao, Y.; Zhang, S. A deep learning enabled subspace spectral ensemble clustering approach for web anomaly detection. In Proceedings of the 2017 International Joint Conference on Neural Networks (IJCNN), Anchorage, AK, USA, 14–19 May 2017; pp. 3896–3903. [CrossRef]
46. Bronte, R.; Shahriar, H.; Haddad, H. Information Theoretic Anomaly Detection Framework for Web Application. In Proceedings of the 2016 IEEE 40th Annual Computer Software and Applications Conference (COMPSAC), Atlanta, GA, USA, 10–14 June 2016; Volume 2, pp. 394–399. [CrossRef]
47. Luo, Y.; Cheng, S.; Liu, C.; Jiang, F. PU Learning in Payload-based Web Anomaly Detection. In Proceedings of the 2018 Third International Conference on Security of Smart Cities, Industrial Control System and Communications (SSIC), Shanghai, China, 18–19 October 2018; pp. 1–5. [CrossRef]
48. Ren, X.; Hu, Y.; Kuang, W.; Souleymanou, M.B. A Web Attack Detection Technology Based on Bag of Words and Hidden Markov Model. In Proceedings of the 2018 IEEE 15th International Conference on Mobile Ad Hoc and Sensor Systems (MASS), Chengdu, China, 9–12 October 2018; pp. 526–531. [CrossRef]
49. Kozik, R.; Choraś, M.; Hołubowicz, W. Hardening Web Applications against SQL Injection Attacks Using Anomaly Detection Approach. In *Image Processing & Communications Challenges 6*; Choraś, R.S., Ed.; Springer International Publishing: Cham, Switzerland, 2015; pp. 285–292.
50. Maggi, F.; Robertson, W.; Kruegel, C.; Vigna, G. Protecting a Moving Target: Addressing Web Application Concept Drift. In *Recent Advances in Intrusion Detection*; Kirda, E., Jha, S., Balzarotti, D., Eds.; Springer:Berlin/Heidelberg, Germany, 2009; pp. 21–40.
51. Valeur, F.; Vigna, G.; Kruegel, C.; Kirda, E. An Anomaly-Driven Reverse Proxy for Web Applications. In Proceedings of the 2006 ACM Symposium on Applied Computing, Dijon, France, 23–27 April 2006; Association for Computing Machinery: New York, NY, USA, 2006; pp. 361–368. [CrossRef]
52. Guangmin, L. Modeling Unknown Web Attacks in Network Anomaly Detection. In Proceedings of the 2008 Third International Conference on Convergence and Hybrid Information Technology, Busan, Korea, 11–13 November 2008; Volume 2, pp. 112–116. [CrossRef]
53. Yu, S.; Guo, S.; Stojmenovic, I. Fool Me If You Can: Mimicking Attacks and Anti-Attacks in Cyberspace. *IEEE Trans. Comput.* **2015**, *64*, 139–151. [CrossRef]

54. Sakib, M.N.; Huang, C. Using anomaly detection based techniques to detect HTTP-based botnet C C traffic. In Proceedings of the 2016 IEEE International Conference on Communications (ICC), Kuala Lumpur, Malaysia, 22–27 May 2016; pp. 1–6. [CrossRef]
55. Medvet, E.; Bartoli, A. On the Effects of Learning Set Corruption in Anomaly-Based Detection of Web Defacements. In *Detection of Intrusions and Malware, and Vulnerability Assessment*; Hämmerli, M.B., Sommer, R., Eds.; Springer: Berlin/Heidelberg, Germany, 2007; pp. 60–78.
56. Davanzo, G.; Medvet, E.; Bartoli, A. Anomaly detection techniques for a web defacement monitoring service. *Expert Syst. Appl.* **2011**, *38*, 12521–12530. [CrossRef]
57. Juvonen, A.; Sipola, T.; Hämäläinen, T. Online anomaly detection using dimensionality reduction techniques for HTTP log analysis. *Comput. Netw.* **2015**, *91*, 46–56. [CrossRef]
58. Wang, W.; Guyet, T.; Quiniou, R.; Cordier, M.O.; Masseglia, F.; Zhang, X. Autonomic Intrusion Detection. *Know.-Based Syst.* **2014**, *70*, 103–117. [CrossRef]
59. Vartouni, A.M.; Kashi, S.S.; Teshnehlab, M. An anomaly detection method to detect web attacks using Stacked Auto-Encoder. In Proceedings of the 2018 6th Iranian Joint Congress on Fuzzy and Intelligent Systems (CFIS), Kerman, Iran, 28 February–2 March 2018; pp. 131–134. [CrossRef]
60. Zolotukhin, M.; Hämäläinen, T.; Kokkonen, T.; Siltanen, J. Analysis of HTTP requests for anomaly detection of web attacks. In Proceedings of the 2014 World Ubiquitous Science Congress: 2014 IEEE 12th International Conference on Dependable, Autonomic and Secure Computing, DASC 2014, Dalian, China, 24–27 August 2014; IEEE: Piscataway, NJ, USA, 2014; pp. 406–411. [CrossRef]
61. Asselin, E.; Aguilar-Melchor, C.; Jakllari, G. Anomaly detection for web server log reduction: A simple yet efficient crawling based approach. In Proceedings of the 2016 IEEE Conference on Communications and Network Security (CNS), Philadelphia, PA, USA, 17–19 October 2016; pp. 586–590. [CrossRef]
62. Zhang, S.; Li, B.; Li, J.; Zhang, M.; Chen, Y. A Novel Anomaly Detection Approach for Mitigating Web-Based Attacks Against Clouds. In Proceedings of the 2015 IEEE 2nd International Conference on Cyber Security and Cloud Computing (CSCloud), New York, NY, USA, 3–5 November 2015; IEEE Computer Society: Piscataway, NJ, USA, 2015; pp. 289–294. [CrossRef]
63. Zhang, M.; Lu, S.; Xu, B. An Anomaly Detection Method Based on Multi-models to Detect Web Attacks. In Proceedings of the 2017 10th International Symposium on Computational Intelligence and Design (ISCID), Hangzhou, China, 9–10 December 2017; Volume 2, pp. 404–409. [CrossRef]
64. Parhizkar, E.; Abadi, M. OC-WAD: A one-class classifier ensemble approach for anomaly detection in web traffic. In Proceedings of the 2015 23rd Iranian Conference on Electrical Engineering, Tehran, Iran, 10–14 May 2015; pp. 631–636. [CrossRef]
65. Kozik, R.; Choras, M. Adapting an Ensemble of One-Class Classifiers for a Web-Layer Anomaly Detection System. In Proceedings of the 2015 10th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing, 3PGCIC, Krakow, Poland, 4–6 November 2015; IEEE: Piscataway, NJ, USA, 2015; pp. 724–729. [CrossRef]
66. Cao, Q.; Qiao, Y.; Lyu, Z. Machine learning to detect anomalies in web log analysis. In Proceedings of the 2017 3rd IEEE International Conference on Computer and Communications (ICCC), Chengdu, China, 13–16 December 2017; pp. 519–523. [CrossRef]
67. Yu, J.; Tao, D.; Lin, Z. A hybrid web log based intrusion detection model. In Proceedings of the 2016 4th IEEE International Conference on Cloud Computing and Intelligence Systems, CCIS 2016, Beijing, China, 17–19 August 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 356–360. [CrossRef]
68. Threepak, T.; Watcharapupong, A. Web attack detection using entropy-based analysis. In Proceedings of theInternational Conference on Information Networking, Phuket, Thailand, 10–12 February 2014; IEEE: Piscataway, NJ, USA, 2014; pp. 244–247. [CrossRef]
69. Swarnkar, M.; Hubballi, N. Rangegram: A novel payload based anomaly detection technique against web traffic. In Proceedings of the 2015 IEEE International Conference on Advanced Networks and Telecommuncations Systems (ANTS), Kolkata, India, 15–18 Decemnber 2015; pp. 1–6. [CrossRef]
70. Xu, H.; Tao, L.; Lin, W.; Wu, Y.; Liu, J.; Wang, C. A model for website anomaly detection based on log analysis. In Proceedings of the 2014 IEEE 3rd International Conference on Cloud Computing and Intelligence Systems, Shenzhen, China, 27–29 November 2014; pp. 604–608. [CrossRef]
71. Park, S.; Kim, M.; Lee, S. Anomaly Detection for HTTP Using Convolutional Autoencoders. *IEEE Access* **2018**, *6*, 70884–70901. [CrossRef]

72. Choraś, M.; Kozik, R. Machine learning techniques applied to detect cyber attacks on web applications. *Log. J. IGPL* **2014**, *23*, 45–56. [CrossRef]
73. Tharshini, M.; Ragavinodini, M.; Senthilkumar, R. Access Log Anomaly Detection. In Proceedings of the 2017 Ninth International Conference on Advanced Computing (ICoAC), Chennai, India, 14–16 December 2017; pp. 375–381.
74. Kozik, R.; Choraś, M.; Hołubowicz, W. Packets tokenization methods for web layer cyber security. *Log. J. IGPL* **2016**, *25*, 103–113. [CrossRef]
75. Kamarudin, M.H.; Maple, C.; Watson, T.; Safa, N.S. A LogitBoost-Based Algorithm for Detecting Known and Unknown Web Attacks. *IEEE Access* **2017**, *5*, 26190–26200. [CrossRef]
76. Yu, Y.; Liu, G.; Yan, H.; Li, H.; Guan, H. Attention-Based Bi-LSTM Model for Anomalous HTTP Traffic Detection. In Proceedings of the 2018 15th International Conference on Service Systems and Service Management (ICSSSM), Hangzhou, China, 21–22 July 2018; pp. 1–6.
77. Nguyen, X.N.; Nguyen, D.T.; Vu, L.H. POCAD: A novel pay load-based one-class classifier for anomaly detection. In Proceedings of the 2016 3rd National Foundation for Science and Technology Development Conference on Information and Computer Science (NICS), Danang, Vietnam, 14–16 September 2016; pp. 74–79. [CrossRef]
78. Lu, L.; Zhu, X.; Zhang, X.; Liu, J.; Bhuiyan, M.Z.A.; Cui, G. One Intrusion Detection Method Based On Uniformed Conditional Dynamic Mutual Information. In Proceedings of the 2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE), New York, NY, USA , 1–3 August 2018; pp. 1236–1241. [CrossRef]
79. Moustafa, N.; Misra, G.; Slay, J. Generalized Outlier Gaussian Mixture technique based on Automated Association Features for Simulating and Detecting Web Application Attacks. *IEEE Trans. Sustain. Comput.* **2018**, 1. [CrossRef]
80. Alrawashdeh, K.; Purdy, C. Fast Activation Function Approach for Deep Learning Based Online Anomaly Intrusion Detection. In Proceedings of the 2018 IEEE 4th International Conference on Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing, (HPSC) and IEEE International Conference on Intelligent Data and Security (IDS), Omaha, NE, USA, 3–5 May 2018; pp. 5–13. [CrossRef]
81. Kaur, R.; Bansal, M. Multidimensional attacks classification based on genetic algorithm and SVM. In Proceedings of the 2016 2nd International Conference on Next Generation Computing Technologies (NGCT), Dehradun, India, 14–16 October 2016; pp. 561–565. [CrossRef]
82. Angiulli, F.; Argento, L.; Furfaro, A. Exploiting N-Gram Location for Intrusion Detection. In Proceedings of the 2015 IEEE 27th International Conference on Tools with Artificial Intelligence (ICTAI), Vietri sul Mare, Italy, 9–11 November 2015; pp. 1093–1098. [CrossRef]
83. Hiremagalore, S.; Barbará, D.; Fleck, D.; Powell, W.; Stavrou, A. transAD: An Anomaly Detection Network Intrusion Sensor for the Web. In *Information Security*; Chow, S.S.M., Camenisch, J., Hui, L.C.K., Yiu, S.M., Eds.; Springer International Publishing: Cham, Switzerland, 2014; pp. 477–489. [CrossRef]
84. Favaretto, M.; Spolaor, R.; Conti, M.; Ferrante, M. You Surf so Strange Today: Anomaly Detection in Web Services via HMM and CTMC. In *Green, Pervasive, and Cloud Computing*; Au, M.H.A., Castiglione, A., Choo, K.K.R.;, Palmieri, F., Li, K.C., Eds.; Springer International Publishing: Cham, Switzerland, 2017; pp. 426–440. [CrossRef]
85. Kozik, R.; Chorás, M. The http content segmentation method combined with adaboost classifier for web-layer anomaly detection system. *Adv. Intell. Syst. Comput.* **2017**, *527*, 555–563. [CrossRef]
86. Kozik, R.; Choraś, M.; Holubowicz, W.; Renk, R. Extreme Learning Machines for Web Layer Anomaly Detection. In *Image Processing and Communications Challenges 8*; Choraś, R.S., Ed.; Springer International Publishing: Cham, Switzerland, 2017; pp. 226–233. [CrossRef]
87. Kozik, R.; Choraś, M.; Renk, R.; Holubowicz, W. Patterns Extraction Method for Anomaly Detection in HTTP Traffic. In *Proceedings of the International Joint Conference*; Herrero, Á., Baruque, B., Sedano, J., Quintián, H., Corchado, E., Eds.; Springer International Publishing: Cham, Switzerland, 2015; pp. 227–236. [CrossRef]
88. Shi, Y.; Wang, S.; Zhao, Q.; Li, J. A Hybrid Approach of HTTP Anomaly Detection. In *Web and Big Data*; Springer International Publishing: Cham, Switzerland, 2017; pp. 128–137. [CrossRef]

89. Kim, T.Y.; Cho, S.B. Web traffic anomaly detection using C-LSTM neural networks. *Expert Syst. Appl.* **2018**, *106*, 66–76. [CrossRef]
90. Jin, X.; Cui, B.; Li, D.; Cheng, Z.; Yin, C. An improved payload-based anomaly detector for web applications. *J. Netw. Comput. Appl.* **2018**, *106*, 111–116. [CrossRef]
91. Wang, W.; Liu, J.; Pitsilis, G.; Zhang, X. Abstracting massive data for lightweight intrusion detection in computer networks. *Inf. Sci.* **2018**, *433–434*, 417–430. [CrossRef]
92. Liu, T.; Zhang, L. Application of Logistic Regression in WEB Vulnerability Scanning. In Proceedings of the 2018 International Conference on Sensor Networks and Signal Processing (SNSP), Xi'an, China, 28–31 October 2018; pp. 486–490. [CrossRef]
93. Betarte, G.; Gimenez, E.; Martinez, R.; Pardo, A. Improving Web Application Firewalls through Anomaly Detection. In Proceedings of the 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA), Orlando, FL, USA, 17–20 December 2018; pp. 779–784. [CrossRef]
94. Li, B.; Yuan, G.; Shen, L.; Zhang, R.; Yao, Y. Incorporating URL embedding into ensemble clustering to detect web anomalies. *Future Gener. Comput. Syst.* **2019**, *96*, 176–184. [CrossRef]
95. Yun, Y.; Park, S.; Kim, Y.; Ryou, J. A Design and Implementation of Profile Based Web Application Securing Proxy. In *Information Security Practice and Experience*; Chen, K., Deng, R., Lai, X., Zhou, J., Eds.; Springer: Berlin/Heidelberg, Germany, 2006; pp. 248–259. [CrossRef]
96. Bolzoni, D.; Etalle, S. Boosting Web Intrusion Detection Systems by Inferring Positive Signatures. In *On the Move to Meaningful Internet Systems: OTM 2008*; Meersman, R., Tari, Z., Eds.; Springer: Berlin/Heidelberg, Germany, 2008; pp. 938–955. [CrossRef]
97. Li, Y.; Guo, L.; Tian, Z.H.; Lu, T.B. A Lightweight Web Server Anomaly Detection Method Based on Transductive Scheme and Genetic Algorithms. *Comput. Commun.* **2008**, *31*, 4018–4025. [CrossRef]
98. Kruegel, C.; Vigna, G.; Robertson, W. A multi-model approach to the detection of web-based attacks. *Comput. Netw.* **2005**, *48*, 717–738. [CrossRef]
99. Cho, S.; Cha, S. SAD: Web session anomaly detection based on parameter estimation. *Comput. Secur.* **2004**, *23*, 312–319. [CrossRef]
100. Yamada, A.; Miyake, Y.; Takemori, K.; Studer, A.; Perrig, A. Intrusion Detection for Encrypted Web Accesses. In Proceedings of the 21st International Conference on Advanced Information Networking and Applications Workshops (AINAW'07), Niagara Falls, ON, Canada, 21–23 May 2007; Volume 1, pp. 569–576. [CrossRef]
101. Yan, C.; Qin, Z.; Shi, Y. Sequence Analysis and Anomaly Detection of Web Service Composition. In Proceedings of the 2008 International Conference on Computer Science and Software Engineering, Wuhan, China, 12–14 December 2008; Volume 3; pp. 1043–1048. [CrossRef]
102. Jamdagni, A.; Tan, Z.; Nanda, P.; He, X.; Liu, R.P. Intrusion Detection Using GSAD Model for HTTP Traffic on Web Services. In Proceedings of the 6th International Wireless Communications and Mobile Computing Conference, Caen, France, 15 January 2010; Association for Computing Machinery: New York, NY, USA; pp. 1193–1197. [CrossRef]
103. Wang, W.; Zhang, X. High-Speed Web Attack Detection through Extracting Exemplars from HTTP Traffic. In Proceedings of the 2011 ACM Symposium on Applied Computing, TaiChung, Taiwan, 21–24 March 2011; Association for Computing Machinery: New York, NY, USA; pp. 1538–1543. [CrossRef]
104. Kruegel, C.; Vigna, G. Anomaly detection of Web-based attacks. In Proceedings of the ACM Conference on Computer and Communications Security, Washington, DC, USA, 27–30 October 2003; ACM Press: New York, NY, USA; pp. 251–261. [CrossRef]
105. Rahnavard, G.; Najjar, M.S.A.; Taherifar, S. A method to evaluate Web Services Anomaly Detection using Hidden Markov Models. In Proceedings of the 2010 International Conference on Computer Applications and Industrial Electronics, Kuala Lumpur, Malaysia, 5–8 December 2010; pp. 261–265. [CrossRef]
106. Das, D.; Sharma, U.; Bhattacharyya, D.K. A Web Intrusion Detection Mechanism based on Feature based Data Clustering. In Proceedings of the 2009 IEEE International Advance Computing Conference, Patiala, India, 6–7 March 2009; pp. 1124–1129. [CrossRef]
107. Li, X.; Xue, Y.; Malin, B. Detecting Anomalous User Behaviors in Workflow-Driven Web Applications. In Proceedings of the 2012 IEEE 31st Symposium on Reliable Distributed Systems, Irvine, CA, USA, 8–11 October 2012; pp. 1–10. [CrossRef]
108. Le, M.; Stavrou, A.; Kang, B.B. DoubleGuard: Detecting Intrusions in Multitier Web Applications. *IEEE Trans. Dependable Secur. Comput.* **2012**, *9*, 512–525. [CrossRef]

109. Xie, Y.; Yu, S.Z. Light-weight detection of HTTP attacks for large-scale Web sites. In Proceedings of the 2008 11th IEEE Singapore International Conference on Communication Systems, Guangzhou, China, 19–21 November 2008; pp. 1182–1186. [CrossRef]
110. Sriraghavan, R.G.; Lucchese, L. Data processing and anomaly detection in web-based applications. In Proceedings of the 2008 IEEE Workshop on Machine Learning for Signal Processing, Cancun, Mexico, 16–19 October 2008; pp. 187–192. [CrossRef]
111. Fan, W.K.G. An adaptive anomaly detection of WEB-based attacks. In Proceedings of the 2012 7th International Conference on Computer Science Education (ICCSE), Melbourne, VIC, Australia, 14–17 July 2012; pp. 690–694. [CrossRef]
112. Kirchner, M. A framework for detecting anomalies in HTTP traffic using instance-based learning and k-nearest neighbor classification. In Proceedings of the 2010 2nd International Workshop on Security and Communication Networks (IWSCN), Karlstad, Sweden, 26–28 May 2010; pp. 1–8. [CrossRef]
113. Kakavand, M.; Mustapha, A.; Tan, Z.; Yazdani, S.F.; Arulsamy, L. O-ADPI: Online Adaptive Deep-Packet Inspector Using Mahalanobis Distance Map for Web Service Attacks Classification. *IEEE Access* **2019**, *7*, 167141–167156. [CrossRef]
114. Moradi Vartouni, A.; Teshnehlab, M.; Sedighian Kashi, S. Leveraging deep neural networks for anomaly-based web application firewall. *IET Inf. Secur.* **2019**, *13*, 352–361. [CrossRef]
115. Li, J.; Fu, Y.; Xu, J.; Ren, C.; Xiang, X.; Guo, J. Web application attack detection based on attention and gated convolution networks. *IEEE Access* **2019**, 1. [CrossRef]
116. Alhakami, W.; ALharbi, A.; Bourouis, S.; Alroobaea, R.; Bouguila, N. Network Anomaly Intrusion Detection Using a Nonparametric Bayesian Approach and Feature Selection. *IEEE Access* **2019**, *7*, 52181–52190. [CrossRef]
117. Kozik, R.; Choraś, M. Protecting the application layer in the public domain with machine learning methods. *Log. J. IGPL* **2018**, *27*, 149–159. [CrossRef]
118. Jin, L.; Wang, X.J.; Zhang, Y.; Yao, L. Anomaly Detection in the Web Logs Using Unsupervised Algorithm. In *Human Centered Computing*; Tang, Y., Zu, Q., Rodríguez García, J.G., Eds.; Springer International Publishing: Cham, Switzerland, 2019; pp. 393–405. [CrossRef]
119. Bhattacharyya, D.K.; Kalita, J.K. *DDoS Attacks: Evolution, Detection, Prevention, Reaction, and Tolerance*; CRC Press: Boca Raton, FL, USA, 2016.
120. OWASP Foundation. Injection Flaws | OWASP, Available online: https://owasp.org/www-community/Injection_Flaws (accesed on 15 May 2020).
121. Wei, K.; Muthuprasanna, M.; Kothari, S. Preventing SQL injection attacks in stored procedures. In Proceedings of the Australian Software Engineering Conference (ASWEC'06), Sydney, NSW, Australia, 18–21 April 2006; pp. 8–198. [CrossRef]
122. Leonard, J.; Xu, S.; Sandhu, R. A Framework for Understanding Botnets. In Proceedings of the 2009 International Conference on Availability, Reliability and Security, Fukuoka, Japan, 16–19 March 2009; pp. 917–922.
123. Hadianto, R.; Purboyo, T.W. A Survey Paper on Botnet Attacks and Defenses in Software Defined Networking. *Int. J. Appl. Eng. Res.* ***2018***, 13, *483–489.*
124. Gurjwar, R.K.; Sahu, D.R.; Tomar, D.S. An approach to reveal website defacement. *Int. J. Comput. Sci. Inf. Secur.* **2013**, *11*, 73.
125. Cluster analysis—Wikipedia. Available online: https://en.wikipedia.org/wiki/Cluster_analysis#Definition. (accesed on 9 February 2020).
126. Unioviedo. kmeans. Available online: https://www.unioviedo.es/compnum/labs/new/kmeans.html. (accessed on 17 May 2020).
127. Frey, B.J.; Dueck, D. Clustering by Passing Messages Between Data Points. *Science* **2007**, *315*, 972–976. [CrossRef] [PubMed]
128. Ester, M.; Kriegel, H.P.; Sander, J.; Xu, X. A Density-Based Algorithm for Discovering Clusters a Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, Portland, OR, USA, 2–4 August 1996; KDD'96; pp. 226–231.

129. Breunig, M.M.; Kriegel, H.P.; Ng, R.T.; Sander, J. LOF: Identifying Density-Based Local Outliers. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*; Association for Computing Machinery: New York, NY, USA, 2000; pp. 93–104. [CrossRef]
130. Dempster, A.P.; Laird, N.M.; Rubin, D.B. Maximum Likelihood from Incomplete Data via the EM Algorithm. *J. R. Stat. Soc. Ser. B (Methodol.)* **1977**, *39*, 1–38.
131. Gupta, M.R.; Chen, Y. Theory and Use of the EM Algorithm. *Found. Trends Signal Process.* **2011**, *4*, 223–296. [CrossRef]
132. Rahul, A.E.; Narukulla, S. Introduction to Data Mining and Machine Learning Algorithms. *Int. J. Res. Eng. Sci. Manag.* **2018**, *1*.
133. Duda, R.O.; Hart, P.E. *Pattern Classification and Scene Analysis*; Duda, R.O., Hart, P.E., Eds.; Wiley: New York, NY, USA, 1973.
134. Schölkopf, B.; Williamson, R.; Smola, A.; Shawe-Taylor, J.; Platt, J. Support Vector Method for Novelty Detection. In *Proceedings of the 12th International Conference on Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 1999; pp. 582–588.
135. Franzese, M.; Iuliano, A. Hidden Markov Models. In *Encyclopedia of Bioinformatics and Computational Biology*; Ranganathan, S., Gribskov, M., Nakai, K., Schönbach, C.B.T.E.O.B.; Eds.; Academic Press: Oxford, UK, 2019; pp. 753–762. [CrossRef]
136. Rabiner, L.; Juang, B. An introduction to hidden Markov models. *IEEE ASSP Mag.* **1986**, *3*, 4–16. [CrossRef]
137. Altman, N.S. An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression. *Am. Stat.* **1992**, *46*, 175–185. [CrossRef]
138. Maron, M.E. Automatic Indexing: An Experimental Inquiry. *J. ACM* **1961**, *8*, 404–417. [CrossRef]
139. Domingos, P.; Pazzani, M. On the Optimality of the Simple Bayesian Classifier under Zero-One Loss. *Mach. Learn.* **1997**, *29*, 103–130. [CrossRef]
140. Webb, G.I.; Boughton, J.R.; Wang, Z. Not So Naive Bayes: Aggregating One-Dependence Estimators. *Mach. Learn.* **2005**, *58*, 5–24. [CrossRef]
141. Hopfield, J.J. Neural networks and physical systems with emergent collective computational abilities. *Proc. Natl. Acad. Sci. USA* **1982**, *79*, 2554–2558. [CrossRef] [PubMed]
142. Liu, G.; Bao, H.; Han, B. A Stacked Autoencoder-Based Deep Neural Network for Achieving Gearbox Fault Diagnosis. *Math. Probl. Eng.* **2018**, *2018*, 5105709. [CrossRef]
143. Puig-Arnavat, M.; Bruno, J.C. Artificial Neural Networks for Thermochemical Conversion of Biomass. *Recent Adv. Thermo-Chem. Convers. Biomass* **2015**, 133–156. [CrossRef]
144. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient estimation of word representations in vector space. *arXiv* **2013**, arXiv:1301.3781.
145. Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.S.; Dean, J. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*; The MIT Press: Cambridge, MA, USA, 2013; pp. 3111–3119.
146. Tomović, A.; Janičić, P.; Kešelj, V. n-Gram-based classification and unsupervised hierarchical clustering of genome sequences. *Comput. Methods Programs Biomed.* **2006**, *81*, 137–153. [CrossRef]
147. Tauritz, D. *Applications of n-grams*; Technical Report; Department of Computer Science, University of Missouri-Rolla: Rolla, MI, USA, 2002.
148. Manning, C.D.; Raghavan, P.; Schütze, H. Scoring, term weighting, and the vector space model. In *Introduction to Information Retrieval*; Manning, C.D., Schütze, H., Raghavan, P., Eds.; Cambridge University Press: Cambridge, UK, 2008; pp. 100–123. [CrossRef]
149. Stephen, R. Understanding inverse document frequency: On theoretical arguments for IDF. *J. Doc.* **2004**, *60*, 503–520. [CrossRef]
150. Roweis, S.T.; Saul, L.K. Nonlinear Dimensionality Reduction by Locally Linear Embedding. *Science* **2000**, *290*, 2323 LP – 2326. [CrossRef]
151. Pearson, K. LIII. On lines and planes of closest fit to systems of points in space. *Philos. Mag. J. Sci.* **1901**, *2*, 559–572. [CrossRef]
152. Hotelling, H. Relations Between Two Sets of Variates. *Biometrika* **1936**, *28*, 321–377. [CrossRef]
153. Jolliffe, I.T. *Principal Component Analysis*; Springer Series in Statistics; Springer: New York, NY, USA, 2002. [CrossRef]

154. Fisher, R.A. The Use of Multiple Measurements in Taxonomic Problems. *Ann. Eugen.* **1936**, *7*, 179–188. [CrossRef]
155. McLachlan, G.J. *Discriminant Analysis and Statistical Pattern Recognition*; John Wiley & Sons: Hoboken, NJ, USA, 2004; Volume 544.
156. Rao, C.R. The Utilization of Multiple Measurements in Problems of Biological Classification. *J. R. Stat. Soc. Ser. B (Methodol.)* **1948**, *10*, 159–203. [CrossRef]
157. Coifman, R.R.; Lafon, S.; Lee, A.B.; Maggioni, M.; Nadler, B.; Warner, F.; Zucker, S.W. Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps. *Proc. Natl. Acad. Sci. USA* **2005**, *102*, 7426–7431. [CrossRef] [PubMed]
158. Coifman, R.R.; Lafon, S. Diffusion maps. *Appl. Comput. Harmon. Anal.* **2006**, *21*, 5–30. [CrossRef]
159. Delaporte, J.; Herbst, B.M.; Hereman, W.; der Walt Stéfan, V. An introduction to diffusion maps. In Proceedings of the 19th Symposium of the Pattern Recognition Association of South Africa (PRASA 2008), Cape Town, South Africa , 27–28 November 2008.
160. Steliga, K.; Szynal, D. On Markov-type inequalities. *Int. J. Pure Appl. Math.* **2010**, *58*, 137–152.
161. Pearson, K.X. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *Lond. Edinb. Dublin Philos. Mag. J. Sci.* **1900**, *50*, 157–175. [CrossRef]
162. Olshausen, B.A. *Bayesian Probability Theory*; The Redwood Center for Theoretical Neuroscience, Helen Wills Neuroscience Institute at the University of California at Berkeley: Berkeley, CA, USA, 2004.
163. Kozik, R.; Choraś, M.; Renk, R.; Holubowicz, W. Semi-unsupervised Machine Learning for Anomaly Detection in HTTP Traffic. In Proceedings of the 9th International Conference on Computer Recognition Systems CORES 2015, Wroclaw, Poland, 25–27 May 2015; Springer International Publishing: Cham, Switzerland, 2016; pp. 767–775. [CrossRef]
164. Lichman, M. *1999 DARPA Intrusion Detection Evaluation Dataset*; MIT Lincoln Laboratory: Cambridge, MA, USA, 2000.
165. Hettich, S.; Bay, S.D. The UCI KDD Archive. Available online: http://kdd.ics.uci.edu (accessed on 15 March 2020).
166. Sommer, R.; Paxson, V. Outside the Closed World: On Using Machine Learning for Network Intrusion Detection. In Proceedings of the 2010 IEEE Symposium on Security and Privacy, Berkeley/Oakland, CA, USA, 16–19 May 2010; pp. 305–316. [CrossRef]
167. Sommer, R. *Viable Network Intrusion Detection: Trade-Offs in High-Performance Environments*; VDM Verlag: Saarbrücken, DEU, 2008.
168. Siddique, K.; Akhtar, Z.; Khan, F.A.; Kim, Y. KDD Cup 99 Data Sets: A Perspective on the Role of Data Sets in Network Intrusion Detection Research. *Computer* **2019**, *52*, 41–51. [CrossRef]
169. Moustafa, N. The UNSW-NB15 data set description.
170. Moustafa, N.M.; Slay, J. The significant features of the UNSW-NB15 and the KDD99 Data sets for Network Intrusion Detection Systems. In Proceedings of the 2015 4th International Workshop on Building Analysis Datasets and Gathering Experience Returns for Security (BADGERS), Kyoto, Japan, 5 November 2015. [CrossRef]
171. Moustafa, N.; Slay, J. UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In Proceedings of the 2015 Military Communications and Information Systems Conference (MilCIS), Canberra, ACT, Australia, 10–12 November 2015; pp. 1–6. [CrossRef]
172. Singh, R.; Kumar, H.; Singla, R. An intrusion detection system using network traffic profiling and online sequential extreme learning machine. *Expert Syst. Appl.* **2015**, *42*, 8609–8624. [CrossRef]
173. Shiravi, A.; Shiravi, H.; Tavallaee, M.; Ghorbani, A.A. Toward developing a systematic approach to generate benchmark datasets for intrusion detection. *Comput. Secur.* **2012**, *31*, 357–374. [CrossRef]
174. Torrano-Gimenez, C.; Perez-Villegas, A.; Alvarez, G. A Self-learning Anomaly-Based Web Application Firewall. In *Computational Intelligence in Security for Information Systems*; Advances in Intelligent and Soft Computing; Herrero, Á., Gastaldo, P., Zunino, R., Corchado, E., Eds.; Springer: Berlin/Heidelberg, Germany, 2009; pp. 85–92.
175. Raïssi, C.; Brissaud, J.; Dray, G.; Poncelet, P.; Roche, M.; Teisseire, M. Web Analyzing Traffic Challenge: Description and Results. In Proceedings of the ECML/PKDD'2007 Discovery Challenge, Warsaw, Poland, 17 September 2007; p. 6.
176. Van Rijsbergen, C. *Information Retrieval*, 2nd ed.; Butterworth-Heinemann: Newton, MA, USA, 1979.

177. Díaz, G.; Bermejo, J.R. Static analysis of source code security: Assessment of tools against SAMATE tests. *Inf. Softw. Technol.* **2013**, *55*, 1462–1476. [CrossRef]
178. Bermejo Higuera, J.R. Metodología de Evaluación de Herramientas de Análisis Automático de Seguridad de Aplicaciones Web Para su Adaptación en el ciclo de vida de Desarrollo. Ph.D. Thesis, Universidad Nacional Educación a Distancia (UNED), Madrid, Spain, 2013.
179. Matthews, B.W. Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochim. Et Biophys. Acta (BBA) - Protein Struct.* **1975**, *405*, 442–451. [CrossRef]
180. Swets, J.A. *Signal Detection Theory and ROC Analysis in Psychology and Diagnostics: Collected Papers*; Scientific Psychology Series; Lawrence Erlbaum Associates, Inc.: Hillsdale, NJ, USA, 1996.
181. OWASP Foundation. OWASP Top Ten, 2017.
182. MITRE Corporation. CAPEC—Common Attack Pattern Enumeration and Classification (CAPEC), 2011.
183. MITRE Corporation. CWE—Common Weakness Enumeration.
184. OWASP Foundation. OWASP Automated Threats to Web Applications. Available online: https://owasp.org/www-project-automated-threats-to-web-applications/ (accessed on 3 March 2020).
185. Antunes, N.; Vieira, M. On the Metrics for Benchmarking Vulnerability Detection Tools. In Proceedings of the 2015 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, Rio de Janeiro, Brazil, 22–25 June 2015; pp. 505–516. [CrossRef]
186. Kruegel, C.; Toth, T.; Kirda, E. Service Specific Anomaly Detection for Network Intrusion Detection. In *Proceedings of the 2002 ACM Symposium on Applied Computing*; Association for Computing Machinery: New York, NY, USA, 2002; pp. 201–208. [CrossRef]