A Comprehensive Review of Anomaly Detection in Web Logs

Mehryar Majd, Pejman Najafi, Seyed Ali Alhosseini, Feng Cheng, Christoph Meinel
Hasso Plattner Institute (HPI), University of Potsdam
Potsdam, Germany
{firstname.lastname}@hpi.de

***Abstract*—Anomaly detection is a significant problem that has been researched within diverse research areas and application domains, especially in the area of web-based internet services or cybersecurity. Many anomaly detection techniques have been developed for specific application domains, while others are more generic. The Log files of Web-server give insight into the state of web-server and applications running on it and enable the detection of abnormal incidents or behavior. This paper focuses on particularly web-server HTTP logs to the problems of Web-server Log Anomaly Detection (WLAD) due to their own nature and features and aims to provide a brief review of different Data-driven techniques to get to the bottom of recent studies and developments made in the context of WLAD. Moreover, in this paper, the literature related to webserver logs analysis, as well as other closely related to the WLAD topic, are taken into consideration for review. We have classified existing techniques into different categories based on the underlying approach adopted. When applying a particular technique, these assumptions can be used as guidelines to assess the method's effectiveness in this area. We also provide a basic security anomaly detection approach for each category and compare the existing methods as variants of the basic technique. Further, we identify the cons and pros of the current practices for each category. We also discuss the computational complexity of the methods, which is an essential issue in the domain of Big Data.**

***Index Terms*—Anomaly detection, Web-server log analysis, Malicious HTTP, Web-server attack, Cybersecurity**

I. Introduction

Web applications have been the most used internet-based services in current business practice, and various application services including computer resources, e.g., Web server, web-app server, and storage using web-based applications over the internet. Therefore, the usage of web applications is inevitable nowadays. Therefore service providers are always interested in monitoring malicious activities to manage their business processes. Recently, the rising number of security events and substantial economic losses, mainly due to Covid-19 outbreaks, warns us that the detection of malicious traffic has become a great challenge for both individuals and enterprises to protect their digital assets and services/applications. Intrusion detection systems (IDS) and Fraud Detection systems (FDS) play a critical role in any organization's network security strategy as well as the lifeblood of network monitoring.

Web-apps are hosted on web servers, computers running an operating system, connected to the back-end database, and running various applications. Figure 1 shows the general schema of the web-apps stack. Web servers are the front doors in the backend that provide services to the end user [1].

According to Statistics, 42% of web applications are exposed to threats and hackers [2]. A recent survey on application security shows that approximately 40% of companies have been victims of a security breach due to an external attack carried out by a web application exploit. This issue has become increasingly complex with the adoption of cloud and containers, and the growing use of APIs [3]. The Open Web Application Security Project (OWASP) lists the top 10 web application vulnerabilities [4], and the Common Attack Pattern Enumeration and Classification (CAPEC) curates many known attack patterns [5]. Nowadays, Intrusion detection systems (IDS) and Fraud Detection systems (FDS) play a critical role in any organization's network security strategy as well as the lifeblood of network monitoring.

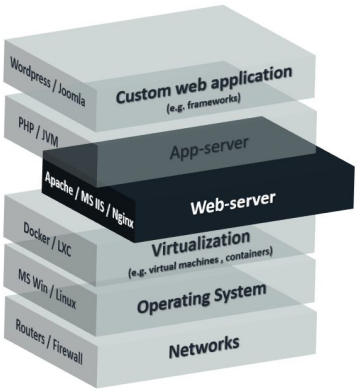

Fig. 1. Typical schema of the web-apps stack for web-app while one can see other stacks underline Web-server and App-server, which could be attacked by injections or other web applications vulnerabilities have been reported by OWASP in 2021 [4].

Traditionally, IDSs have two central schemes for web-server attacks: signature-based intrusion detection, known as misuse detection [6], [7], and Behavior-based solutions, also known as anomaly-based threat detection or anomaly detection [8]. Misuse detection can only detect known attacks and filter HTTP requests based on a predefined rule or pattern. Most Web Application Firewalls (WAF) use misuse detection methods, but the malicious request keywords are replaced or encoded multiple times, which can bypass WAF [9], [10]. Anomaly detection use deviation of normal data pattern among

activities that do not conform to normal/expected behavior. It can detect unknown and new attacks by creating a model of "normal" use.

Web logs collected from the web-servers (e.g., Apache HTTP Server), proxy servers or reverse proxy servers (e.g., Nginx Web Server) are a valuable source of information due to their characteristics discussed in subsection II-A, allowing one to find traces of possible attacks. Thus, Auditing and reviewing the logs from web servers frequently by developing the smart pipeline and improving threat intelligence can effectively establish a comprehensive security posture. In this regard, log analysis can help not only optimize or debug system performance but also detect malicious activities that have managed to bypass the majority of the IDS solutions. Anomalies in web-server logs are the patterns in data that deviate significantly from the expected logs. This deviation could indicate deviation from normal behavior that could be considered malicious [11].

Table I. Examples of web-server logs highlighting some attacks and exploitation of injections (e.g., XSS, SQLi, Log4j), where the attacker is attempting to manipulate the headers such as *user agent (UA)*, *referrer*, and *HTTP method*. The last example is a representation of invalid characters (mostly unintentionally) in API within web-request possibly by the human factor to reach the end-point/service.

No.	Suspicious log event examples
1	"GET /<script>PAYLOAD_INJECTED</script> HTTP/1.1" 403 - "UA"
2	"GET HTTP/1.1" 403 - "$jndi:ldap://PAYLOAD_INJECTED"
3	"GET HTTP/1.1" 403 - "!(()&&!|*|*| + PAYLOAD_INJECTED "
4	"GET $PAYLOAD_INJECTED + /windows/win.ini HTTP/1.1" 404 - "UA"
5	"GET XXX/HPI/H[I/XXX/ HTTP/1.1" 404 - "UA"

The main contributions of this paper are summarised below:

- Providing necessary background knowledge and identifying requirements/motivations (Section I) and challenges for data-driven-based web log analysis. (Section II).
- Review and categorize available analytical approaches, applications, and use-cases. (Section III).
- Discussing the open questions and challenges and point out several valid key research issues for WLAD (Section IV).

II. Background

A. Characteristics of Web server log

Mainly, web servers deploy the CLF (Common Log Format) for their server log files; there are other following types in which data can be separated into different logs (instead of being combined into a single file), as it is demonstrated in taxonomy in Fig. 2.

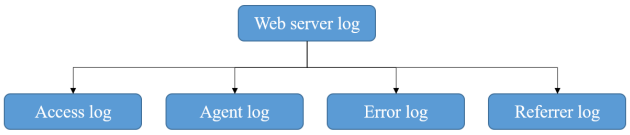

Fig. 2. Taxonomy of Web server/proxy logs.

Every request is sent over the resource/website (pages, APIs) to the web server, and the response is sent back to the requester. Meanwhile, a log message is stored or logged in the web server log file, maintaining a page request history.

Every web-app service has a Uniform Resource Locator (URL), which could be created either after clicking a link on a web page, bookmark/email, or by entering the URL directly into the address bar. Then the protocol is connected to the domain name, and the domain name is connected to the file path. In RESTful systems, clients send requests to retrieve or modify resources, and servers send responses to these requests. The anatomy of the full web server log includes information about the *client IP address*, request *date/time*, requested *page*, HTTP *code*, HTTP *method/verb*, *bytes* served, *user agent (UA)*, and *referrer*. The General Anatomy of the URL mainly consists of *protocol*, the *domain name*, the *file path*, and *URL parameters/query strings* within the RESTful API query. The anatomy of full Web server logs, as well as general URL, are shown in Fig. 3 and Fig. 4, respectively.

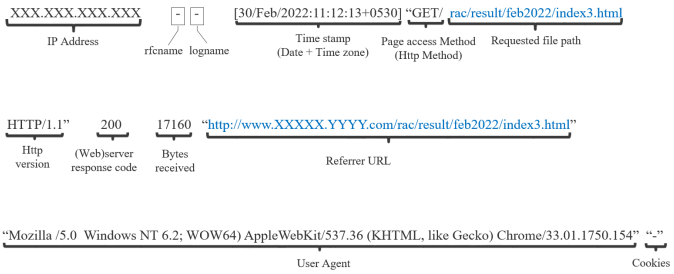

Fig. 3. Anatomy of full Web server/proxy logs.

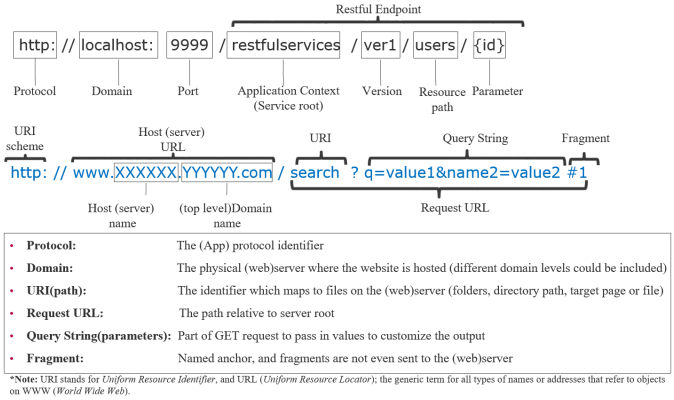

Fig. 4. General URL Anatomy.

III. Web Log Analytical Approaches

The outlier detection is well-researched, and Anomaly detection on textual data such as logs has been researched. This paper focuses on the context of Data-driven & AI-based solutions for WLAD. The following approaches have been explored and categorized based on underlying detection techniques. Finally, the most related research information formed into the Table II with different perspectives.

A. Underlying Detection Techniques

1) Rule-based Models: Despite the fact that rule-based systems are effective in detecting the known anomalies for which rules are given, they fail to adapt to the evolving network environment. Moreover, the detection rate of anomalies using different rule-based approaches depends heavily on network administrators' expertise and domain knowledge about the known attacks [12]. Due to the fact that unknown attacks often have no designed rules by security experts, they may remain undetected by such systems [13]. Therefore to cover this shortcoming of rule-based systems, various machine learning-based algorithms have been proposed in recent years for Log-based Anomaly Detection [14, 15, 16, 17, 18, 19, 20, 21].

2) Statistical Models: In earlier years, some statistical techniques were utilized to develop WLAD systems. [22] proposed a quarantine service system that generates signature queries (using Regex [23]) to detect anomalous HTTP queries received from malicious users in log files of the web server. He offered two methods to analyze the value of the query parameters based on the *length* and *character* distribution to serve the detection model.

3) Supervised and Unsupervised ML Models: Supervised learning methods rely on tags in which data has been labeled, while unsupervised learning methods are based on clustering [24, 25] and invariant mining. Supervised learning has a very good effect in detecting known malicious behavior or abnormal state, but it cannot detect unknown attacks, as it depends on prior knowledge. The unsupervised method can be used instead to detect unknown exceptions, but most of the methods need to improve their accuracy in the absence of label/ground truth information or anomaly rate within data for WLAD.

Unsupervised techniques can be used to uncover hidden structures, like finding groups of events with similar patterns, but it's difficult to implement and is not used as supervised learning. Zhao et al. proposed a novel feature-extracting mechanism, including log-line tokenization using an LSTM-based anomaly detection approach to identify attacks on two widely-used datasets for device access log and net access log. They showed that their proposed model outperforms one-class SVM, GMM, and Principal Components Analysis (PCA).[26] Apart from many studies for time-series analysis for clustering that have been researched so far, [24] summarize recent studies on log data utilizing clustering techniques that have been proposed lately. Earlier they researched string clustering [25] and proposed a dynamic log file anomaly detection methodology that incrementally groups log lines within time windows.

4) Deep Hybrid Models (DHM): Typically most of the approaches cannot handle unknown log types without taking advantage of the log semantic information. So most existing web log-based anomaly detection methods use a log parser to get log event indexes or event templates and then utilize machine learning methods to detect anomalies. Lv et al. proposed ConAnomaly, a log-based anomaly detection model composed of a log sequence encoder (log2vec) based on the Word2vec model and multi-layer Long Short Term Memory Network (LSTM) [27]. That captures semantic information in the log but also leverages log sequential relationships. Another approach proposed by Liu et al. is a heterogeneous graph embedding-based modularized method that converts log entries into a heterogeneous graph [28]. Wang et al. proposed an offline feature extraction model so-called LogEvent2vec, which takes the log event as input of word2vec to extract the relevance between log events and vectorize log events directly. [29, 30, 31] Also, Laskar et al. proposed an approach that combines the Isolation Forest with the *k*-Means algorithm, so-called the *IForest-KMeans* model for anomaly detection to detect anomalies in Big-data.[13] The most recent study [32] shows promising results for WLAD, passing NLP-tokenized feature vectors to the Tree-based EAD model, so-called ELSV.

B. Feature Representation

Mainly, the ML-based approaches for WLAD are very similar to malicious URL detection (MUD), which comprise two steps: first, to obtain lexical and host-based feature selection, and second, to use these features for training the predictive model to detect malicious URLs [33]. Also, we should take care of external dependency to acquire information, the associated time cost concerning feature collection and feature preprocessing, and the dimensionality of the features obtained. Lexical features are very efficient to collect, as they are basically direct derivatives of the URL string within web request (subject to web server log), but they have some limitations like failing to efficiently detect new words (unseen features) due to lack of a knowledge-based system (KBS) augmentation to examine web request or URL [33]. Another point is mostly they are high-dimensional because they are all stored as Bag-of-Words features and feature size consequently affects the training and test-time. Despite NLP-based tokenization and semantic embedding, the most recent studies researched a different combination to feed their model, such as Byte-Pair Encoding (BPE) [34]. Fig. 5 shows anomaly detection approaches in related work based on major components (Model and Feature Map).

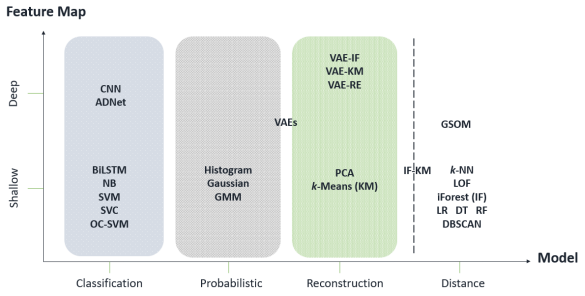

Fig. 5. Anomaly detection approaches arranged in the plane spanned by two major components (Model and Feature Map) point of view distinguished with three main groups of approaches (Classification, Probabilistic, and Reconstruction) that all formulate Shallow and Deep models. These three groups are complemented by purely distance-based methods for shortlisted related works in Table II. The Decision Tree (DT) and its successors, such as iForest (IF) and RandomForest (RF), are invariant to the choice of distance metric (since they do not use a distance metric in the first place), but randomized splitting/partitioning instances recursively [35].

C. Applications Scenarios

Finding anomalous incidents or logs in the logged data that might represent malicious activity has a wide range of applications in cybersecurity and network security, with common characteristics that are interesting to the analyst. There are some use-cases that explored user behaviour analytics, and user profiling [36] for User Behaviour Anomaly (UBA) detection [37, 38, 39], a behavioural IDS [40], or Malicious Behavior Detection [41]. Additionally, Time-series Analysis (TSA) use-cases for performance and optimization of web-server/web-dispatcher [42, 43].

IV. Discussion and Open Questions

A. Outlierness Vs. Maliciousness

One of the important key points while using outlier detection methods in real-life cybersecurity application problems is understanding what outlierness is. Often outlier detection algorithms are used to identify outliers based on *outlier score*, which assigns an anomalous score by the algorithm to each data instance to indicate the degree of outlierness of each measurement, and it is parametric-depended. So it means in the absence of data anomaly rate (sometimes so-called contamination ratio), often needs to set an anomalous threshold *threshold* as one of the algorithm's hyperparameters. The used model could predict the different values as anomalous/outlier with various configurations with respect to the model generalization condition and cause a considerable amount of false-positive (FP) due to the quality of the anomaly detection process. In practice, the characteristics of security-related data traffic are typically non-stationary, which means traffic distributions can vary over time [55]. Also, the superiority of adaptive thresholds over static in cybersecurity has been researched [56]. Thus better to consider Self-Adaptive Threshold [57] approaches within (self-)Adaptive Anomaly Detection [58] for WLAD problems that regularly adapt to newer log patterns to ensure accurate anomaly detection.

On the other hand, It is hard to characterize maliciousness, and harder yet to develop a maliciousness index to apply to models, and there is a research gap for assessment and measurement metrics for cybersecurity-related maliciousness [59].

Bad entries problems can be a good example of this issue, considering there could be invalid characters within a web request as it is shown in Table I and Extract, Transform and Load (ETL) tools parsed data differently (depending on ETL configuration or possible bugs in ETL pipeline), which could affect on detection algorithm performance. These human factor outliers may generate due to unwanted typo mistakes unintentionally, making detecting malicious incidents challenging in micro-level clustering.

B. Feature Engineering (FE) & Contextualization

Web server logs and Web log data analysis are important in intrusion detection, and different ML techniques have been assigned for abnormal detection. However, compared to abundant research on machine learning, ways to extract features from log data are still under research. Considering the text-based nature of web server logs, applying NLP approaches helps to provide better feature representation for learning models [60]. This approach to various downstream machine learning algorithms has been applied and proved its usefulness[47]. In the absence of labeled web-server log data, limited research has been done over out-of-date public datasets with noticeable accuracy can be found in Table II. However, in practice scale of this data is in the scope of Big-Data or stream data which turns this task much more challenging due to different Feature Engineering stages for detecting different security threads as well as Web Application Vulnerabilities [4]. It has been shown for the tasks that need to detect injections, e.g., Cross-Site Scripting (XSS), SQL Injection (SQLI), etc. over HTTP requests; it is also useful to consider the feature extraction stage along with feature selection to extract meaningful features out of specific categorical features over URL parameters (addressed in the Table I). Those extracted features can potentially represent the length of payloads better besides numerical features [48]. Thus considering the synthetic nature of the web server log feature-wisely, even though we used unsupervised methods, adapting suitable feature engineering (feature selection/extraction) depending on the task is inevitable for the analysis and parsing of unstructured cybersecurity incident data. Leveraging NLP-based tokenization and other semantic embeddings such as word-embedding and character-embedding to create a meaningful feature map is a key point [61].

C. High-dimensional Data

Typically for low-dimensional tasks, e.g., 1-2 dimensional data, identifying outliers/anomalies could be resolved by plotting the data (points far away from the rest). Algorithms generally consider for anomaly detection in low dimensional data are not suitable for high dimensional data. Thus, unsupervised anomaly detection is close to being a hopeless task due to the *curse of dimensionality* [62], which - in the sense of anomaly detection - means that every point eventually becomes an outlier. The problems of anomaly detection in high-dimensional data are 3-fold to detect: (a) global anomalies, (b) local anomalies, and (c) micro-clusters of anomalies [63].

Global anomalies can be detected easily since they are very different from dense areas with respect to their attributes. In contrast, a local anomaly is only an anomaly when it is distinct from, and compared with, its local neighborhood. Micro-clusters of anomalies have been paying little attention to this problem relative to the other two categories. Recently, some high-dimensional outlier detection algorithms have addressed this problem to some extent by grouping instances together by selecting a representative member from each cluster before the Nearest Neighbor (NN) distances computation over them [64]. However, it suffers from a few limitations that significantly hinder its ability to detect anomalies under certain situations, which are addressed and improved using the STRAY ("Search and TRace AnomalY") algorithm, including k-Nearest Neighbor (KNN) distance calculations [65].

Table II. Summary of popular anomaly detection algorithms including machine learning and deep learning techniques used for **WLAD** domain as well as malicious HTTPS requests problems in the cybersecurity domain.

Study	Problem domain	Feature representation NLP/Data Eng. method	Classification/Clustering detection technique	Dataset	Figure of merit
[34]	WLAD	tokenization:Bigrams + AE FastText-based features BPE + USE embeddings BPE + TF-IDF + SS + AE	OC-SVM iForests	HTTP CSIC 2010 ISCX IDS 2012	ACC_{IF} = 93.00%, FPR_{IF} = 10.00% $ACC_{OC\text{-}SVM}$ = 94.00%, $FPR_{OC\text{-}SVM}$ = 8.00%
[44]	Log Server Analytics via cluster analysis of WebLogic logs	different tokenization i) linguistic method ii) N-gram	Cosine similarity Growing Self-Organizing Map (GSOM)	Private datasets	not reported
[32]	WLAD	different tokenization i) Word2Vec ii) TextCNN	XGBoost + LightGBM + CatBoost	HTTP CSIC 2010	ACC_{ELSV} = 99.33%, $F1_{ELSV}$ = 98.70% AUC_{ELSV} = 99.74%
[45]	WLAD	Tokenization (Bi-grams) + i)Diffusion Maps (DM) incl. Nyström extension ii)Random Projection (RP) incl. Out-of-sample extension iii) Principal Component Analysis (PCA)	statistical model computing ($\mu, \sigma = 3$)	Private datasets	not reported
[46]	HTTP Request Parameter Anomaly Detection (RPAD) Attack behavior detection	frequency-based N-gram PCA	SVM Adaptive learning so-called (AMOD) using SVM HYBRID	HTTP CSIC 2010 Private datasets	ACC_{AMOD} = 99.50%, FPR_{AMOD} = 0.001%
[47]	WLAD	Transformation by Regular Expression (RE)+ one-hot-encoding (OHE)	11-layer ADNet	Private datasets	$ACC_{\alpha=0.25}$ = 87.60% $F1_{\alpha=0.25}$ = 87.00%
[48]	Web attack detection HTTP attack detection	customized categories: type, length, token number, encoding type Statistical Preprocessing	DBSCAN, Apache-Scalp tools	Private datasets	$ACC_{RPAD\text{-}DBSCAN}$ = 99.98%, $ACC_{RPAD\text{-}SCALP}$ = 99.98%, $ACC_{RPAD\text{-}LOF}$ = 99.67%, $ACC_{RPAD\text{-}IF}$ = 81.68%,
[49]	WLAD	N-gram + TF-IDF	Isolation Forest K-means SVM	Private datasets	PR_{IF} = 94.00% RE_{IF} = 92.00% $F1_{IF}$ = 93.00%
[22]	WLAD	Signature-based analysis over the query parameters based on the i) *length* and ii) *character* distribution using Regular Expressions	statistical model computing (μ, σ)	Private datasets	not reported
[26]	LAD	Tokenization along with Bidirectional Event Mode (BEM) by: i) feature Vector ii) Characters	single-layer BiLSTM	LANL(Kent 2016), R6.2	$AUC_{Knet2016}$ = 98.40%, $AUC_{R6.2}$ = 91.30%
[50]	Web scans & attack detection	Analyze path & query of requested URI	Rule-based detection model	Generated dataset Apache-http-logs	ACC = 99.38%, F1 = 85.71%
[33]	Malicious Website	Analyze feature importance	Linear SVC, LR, RF	A large-scale real dataset of data-driven Web applications	ACC = 99.03%, FP = 1%
[51]	Malicious Website Detection using UE	URL embedding (UE) using Huffman tree instead of Feature Engineering (FE)	DT, LR, CNN, NB, and SVM	360 NetLab, Alexa datasets	ACC = 99.03%, FP = 1%
[52]	Enhance IDS to detect HTTP attacks	Analyze feature importance	Naïve Bayes classifier	NSL-KDD dataset	ACC = 99.03%, FP = 1%
[53]	Malicious URL Detection (MUD)	host-based features, domain-based features and lexical features	classification based on association (CBA): Logistic Regression (LR)	PhishTank, Github, and Kaggle websites	ACC_{LR} = 91.0%, FPR_{LR} = 78.0%
[54]	HTTP Requests Analysis for Anomaly Detection of Web Attacks	N-gram, n 1, 2, 3 over Web resources Attribute values and User agents	SVDD, K-means, DBSCAN, SOM, and LOF	Private datasets	ACC_{SVDD} = 99.2%, $AUC_{K\text{-}means}$ = 100%, AUC_{DBSCAN} = 97.5%, AUC_{LOF} = 97.5%

D. Big-Data Analytics

With the advent of Big-Data, the processing efficiency of anomaly detection techniques becomes increasingly complex, specifically When the underlying probability distribution is unknown along with high data size. In other words, the *volume* feature of Big-Data stresses the storage, memory, and computation, and these requirements need to be increased [66]. The problem is most outlier detection algorithms can perform on small data size computation-wisely, whilst outlier detection tasks through Big-Data require distributed processing to scale out. In this regard, some outlier detection algorithms are not developed for distributed systems yet. The majority of today's ML-based algorithms are designed for single-thread computation, whereas real-world Big-Data problems require distributed systems.

Following recent surveys [67] for general systematic log anomaly (not only cyber-related logs) shows that Word-embedding has proven significant results for capturing semantic information from log messages. The unsupervised deep learning models such as Variational Auto-encoder (VAE) and Deep Variational Auto-encoder (DVAE) provided promising results and it has already been shown that the DL algorithms perform better subject to the amount of data during the learning process compared to classic ML models plateaus [68].

E. Imbalanced Data Evaluation

Typically, evaluating unsupervised anomaly detection techniques provides us with a better understanding of their performance in the presence of labeled data. There are two types

of errors we can consider for an anomaly detection model: (i) False Positive (FP) is about predicting a true normal data as being abnormal, causing high False Positive Rate (FPR) or *sensitivity*, and (ii) False Negative (FN) predicting a true anomaly as being normal and it means that model mispredict anomaly, which result in high False Positive Rate (FPR) or *specificity*. There is no rule of thumb to handle these two types of errors since it is vary depending on the application. Unlike medic data processing, in cyber-related monitoring tasks that involve large amounts of data, it can be more desirable to have a low FP, usually at the expense of a higher miss rate (TNR). Given the anomaly score (that indicates the "degree of anomalousness") $s : X \longrightarrow R$ of the model, where the decision threshold τ, can be achieved by following *Decision* function:

$$Decision = \begin{cases} outlier\ (positive\ class)\ if \quad s(x) \geqslant \tau, \\ inlier\ (negative\ class) \quad if \quad s(x) \geqslant \tau, \end{cases} \quad (1)$$

Thus τ needs to be calibrated using domain experts' knowledge for the specific application to minimize type I and type II errors (FP and FN). The Area Under the Receiver Operating Characteristic curve (AUROC), usually called Area Under the Curve (AUC), is used in classification analysis in order to determine which of the used models predicts the classes best by interpreting the probability that a random positive sample will have a higher score than a random negative sample. In other words, It provides an evaluation measure that considers the full range of decision thresholds on a given test-set [69]. The Receiver Operating Characteristic (ROC) curve plots all (false alarm rate, recall)-pairs that result from iterating over all possible thresholds covering every test set decision split, and the area under this curve is the AUC measure [70]. Suppose A and B are the distributions of scores the model produces for data points that are actually in the positive and negative classes, respectively, and τ denotes the cutoff threshold. Thus are related via the following relationship:

$$AUC = \int_0^1 TPR(x)dx = \int_0^1 P(A > \tau(x))dx \quad (2)$$

The downside of the AUC is that it can produce overly optimistic scores in the case of highly imbalanced data. Therefore the Area Under the Precision-Recall Curve (AUPRC) is more informative and appropriate to use when precision is more relevant than the false alarm rate [71, 72]. A common robust way to compute the AUPRC is via Average Precision (AP) [73]. One caveat of the AUPRC (or AP) is that the random guessing baseline is given by the fraction of anomalies in the test-set data and thus varies use-case by use-case. This issue makes the AUPRC (or AP) harder to interpret and less comparable over different application scenarios, but in those scenarios where the data is not highly imbalanced, the AUROC and AUPRC (or AP) measurements show the same trends [71, 74, 75]. Due to the point that AUC has been used most of the time to mean AUROC, which is a bad practice since it has been shown AUC is ambiguous (could be any curve) while AUROC is not [76]. In short, the ROC curve is suitable when the observations are balanced between each class or how good the model performs with no knowledge of the class imbalance, whereas the precision-recall (PR) curve is appropriate for imbalanced datasets or it uses the estimated class imbalance baseline to answer how good the model performs, given imbalanced data.

Although most of the related studies we shortlisted in Table II report high accuracy (ACC), however, there is incomplete information concerning error types, which is better to report the F1 score to understand better the used model performance, especially for imbalanced data tasks for further investigations to improve. Lately, there has been another straightforward evaluation approach to demonstrate or compare AD algorithm performance by calculating anomalies that could be found in the top *n* data in addition to ROC-curve or PR-curve plots [77].

V. Conclusion and Future Works

This paper solely reviewed the literature on web-server logs analysis and WLAD, that one can also expand web-server logs to other similar structure logs such as HTTP Logs, Proxy Logs, etc. This study aimed to investigate and identify the various anomaly detection approaches for web server logs and evaluate their suitability and feasibility in the big data realm. For each category of anomaly detection techniques, we present the assumption regarding the notion of normal and anomalous data along with its strength and weakness. In addition, we discussed some of the main challenges and open topics in the context of WLAD, particularly, the ability to learn from outlierness leading toward maliciousness, as outlierness doesn't necessarily translate into maliciousness. These assumptions can be used as guidelines to assess the technique's effectiveness in the web-application domain and to detect right malicious logs with low false-positive rates. The lack of labeled web log data makes provision of the classic learnings challenging for web log anomaly detection on skewed class distribution; nevertheless, meaningful feature mapping besides the novel learnings like positive-unlabeled learning and active learning as well as self-supervised learning could improve weblog-specific augmentation. New learning-based anomaly detections are still active research, and a possible future work would be to extend and update this survey as more sophisticated techniques are proposed.

References

[1] Acunetix. Acunetix 2020 web application vulnerability report.

[2] Adem Tekerek. A novel architecture for web-based attack detection using convolutional neural network. *Computers & Security*, 100:102096, 2021.

[3] Sandy Carielli. The state of application security, 2021, 2021.

[4] Dave Wichers. Owasp top-10 2021. *OWASP Foundation, February*, 2021. Accessed: 28-02-2022.

[5] CAPEC Community. Common attack pattern enumeration and classification, 2022. Accessed: 2022-05-18.

[6] Magnus Almgren, Hervé Debar, and Marc Dacier. A lightweight tool for detecting web server attacks. In *NDSS*. Citeseer, 2000.

[7] Martin Roesch et al. Snort: Lightweight intrusion detection for networks. In *Lisa*, volume 99, pages 229–238, 1999.

[8] Christopher Kruegel and Giovanni Vigna. Anomaly detection of web-based attacks. In *Proceedings of the 10th ACM conference on Computer and communications security*, pages 251–261, 2003.

[9] Pavol Lupták. Bypassing web application firewalls. In *Proceedings of 6th International Scientific Conference on Security and Protection of Information*, pages 79–88, 2011.

[10] Stefan Prandl, Mihai Lazarescu, and Duc-Son Pham. A study of web application firewall solutions. In *International conference on information systems security*, pages 501–510. Springer, 2015.

[11] Maryam M Najafabadi, Taghi M Khoshgoftaar, Chad Calvert, and Clifford Kemp. User behavior anomaly detection for application layer ddos attacks. In *2017 IEEE International Conference on Information Reuse and Integration (IRI)*, pages 154–161. IEEE, 2017.

[12] Marina Thottan and Chuanyi Ji. Anomaly detection in ip networks. *IEEE Transactions on signal processing*, 51(8):2191–2204, 2003.

[13] Md Tahmid Rahman Laskar, Jimmy Xiangji Huang, Vladan Smetana, Chris Stewart, Kees Pouw, Aijun An, Stephen Chan, and Lei Liu. Extending isolation forest for anomaly detection in big data via k-means. *ACM Transactions on Cyber-Physical Systems (TCPS)*, 5(4):1–26, 2021.

[14] Markus Fält, Stefan Forsström, and Tingting Zhang. Machine learning based anomaly detection of log files using ensemble learning and self-attention. In *2021 5th International Conference on System Reliability and Safety (ICSRS)*, pages 209–215. IEEE, 2021.

[15] Lakshmi Geethanjali Mandagondi. Anomaly detection in log files using machine learning techniques. Master's thesis, 2021.

[16] Qingwei Lin, Hongyu Zhang, Jian-Guang Lou, Yu Zhang, and Xuewei Chen. Log clustering based problem identification for online service systems. In *2016 IEEE/ACM 38th International Conference on Software Engineering Companion (ICSE-C)*, pages 102–111. IEEE, 2016.

[17] Min Du, Feifei Li, Guineng Zheng, and Vivek Srikumar. Deeplog: Anomaly detection and diagnosis from system logs through deep learning. In *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, pages 1285–1298, 2017.

[18] Weibin Meng, Ying Liu, Yichen Zhu, Shenglin Zhang, Dan Pei, Yuqing Liu, Yihao Chen, Ruizhi Zhang, Shimin Tao, Pei Sun, et al. Loganomaly: Unsupervised detection of sequential and quantitative anomalies in unstructured logs. In *IJCAI*, volume 19, pages 4739–4745, 2019.

[19] Xu Zhang, Yong Xu, Qingwei Lin, Bo Qiao, Hongyu Zhang, Yingnong Dang, Chunyu Xie, Xinsheng Yang, Qian Cheng, Ze Li, et al. Robust log-based anomaly detection on unstable log data. In *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pages 807–817, 2019.

[20] Lin Yang, Junjie Chen, Zan Wang, Weijing Wang, Jiajun Jiang, Xuyuan Dong, and Wenbin Zhang. Plelog: Semi-supervised log-based anomaly detection via probabilistic label estimation. In *2021 IEEE/ACM 43rd International Conference on Software Engineering: Companion Proceedings (ICSE-Companion)*, pages 230–231. IEEE, 2021.

[21] Vannel Zeufack, Donghyun Kim, Daehee Seo, and Ahyoung Lee. An unsupervised anomaly detection framework for detecting anomalies in real time through network system's log files analysis. *High-Confidence Computing*, 1(2):100030, 2021.

[22] R Faradzhullaev. Analysis of web server log files and attack detection. *Automatic Control and Computer Sciences*, 42(1):50–54, 2008.

[23] Jeffrey EF Friedl. *Mastering regular expressions*. " O'Reilly Media, Inc.", 2006.

[24] Max Landauer, Florian Skopik, Markus Wurzenberger, and Andreas Rauber. System log clustering approaches for cyber security applications: A survey. *Computers & Security*, 92:101739, 2020.

[25] Max Landauer, Markus Wurzenberger, Florian Skopik, Giuseppe Settanni, and Peter Filzmoser. Dynamic log file analysis: An unsupervised cluster evolution approach for anomaly detection. *computers & security*, 79:94–116, 2018.

[26] Zhijun Zhao, Chen Xu, and Bo Li. A lstm-based anomaly detection model for log analysis. *Journal of Signal Processing Systems*, 93(7):745–751, 2021.

[27] Dan Lv, Nurbol Luktarhan, and Yiyong Chen. Conanomaly: Content-based anomaly detection for system logs. *Sensors*, 21(18):6125, 2021.

[28] Fucheng Liu, Yu Wen, Dongxue Zhang, Xihe Jiang, Xinyu Xing, and Dan Meng. Log2vec: A heterogeneous graph embedding based approach for detecting cyber threats within enterprise. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 1777–1794, 2019.

[29] Yigit Erkal, Mustafa Sezgin, and Sedef Gunduz. A new cyber security alert system for twitter. In *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, pages 766–770. IEEE, 2015.

[30] Çağrı B Aslan, Rahime Belen Sağlam, and Shujun Li. Automatic detection of cyber security related accounts on online social networks: Twitter as an example. In *Proceedings of the 9th International Conference on Social Media and Society*, pages 236–240, 2018.

[31] Sheikh Shah Mohammad Motiur Rahman, Khalid Been Md Biplob, Md Rahman, Kaushik Sarker, Takia Islam, et al. An investigation and evaluation of n-gram, tf-idf and ensemble methods in sentiment classification. In *International Conference on Cyber Security and Computer Science*, pages 391–402. Springer, 2020.

[32] Wei Wan, Xin Shi, Jinxia Wei, Jing Zhao, and Chun Long. Elsv: An effective anomaly detection system from web access logs. In *2021 IEEE International Performance, Computing, and Communications Conference (IPCCC)*, pages 1–6. IEEE, 2021.

[33] Sajid Anwar, Feras Al-Obeidat, Abdallah Tubaishat, Sadia Din, Awais Ahmad, Fakhri Alam Khan, Gwanggil Jeon, and Jonathan Loo. Countering malicious urls in internet of things using a knowledge-based approach and a simulated expert. *IEEE Internet of Things Journal*, 7(5):4497–4504, 2019.

[34] Nitesh Suresh Sehwani. No features needed: Using bpe sequence embeddings for web log anomaly detection. In *Proceedings of the 2022 ACM on International Workshop on Security and Privacy Analytics*, pages 78–85, 2022.

[35] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *2008 eighth ieee international conference on data mining*, pages 413–422. IEEE, 2008.

[36] Min Chen, Ali A Ghorbani, et al. A survey on user profiling model for anomaly detection in cyberspace. *Journal of Cyber Security and Mobility*, 8(1):75–112, 2019.

[37] Jingwen You, Xiaojuan Wang, Lei Jin, and Yong Zhang. Anomaly detection in the web logs using user-behaviour networks. *International Journal of Web Engineering and Technology*, 14(2):178–199, 2019.

[38] Jianjiang Lu, Baowen Xu, and Hongji Yang. Matrix dimensionality reduction for mining web logs. In *Proceedings IEEE/WIC International Conference on Web Intelligence (WI 2003)*, pages 405–408. IEEE, 2003.

[39] Xiong Luo, Xiaoqiang Di, Xu Liu, Hui Qi, Jinqing Li, Ligang Cong, and Huamin Yang. Anomaly detection for application layer user browsing behavior based on attributes and features. In *Journal of Physics: Conference Series*, volume 1069, page 012072. IOP Publishing, 2018.

[40] Grant Pannell and Helen Ashman. User modelling for exclusion and anomaly detection: a behavioural intrusion detection system. In *International Conference on User Modeling, Adaptation, and Personalization*, pages 207–218. Springer, 2010.

[41] Mahdi Rabbani, Yongli Wang, Reza Khoshkangini, Hamed Jelodar, Ruxin Zhao, Sajjad Bagheri Baba Ahmadi, and Seyedvalyallah Ayobi. A review on machine learning approaches for network malicious behavior detection in emerging technologies. *Entropy*, 23(5):529, 2021.

[42] Jundi Wang and Zhao Kai. Performance analysis and optimization of nginx-based web server. In *Journal of Physics: Conference Series*, volume 1955, page 012033. IOP Publishing, 2021.

[43] Mark S Squillante, David D Yao, and Li Zhang. Web traffic modeling and web server performance analysis. In *Proceedings of the 38th IEEE Conference on Decision and Control (Cat. No. 99CH36304)*, volume 5, pages 4432–4439. IEEE, 1999.

[44] Fedotov Daniil. Log server analytics. Master's thesis, Czech Technical University in Prague, Faculty of Information Technology, 2022.

[45] Antti Juvonen, Tuomo Sipola, and Timo Hämäläinen. Online anomaly detection using dimensionality reduction techniques for http log analysis. *Computer Networks*, 91:46–56, 2015.

[46] Ying Dong, Yuqing Zhang, Hua Ma, Qianru Wu, Qixu Liu, Kai Wang, and Wenjie Wang. An adaptive system for detecting malicious queries in web attacks. *Science China Information Sciences*, 61(3):1–16, 2018.

[47] Junlang Zhan, Xuan Liao, Yukun Bao, Lu Gan, Zhiwen Tan, Mengxue Zhang, Ruan He, and Jialiang Lu. An effective feature representation of web log data by leveraging byte pair encoding and tf-idf. In *Proceedings of the ACM Turing Celebration Conference-China*, pages 1–6, 2019.

[48] Yizhen Sun, Yiman Xie, Weiping Wang, Shigeng Zhang, Yuxi Wu, and Jingchuan Feng. Rpad: An unsupervised http request parameter anomaly detection method. In *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pages 1215–1221. IEEE, 2020.

[49] Wei Zhang and Lijun Chen. Web log anomaly detection based on

isolated forest algorithm. In *2019 IEEE 14th International Conference on Intelligent Systems and Knowledge Engineering (ISKE)*, pages 755–759. IEEE, 2019.

[50] Merve Baş Seyyar, Ferhat Özgür Çatak, and Ensar Gül. Detection of attack-targeted scans from the apache http server access logs. *Applied computing and informatics*, 14(1):28–36, 2018.

[51] Xiaodan Yan, Yang Xu, Baojiang Cui, Shuhan Zhang, Taibiao Guo, and Chaoliang Li. Learning url embedding for malicious website detection. *IEEE Transactions on Industrial Informatics*, 16(10):6673–6681, 2020.

[52] MM Abd-Eldayem. Original article a proposed http service based ids. *Egypt. Informatics J*, 15(1):13–24, 2014.

[53] Rupa Chiramdasu, Gautam Srivastava, Sweta Bhattacharya, Praveen Kumar Reddy, and Thippa Reddy Gadekallu. Malicious url detection using logistic regression. In *2021 IEEE International Conference on Omni-Layer Intelligent Systems (COINS)*, pages 1–6. IEEE, 2021.

[54] Mikhail Zolotukhin, Timo Hämäläinen, Tero Kokkonen, and Jarmo Siltanen. Analysis of http requests for anomaly detection of web attacks. In *2014 IEEE 12th International Conference on Dependable, Autonomic and Secure Computing*, pages 406–411. IEEE, 2014.

[55] Joshua Oldmeadow, Siddarth Ravinutala, and Christopher Leckie. Adaptive clustering for network intrusion detection. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 255–259. Springer, 2004.

[56] Matan Levi, Yair Allouche, and Aryeh Kontorovich. Advanced analytics for connected car cybersecurity. In *2018 IEEE 87th vehicular technology conference (VTC spring)*, pages 1–7. IEEE, 2018.

[57] Haiyan Wang. Anomaly Detection of Network Traffic Based on Prediction and Self-Adaptive Threshold. *International Journal of Future Generation Communication and Networking*, 8:205–214, 2015.

[58] Maged Abdelaty, Roberto Doriguzzi-Corin, and Domenico Siracusa. Aads: A noise-robust anomaly detection framework for industrial control systems. In *International Conference on Information and Communications Security*, pages 53–70. Springer, 2019.

[59] Zoe M King, Diane S Henshel, Liberty Flora, Mariana G Cains, Blaine Hoffman, and Char Sample. Characterizing and measuring maliciousness for cybersecurity risk assessment. *Frontiers in Psychology*, 9:39, 2018.

[60] Armando J Ochoaq and Mark A Finlayson. Analysis and parsing of unstructured cyber-security incident data: poster. In *Proceedings of the 12th Conference on Security and Privacy in Wireless and Mobile Networks*, pages 345–346, 2019.

[61] PV Veena, M Anand Kumar, and KP Soman. An effective way of word-level language identification for code-mixed facebook comments using word-embedding via character-embedding. In *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 1552–1556. IEEE, 2017.

[62] Anitha Ramchandran and Arun Kumar Sangaiah. Unsupervised anomaly detection for high dimensional data—an exploratory analysis. In *Computational intelligence for multimedia big data on the cloud with engineering applications*, pages 233–251. Elsevier, 2018.

[63] Markus Goldstein and Seiichi Uchida. A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data. *PloS one*, 11(4):e0152173, 2016.

[64] Leland Wilkinson. Visualizing big data outliers through distributed aggregation. *IEEE transactions on visualization and computer graphics*, 24(1):256–266, 2017.

[65] Priyanga Dilini Talagala, Rob J Hyndman, and Kate Smith-Miles. Anomaly detection in high-dimensional data. *Journal of Computational and Graphical Statistics*, 30(2):360–374, 2021.

[66] Srikanth Thudumu, Philip Branch, Jiong Jin, and Jugdutt Jack Singh. A comprehensive survey of anomaly detection techniques for high dimensional big data. *Journal of Big Data*, 7(1):1–30, 2020.

[67] Rakesh Bahadur Yadav, P Santosh Kumar, and Sunita Vikrant Dhavale. A survey on log anomaly detection using deep learning. In *2020 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions)(ICRITO)*, pages 1215–1220. IEEE, 2020.

[68] Andrew Ng. Machine learning yearning: Technical strategy for ai engineers in the era of deep learning draft version 0.5. *Harvard Bus. Publishing, Boston, MA, USA, Tech. Rep*, 2016.

[69] Andrew P Bradley. The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern recognition*, 30(7):1145–1159, 1997.

[70] Tom Fawcett. An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874, 2006.

[71] Jesse Davis and Mark Goadrich. The relationship between precision-recall and roc curves. In *Proceedings of the 23rd international conference on Machine learning*, pages 233–240, 2006.

[72] Faruk Ahmed and Aaron Courville. Detecting semantic anomalies. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3154–3162, 2020.

[73] Kendrick Boyd, Kevin H Eng, and C David Page. Area under the precision-recall curve: point estimates and confidence intervals. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 451–466. Springer, 2013.

[74] Jakob Sternby, Erik Thormarker, and Michael Liljenstam. Anomaly detection forest. In *ECAI 2020*, pages 1507–1514. IOS Press, 2020.

[75] Lukas Ruff. *Deep one-class learning: a deep learning approach to anomaly detection*. Technische Universitaet Berlin (Germany), 2021.

[76] Marc Claesen, Jesse Davis, Frank De Smet, and Bart De Moor. Assessing binary classifiers using only positive and unlabeled data. *arXiv preprint arXiv:1504.06837*, 2015.

[77] Kanatoko Satoh Tadashi. Xbos-anomaly-detection. https://github.com/Kanatoko/XBOS-anomaly-detection.git, 2021.