Computers & Security 124 (2023) 102997

Contents lists available at ScienceDirect

Computers & Security

journal homepage: www.elsevier.com/locate/cose

A critical review of the techniques used for anomaly detection of HTTP-based attacks: taxonomy, limitations and open challenges

Jesús E. Díaz-Verdejo[b,*], Rafael Estepa Alonso[a], Antonio Estepa Alonso[a], German Madinabeitia[a]

[a] Dpt. of Telematics University of Seville c Seville Spain
[b] Dpt. Signal Theory, Telematics and Communications - CITIC University of Granada c/ Periodista Daniel Saucedo Aranda, s/n Granada 18071 Spain

ARTICLE INFO

Article history:
Received 15 July 2022
Revised 19 October 2022
Accepted 26 October 2022
Available online 29 October 2022

Keywords:
Cybersecurity
Anomaly-based intrusion detection
Web attacks

ABSTRACT

Intrusion Detection Systems (IDSs) and Web Application Firewalls (WAFs) offer a crucial layer of defense that allows organizations to detect cyberattacks on their web servers. Academic research overwhelmingly suggests using anomaly detection techniques to improve the performance of these defensive systems. However, analyzing and comparing the wide range of solutions in the scientific literature is challenging since they are typically presented as isolated (unrelated) contributions, and their results cannot be generalized. We believe that this impairs the industry's adoption of academic results and the advancement of research in this field. This paper aims to shed light on the literature on anomaly-based detection of attacks that use HTTP request messages. We define a novel framework for anomaly detection based on six data processing steps grouped into two sequential phases: preprocessing and classification. Based on this framework, we provide a taxonomy and critical review of the techniques surveyed, emphasizing their limitations and applicability. Future approaches should take advantage of the syntax and semantics of the Uniform Resource Locator (URL), be scalable, and address their obsolescence. These aspects are frequently overlooked in the literature and pose a significant challenge in the current era of web services. For better comparability, authors should use adequate public datasets, follow a thorough methodology, and use appropriate metrics that fully show the pros and cons of the approach.

1. Introduction

Today's web services and websites are supported by millions of vulnerable web servers connected to the Internet that constitute an attractive swag to attackers. The extensive use of web services has spurred the number of web-facing computers on the Internet to more than 1 billion in Dec 2021 according to a recent survey new (2021a). As a result, the number of attacks targeted at web servers has increased exponentially in recent years new (2021b), and therefore has the risk level for many organizations since a compromised web server can be used as an attack vector for more complex attacks Husák et al. (2021). This justifies a rising interest in detecting attacks on web servers in academic and commercial contexts.

Traditionally, the detection of web attacks has been primarily based on intrusion detection systems (IDS) Agarwal and Hussain (2018) and WAFs (Web Application Firewalls) Pałka and Zachara (2011). IDSs are more generic and can examine either network or host data for attacks, while WAFs are restricted to analyzing the requests sent to a web server and tend to include filtering and blocking features. However, there is little difference between both when it comes to detecting web attacks, which can be attributed to historical reasons.

Most IDS and WAFs are based on signature detection (i.e., examining data for traces of known patterns). This type of detection is efficient for known attacks for which patterns have been previously identified. However, it is inefficient for zero-day attacks (which account for more than 30% according to some estimates Watchguard (2016)). For this reason, signature-based systems are frequently enhanced with the feature of anomaly detection Betarte et al. (2019); Hajj et al. (2021); Khraisat et al. (2019). Nevertheless, the success and adoption of the plethora of schemes suggested in the literature remain unclear. This could be partly attributable to the fact that works are frequently presented as isolated contributions rather than incremental advances and are often difficult to compare due to the diversity of the techniques used or

* Corresponding author.
E-mail addresses: jedv@ugr.es (J.E. Díaz-Verdejo), rafaestepa@us.es (R. Estepa Alonso), aestepa@us.es (A. Estepa Alonso), german@us.es (G. Madinabeitia).

https://doi.org/10.1016/j.cose.2022.102997
0167-4048/

flawed methodologies. Furthermore, practical aspects such as scalability or the validity of the results in a different testbed, are frequently overlooked by the present research.

This article critically reviews the scientific literature on anomaly detection in the information elements of an HTTP transaction (i.e., HTTP request and/or response) used for web attacks. These elements often include the URI field used to request a web resource, either exclusively or combined with other elements such as HTTP headers or body. The attacks that leverage these elements includes code injection, cross-site scripting or SQL injection (among others), and its detection is challenging due to wide variability. We exclude from our study anomaly detection schemes that disregard the content of HTTP messages (e.g., DoS attacks detection through traffic properties) or other web attack vectors such as form-jacking, drive-by downloads, etc.

Contributions The distinct contributions of this paper include:

- A novel framework for anomaly detection in HTTP requests based on a sequence of six data-processing steps grouped into two phases: preprocessing and classification. We particularly extend the preprocessing phase (often overlooked) that allows one to obtain the features that will be key in the results.
- A taxonomy of the techniques surveyed based on our framework. As such, our taxonomy is focused on the operations for selecting features (although we also cover classification).
- A critical review of the techniques and approaches for detecting anomalies in HTTP request messages, emphasizing their limitations and applicability.
- We identify and discuss several challenges and open issues that could interest the scientific community.

The remainder of this paper is as follows Section 2 provides a brief background and reviews related works. Section 3 describes our generic process for detecting anomalies in HTTP messages. Section 4 provides a taxonomy of the techniques found in the literature for each step of this process. Section 5 offers a critical analysis of the main limitations found in the literature and identifies the main open challenges in this research field. Finally, Section 6 concludes the article.

2. Background and related works

The HTTP protocol allows one to access a resource specified through an address termed Uniform Resource Locator (URL), which is a particular case of a Uniform Resource Identifier (URI). The generic pattern of a URI is http://host/alias/directory/file?request/fragment, where the request field may include a sequence of "name=value" pairs separated by the character' &'. For example http://ev.us.es/webapps/18-T-EC?action=frameset&subaction=view/end. The client establishes a TCP connection to send and receive HTTP messages to obtain a web resource. In the request message, the first line (request line) includes the method (e.g., GET, POST) followed by the desired resource. A sequence of header lines with the format name: value follows the request line. Some well-known headers are Host, User-Agent, If-Modified-Since, Connection, Transfer-Encoding or Cookie. A blank line will mark the end of the headers and the beginning of the body of the message. In the responses, the first line's syntax is modified (response line), and other headers such as Last-modified, Content-Type or Content-Length may be included. A secure connection uses the TLS protocol between HTTP and TCP, which forces the IDS to either act as a proxy of the connection or to use the server log.

Detecting attacks on web servers is a relevant problem that has been widely referenced in the literature Agarwal and Hussain (2018). Several attacks on web servers (e.g., XSS, Injection) use different elements of the HTTP request, including headers such as Cookie, Referer, or Response. However, most known attacks use the URI (exclusively or combined with other HTTP elements). In Indeed, more than 85% of the HTTP attack signatures used by some well-known WAF/IDS (e.g., Snort, ModSecurity) include the URI field.

Since signatures are intended to detect known attacks, IDSs Zuech et al. (2015) and WAFs Pałka and Zachara (2011) can be enhanced with anomaly detection techniques to extend their detection range to unknown or zero-day attacks. The primary method for detecting anomalies processes the input data and extracts a set of selected features Ahmed et al. (2016). Then, it applies a model of normality that assigns a score that gauges the deviation from the expected behavior of these features. The request is considered anomalous if the score exceeds a certain threshold. However, some techniques directly label the request as anomalous or not without using a score.

2.1. Related works

Several systematic reviews are devoted to intrusion detection based on network anomalies Blázquez-García et al. (2022); Chalapathy and Chawla (2019); Cook et al. (2020); Moustafa et al. (2019); Pang et al. (2022). It is also possible to find some papers providing a critical review on this topic García-Teodoro et al. (2009); Sommer and Paxson (2010). But in contrast to these works, we focus on the web and HTTP messages as input data rather than traffic and communications properties. There are also revision works related to the detection of web attacks, but differently from ours, they focus on specific types of attacks such as XSS Sarmah et al. (2018), SQL injection Nasereddin et al. (2021), web shell Jaafar et al. (2019) or HTTP DDoS Jaafar et al. (2019); Prajapati et al. (2019). Finally, other surveys have addressed the problem of web attacks but are restricted to a specific set of techniques, such as Alaoui and Nfaoui (2022), which only considers Deep Learning. We do not restrict our analysis to any particular technique.

Few surveys are devoted to detecting attacks on web servers based on HTTP messages from a neutral stance regarding the types of attacks and techniques considered. Thus, Babiker et al. (2018) has a limited scope that is not focused on anomaly detection. Only two survey papers are relevant for this work as they provide an excellent systematic review of the literature Sureda Riera et al. (2020) and Agarwal and Hussain (2018). Unlike them, we critically analyze the techniques, emphasizing their limitations and applicability. We also offer a taxonomy based on a framework that eases the categorization of the surveyed techniques. Only Agarwal and Hussain (2018) establishes a taxonomy of the systems under study based on a set of dimensions related to the detector capabilities (from the operative point of view). Their taxonomy lists the approaches, ignoring other aspects such as the technique class. The authors also identify some of the challenges we appoint from a different perspective and terminology.

3. Generic process for the detection of anomalies in HTTP messages

Figure 1 shows a sequence of operations commonly followed to detect anomalies in HTTP requests. The sensor in Figure 1 produces the observation of the request used as input for the detector system that will end up classifying it as normal or abnormal (i.e., attack). The processing steps that lead to this classification can be grouped into two categories or blocks: (a) preprocessing/feature generation and (b) detection/classification. The first block encompasses all operations aimed at producing a representation of the request suitable for the second block (detection). These categories

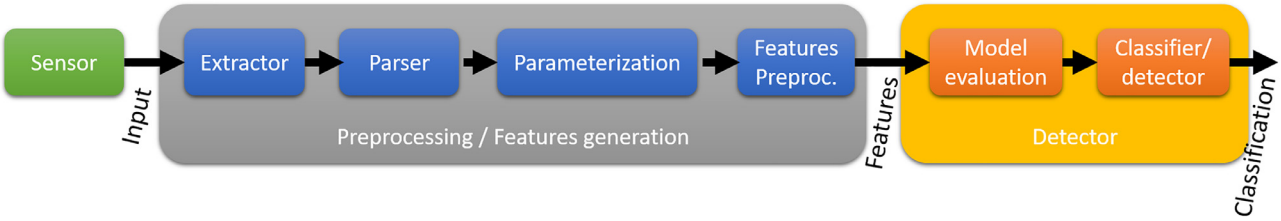

Fig. 1. Process for detecting anomalies in the URL.

will allow us to categorize different proposals according to the focus of their contribution.

3.1. Block: preprocessing / feature generation

The goal of this block is to obtain relevant features that are suitable for the detector block. It comprises the following operations:

- Extract. The first step is to extract the actual data of interest from the observation generated by the sensor. The URI will always be extracted in our case, but it could also include other elements, such as cookies or response codes.
- Parse. This step includes extracting data fields from the extractor's output after a syntactic and/or semantic analysis. For instance, breaking down the URI into its constituent fields (e.g., path, attribute, value), protocol version, etc. This step aims to generate an input vector with elements characterized according to an interest criterion.
- Parametrization. This essential operation obtains information from the parser's output by analyzing its properties for example, by assessing probabilities, identifying patterns, or n-grammars.
- Preprocessing of features. In some cases, parametrization results are too complex to be directly used in next steps due to excessive variability or dimensionality (e.g., n-grammar). This step reduces the complexity in such cases through techniques such as Principal Component Analysis (PCA), clustering, etc.

The output of this block is a representation (Features in Figure 1) of the request that contains sufficient information for the detection module to perform its job.

3.2. Block: decision module

This block aims to determine whether the observation is an attack or not (although it is possible to find works discriminate between three categories: normal, abnormal, and attack). We can differentiate two operations in this block.

- Model evaluation. Evaluate the input according to a model preestablished during training and obtain metrics related to similarity with respect to the model of normality.
- Classifier / detector. The final decision (or classification) is made based on the previous metrics.

Typically, these two operations are integrated. Nevertheless, it depends on the technique used and the existence of an explicit model. For example, the model is implicit in neural networks using deep learning, whereas a system that mixes Gaussians will use an explicit model.

4. Taxonomy of techniques used in the literature

Table 1 shows a taxonomy of the techniques found in the literature for the aforementioned operations.

4.1. Input data

The most common approach is to consider the payload in the HTTP request, but some works use other data sources. For example, complete HTTP sessions Ezeife et al. (2008); Fdez-Riverola et al. (2006), full packets An et al. (2021); Zhang et al. (2016), pairs of request / response Corona et al. (2009), the URI of requests Estévez-Tapiador et al. (2005), or log files Mimura (2020).

4.2. Preprocessing block

The input data can be characterized through various techniques that provide a similar range of representations. The foundational work by Virgna and Kruegel Kruegel and Vigna (2003); Kruegel et al. (2005) is considered a reference for the preprocessing stage and has given birth to a class of detectors commonly known as PAYL (short for payload). These detectors use statistical distributions and n-grammars to represent the fields that make up the URI. Alternatively, we can find some work exploring feature engineering Tufan et al. (2021) or directly applying parametrizations previously proposed in generic intrusion detection scenarios Gupta and Modak (2021).

- Extractor. We can find four basic approaches for this operation, each strongly related to the type of attack to be detected and the approach followed in the parametrization. These are: considering the raw data in the sample without filtering Kim et al. (2020), obtaining all fields from the sample Ariu et al. (2011); obtaining a subset of the fields in the input such as URI, method and reference Perdisci et al. (2009), or exclusively the URI Kruegel and Vigna (2003).
- Parser. There is more diversity in the techniques used for this operation. Although some works skip this step García Adeva and Pikatza Atxa (2007), the most common approach is to perform a syntax / semantic analysis to extract information from some fields and apply some procedure for normalization or segmentation to arrange the data. Some approaches parse the values from all fields of interest Ariu et al. (2011). Others segment the URI and extract its constituent elements Estévez-Tapiador et al. (2005). Some others generate attribute-value pairs (AVP) Corona et al. (2009), while others replace strings or characters with prearranged values Mac et al. (2018) or locate and mark specific keywords Ezeife et al. (2008). These approaches can be combined, such as in Yu et al. (2018) where the keyword values found in the query part of the URI, such as UNION, SELECT, etc., are substituted by predefined markers.
- Parametrization. This step applies a wide range of techniques to characterize the data representation. Some approaches use n-grammar (central in PAYL systems) Perdisci et al. (2009), other approaches use statistical measurements Zhang et al. (2016), bag-of-words (BoW) Ren et al. (2018), pattern analysis Yu et al. (2018), dictionaries Tang et al. (2020), or probabilistic models Estévez-Tapiador et al. (2005). To a lesser extent,

Table 1
Common techniques used in each step.

Input

Request	[26]
Session	[27, 28]
Request/response	[29]
Full packet	[30, 31]
in which Logs	[32]

⇓

Preprocessing / Features generation

Extractor		Parser		Parametrization		Feature preprocessing	
Full data	[33]	Keywords	[27]	Statistical	[30, 35]	PCA	[42]
All fields	[34]	Field values	[34]	Bag of words	[39]	Clustering	[7]
Selected fields	[35]	Field-value pairs	[29]	n-grams	[35]	Filters	[32]
URI	[36]	Uri values	[26]	Probabilities	[26]	Patterns	[43]
		Search & replace	[37]	Patterns	[40]	Composition	[44]
		None	[38]	Dictionaries	[41]	Feature selection	[45]
						None	[26]

⇓

Preprocessing output

Vector (numerical)	[46, 42]
Vector (hybrid)	[47]
Sequence	[26, 48]
Matrix/image	[49]

⇓

Detector

Model		Classifier/Detector	
		SVM	[44]
Markov model	[26]	Deep learning (generic)	[43, 53]
PDF	[7]	CNN & LSTM	[54, 33]
Representatives	[50]	Threshold	[26]
Statistical	[47]	kNN	[38]
Implicit	[51]	Ensemble	[48]
Others	[52]	Isolation & Random Forest	[32]
		Metrics	[42, 52]

we can also find other techniques. In this block, it is not uncommon to combine various techniques to obtain a vector of characteristics, such as in Perdisci et al. (2009), where the n-grammar and statistics measurements are combined. In the case of n-grammar, the elements at the input are represented after segmentation in blocks of size n. Then, a vocabulary of blocks of n consecutive characters is learned. Another common approach is to use a vector of statistics inferred from the URI, such as size, field length, and number of special characters. In the bag-of-words technique, a vocabulary is established. Then, each observation is added to a vector of presence that includes each word observed from the vocabulary. Pattern analysis techniques include the presence or absence of specific patterns in the parametrization. Techniques based on dictionaries set possible values for each element or parameter. Finally, techniques based on probabilistic models set a probability for each event or observation, such as finding a specific keyword in the path field of a URI. An interesting property of parametrization is adding information about sorting to different fields and values. Nevertheless, in some works, this sorting is overlooked Li et al. (2020) while in others, the sorting is done locally considering subsequences such as in n-grammars Perdisci et al. (2009), or it is explicitly Criscione et al. (2010) or implicitly Song et al. (2009) inserted.

- Preprocessing of features. Some works apply further techniques for processing the vector of characteristics that results from parametrization. These techniques aim to reduce the dimensionality of such a vector or the number of samples after discretizing values. For instance, n-grammars and BoW can provide a vector with high dimensionality (one dimension per each combination/word in the vocabulary, which grows exponentially in the case of n-grammar) and is of 'sparse' type. This is also the case in the blind parametrization that occurs when generic characteristics (i.e., not related to HTTP) include numerous components that are not always related to the message payload. The techniques applied in this operation are generally analysis of principal components (PCA) Kakavand et al. (2019), clustering Betarte et al. (2019), filtering/smoothing/quantization of values Mimura (2020), and pattern analysis Liu et al. (2020). It is also possible to find other approaches using a composition of various techniques Vartouni et al. (2019) or introducing feature selection analysis during the tuning of the system Gupta and Modak (2021).

As a result of the preprocessing stage, we obtain a characterization of the request in the form of a vector of characteristics Li et al. (2018), which is typically numerical Kakavand et al. (2019), sometimes combined with categories Criscione et al. (2010). In

other proposals, HTTP requests are represented as sequences of numerical values or tags of elements of a vocabulary Estévez-Tapiador et al. (2005). Finally, a recent proposal transforms the request into a matrix that can be considered as an image of the request Tekerek (2021). The latter technique is typically related to the use of deep learning techniques in the detection stage.

4.3. Detection block

As aforementioned, although we break down this block in the model and classifier, this division does not exist in some approaches, such as deep learning.

- Model. We can group the proposals according to the following classes: Markov models Estévez-Tapiador et al. (2005) –which includes n-grammar–, probability distributions that generally take the form of mixing Gaussians Betarte et al. (2019); set of constituents Torrano-Gimenez et al. (2009) which are used as examples for classification; statistical models Criscione et al. (2010), and implicit model Hao et al. (2019). In some singular cases, there is no explicit model of the request, as is the case of metrics with detection thresholds Maestre Vidal et al. (2020).
- Classification. The techniques for classification are diverse and cover a wide range within anomaly detection (e.g., statistical, clustering, deep learning, knowledge-based, etc.). There is a trend in using AI-based techniques these days, but a significant number of techniques are not AI-based, such as the seminal work by Kruegel Kruegel et al. (2005), based on statistical models. Clearly, the technique used is strongly determined by the model and the representation of the request obtained after the first stage. Thus, we can find from simple techniques such as threshold detectors to more complex Deep Learning systems. The former is commonly used in the context of Markov-based models Estévez-Tapiador et al. (2005), probability distributions Kakavand et al. (2019), or metrics Maestre Vidal et al. (2020). In the case of neural networks, we can find generic ones such as MLP Liu et al. (2020) and SOM Zolotukhin et al. (2012), or recurrent networks that include analysis of CNN Park et al. (2018) and LSTM sequences Kim et al. (2020). It is also common to use Support Vector Machines (SVM), and variants of it Vartouni et al. (2019), classifiers such as kNN García Adeva and Pikatza Atxa (2007) or random forest RF Mimura (2020). There is abundant literature in which the authors combine various techniques and detectors to create ensemble systems Geraily and Jahan (2012).

Although not all combinations of the techniques shown for each step are possible, there is an abundant number of combinations described in the literature. In this regard, we can find two main groups of approaches for intrusion detection. Thus, the first group focuses on the preprocessing of the HTTP request message with the undeclared target of improving the significance of the features or representations obtained for the identification of the attacks, i.e., they try to extract potentially discriminative characteristics from the analysis of the URIs. This is the case, for example, of the PAYL family Perdisci et al. (2009) or others based on identifying the elements of the URIs Duessel et al. (2017). The proposals in this group are numerous and include more variants regarding the preprocessing block (Fig. 1), while the detector block is chosen according to the obtained features. On the other hand, the second group of proposals directly focuses on the classification itself, that is, select a classification method and try to apply it to the problem. Thus, the classification method determines the applicability of the techniques for each step in the preprocessing block. In this case, the direct use of a set of predefined representations is widespread Wu et al. (2018). It is particularly striking that some authors somehow 'force' a transformation of data to enable the use of a different technique that would not be applicable otherwise. For example, some proposals based on CNN or LSTM transform data into a 'picture' (matrix) so that its application is possible Park et al. (2018). Indeed, the most recent trend in the literature is forged in comparing various ML techniques or the inclusion of a variant from a pre-established representation (primarily due to the adaptation to a preexisting dataset). Therefore, the research trend now seems to be toward the detection module rather than a new representation of requests.

5. Limitations found in the literature and open challenges

Determining the normality of an HTTP request is extremely challenging in the present context of the dynamism of web services and strong dependence on the specific service. Effective and efficient systems must overcome some limitations of theoretical and practical nature that, in our opinion, persist today. Some of them have already been noted in previous work Sommer and Paxson (2010)García-Teodoro et al. (2009) and are common to the research field of anomaly-based intrusion detection.

5.1. Overlook of the syntax of the URL

A peculiar characteristic of web services is the existence of a semantic and syntactic structure in the payload, particularly in the URI. This is crucial in the specificity of the various intrusion detection approaches to HTTP traffic. However, several works overlook this fact and base their detection technique on the extraction of features from the entire HTTP payload Park et al. (2018), often ignoring the position and semantics of the string/words in the request Kim et al. (2020) or characterizing the request line as a whole. This is the case in some proposals that belong to the PAYL family, where, despite an initial segmentation of the URI into fields, a sliding window of size n (n-grammar) is applied over the entire URI Song et al. (2009), or selected fields Zolotukhin et al. (2012). The underlying idea is that attacks include specific characters and short sequences of characters inserted into legitimate requests. However, this can be best identified using signatures instead of anomaly detection.

Challenge 1. *Research proposals should explore approaches that consider the syntax and semantics of the HTTP payloads (e.g., resource location or attribute and values associated) during characterization to establish some form of vocabulary. Given that the vocabulary size is likely to be unbounded, proposals should be ready for that too.*

5.2. Lack of scalability due to large variability & vocabulary

Most current web services use databases that handle large volumes of data in many different application scenarios. This context usually involves high variability and complexity in the data included in the HTTP requests, making it difficult to detect anomalies. This variability has two different dimensions: inter-service and site-internal. The former is related to the service's specificity, as there is a strong dependence on the content of a URI with the website being considered. Significant differences are expected among sites due not only to the different nature of the application (e.g., a web-based newspaper vs. a library site) but also to the site itself (e.g., science library vs. arts library). The second dimension is related to the dynamic nature of the service provided, which is associated with large datasets with heterogeneous content.

Inter-service variability is generally approached through the analysis of individual websites. This induces site-specific models, which poses the problem of the applicability of one model in different contexts/scenarios. We will address this later in Section 5.5.

The internal variability of the site appears in the form of the different strings/values that may appear in each component of the URI. This is particularly relevant for the query part, as it includes values from the database. This problem conditions the existing approaches differently and even forces the adoption of some variants to minimize the issues derived.

Therefore, approaches based on dictionaries such as Tang et al. (2020) or based on dictionary statistical information Estévez-Tapiador et al. (2005) will likely face scalability problems due to the volume and variability of words observed during training. Furthermore, some words not observed during training may appear during the test phase, compromising the model's precision and applicability. Therefore, these approaches are likely to suffer from the problem of *insufficient training* or the issue of *out-of-vocabulary*Estévez-Tapiador et al. (2005), since it is not possible to guarantee that all possible words/parameters have been observed during training. This issue translates into observing words/parameters out of the learned vocabulary that will likely be considered abnormal / attack. As a result, this problem poses a serious limit to the applicability of dictionary-based approaches.

A common alternative to overcome this limitation is to use n-grammars, as done by the PAYL family Kruegel et al. (2005). In this case, fixed-length substrings are considered the basis for the analysis. Thus, the size of the vocabulary, that is, combinations of n characters for an n-order grammar, for training is upper bounded (e.g., 256 characters for 1-grammars or 256^2 for 2-grammars if extended ASCII is considered for codification). However, if n is small, the representativity of the grammar compared to dictionaries is poor. In contrast, if n is large, the size of the vocabulary to be watched grows exponentially with n, which may result in scalability problems. Furthermore, the problem of *insufficient training* can still be present, as there is no guarantee about observing all possible combinations. At the same time, we will need to estimate the probabilities associated with a large number of possible combinations of characters, thus challenging the estimation and representation.

Other authors propose to limit the size of the vocabulary by only considering some tokens or keywords Kozi et al. (2015), only the most frequent/relevant words Mimura (2020), or by including some form of pattern analysis Liang et al. (2017). But this also limits the detection capabilities to attacks directly related to those patterns or keywords (e.g., SQL injection or buffer overflow attacks).

Finally, one of the most simple solutions adopted for these problems is to avoid using any vocabulary in the representation of the URIs or other parts of the HTTP requests. This is done by extracting features from the entire URI, as done in Park et al. (2018), or by considering the entire URIs or HTTP request as inputs to the classifier, as done in some kNN-based approaches Kirchner (2010). However, again, these approaches limit the detection capabilities. Ignoring the vocabulary in the analysis is prone to hide some of the relevant properties for the identification of an attack (e.g., the appearance of '/etc/passwd' in a data leakage attack), while considering the entire URIs as samples do even scale worse than using dictionaries and generates problems of representativity.

Challenge 2. *An open challenge is to find new approaches that can cope with large vocabularies in a scalable manner so that new words are not necessarily directly translated into anomalies. These approaches also need to generalize learning outcomes and be validated with large datasets with high variability in the vocabulary.*

5.3. Obsolescence of the model learned

The dynamic nature of web services causes the information in the HTTP request to change over time due to changes in the service (i.e., path) or changes in the content and data associated. This data shift problem Moreno-Torres et al. (2012) causes that the model learned during a temporal window becomes inadequate or, at least, less representative after a while. This problem has been addressed only marginally in some works Maggi et al. (2009), and, in some cases, the only solution proposed is to start the training phase over. With few exceptions Estepa et al. (2020), none of the works that address data shifting offer schemes for identifying the right time to re-train the model, suggesting simply the notion of periodic training. On the other hand, some works try to adapt the model rather than re-train it through techniques that interpolate the previous and current models Maggi et al. (2009), or generic methods for incremental learning.

Challenge 3. *Ideally, research proposals should show or analyze performance validity with time and suggest mechanisms for re-training and adaptation to temporal changes. This is not an easy undertaking as it first requires one to count with a large dataset (i.e., an extended period) that includes variations over time. The obtention and labeling of such a dataset would be costly. However, this raises problems related to the adversarial environment Maestre Vidal et al. (2020) since attackers will likely try to modify their requests to avoid being detected. Indeed, they could even try to adapt the models to their best convenience in the case of re-training.*

5.4. Availability of adequate datasets

A significant limitation found in the literature is related to the datasets used for training and evaluation in terms of data representativity (concerning the modeled service and attacks) Ring et al. (2019) and the ability to generalize the performance of the technique Sommer and Paxson (2010). Most of the problems –outlined in some works García-Teodoro et al. (2009) Tavallaee et al. (2010)– are common to the IDS field.

The most relevant issue is related to the size of the dataset. The volume of data used in the literature tends to be modest (most datasets include less than 100k requests) for the number of features the detection relies upon and the actual variability of the URL. In Table 2 we show some properties –including its size– of some public datasets found in the literature that contain labeled traffic that includes the URI and that are widely used for experimentation in web intrusion detection. We also show the dataset's type, duration, and the number of usable URIs. It can be observed that only two datasets incorporate traffic from web services and that some of them do not include practical URIs even though they are used in some works. It is important to note that only incoming HTTP traffic is of interest for modelling web servers. Another relevant characteristic is the low number of URIs, which is undoubtedly insufficient for tuning and testing most techniques and lacks representativity for highly dynamic services. This scarceness seems to explain why most authors rely on in-home datasets.

The main issues with current datasets can be summarized as follows:

- Scarcity of public datasets suitable for experimentation in modern real-life systems that are large enough, up-to-date and use real-life traffic (i.e., that exclude simulated scenarios such as Riera et al. (2020)) Kenyon et al. (2020). Surprisingly, a significant portion of the datasets used in the literature, even today, include DARPA'99 or its derivatives (KDD'99 and NSL-KDD). Some works are not suitable for attack detection due not only to design flaws or the use of simulated data sets Mchugh (2000) but also because the web services and attacks today differ from those in 1999. The lack of sufficient volume in the datasets is particularly important for approaches based on neural networks, where the number of degrees of freedom (i.e., fit parameters) can be even greater than the number of

Table 2
Public datasets used in the literature for web intrusion detection systems assessment.

Dataset	Format (URIs)	Classes	Real/Synth	Labels	Only HTTP	Year	Duration	useful URIS	Comments
DARPA'99	PCAP	N/A/U	S	G	N	1999	5w	100k	Widely used for IDS research, flawed, obsolete, not http specific traffic/attacks, only 2 web servers
KDD'99	CSV	N/A/U	S	G	N	1999	5w	-	Parameterized flows, directly derived from DARPA'99, inherit DARPA'99 flaws
NSL-KDD	PCAP	N/A/U	S	GS	N	2009	5w	10k	Sanitized version of KDD'99, not for URI based analysis
CSIC2010	LIST	N/An	S	G	Y	2010	-	96k	Only URI, mixes attacks with anomalies, single server
UNSW-NB15	PCAP	N/A	RS	G	N	2015	31h	27k	Lack of details about web servers and traffic
ISCX-URL2016	CSV	N/A	S	G	Y	2016	-	-	Outgoing traffic, 114k URIs, not suitable to train a site, highly umbalanced (only 35k normal URIs)
CIC-IDS2017	PCAP	N/A	S	G	N	2017	5d	272k	392k URIs, mix of outgoing and incoming traffic, most incoming URIS are attacks, low variability in URIs (only 6k queries and many repeated URIs)

* Classes: N (normal), A (attacks), U (unlabeled), An (anomaly). ** Labels: G (As generated), S (Supervised).

available samples used in some works. This fact can produce overfitting, so the model will faithfully reproduce the training datasets but lacks applicability in an actual deployment.

- Lack of specificity. Many of the most widely used datasets are not specific to the web and include the full traffic capture (e.g., DARPA'98) and/or flow-based parameters (e.g., KDD'99). This fact aggravates the issue of insufficient dataset size, since only a fraction of the data is related to HTTP. Furthermore, flow-based analysis restrains the content of URIs, rendering these datasets unusable for approaches based on URI analysis. Only a couple of datasets are targeted at detecting web attacks (CSIC2010 and ISCX-URL2016). On the other hand, not all web traffic in a dataset is of interest. As the assessment and modeling are performed on a per-server basis, only incoming requests to the monitored sites are usable. Again, this reduces the actual number of usable URIs. In this regard, among the data sets in Table 2, only CSIC2010 contains only incoming requests to a website.
- The attacks included in the datasets are scarce, outdated, and are mainly simulated. Given that the number of attacks in real life is quite unbalanced with respect to legitimate requests, labeling attacks from a real-world trace would require one to collect vast volumes of traffic so that attacks are representative. On the other hand, it would be extremely costly to label a large volume of traffic manually. For this reason, some works generate attack datasets using several sources – e.g. public OSINT repositories, vulnerability scanners, exploits, etc. García et al. (2006)–, aiming to cover a wide range of attack types Erlacher and Dressler (2018) or, on the contrary, restricting the type of attacks to specific ones (e.g., sql-injection Akrout et al. (2014)). However, the *manual* inclusion of attacks in the dataset generates a bias regarding the balance between normal and attack traffic that can obfuscate the metrics used to assess the detectors. Recently, some authors have been using datasets to detect anomalies (that is, not necessarily attacks), such as in CSIC2010 Nguyen et al. (2011), which includes simulated data that, apart from attacks, contain values in the parameters that do not respect the semantics of the parameters. However, these outlier data do not represent attacks but simply incorrect data. Therefore, the works that use this data set (e.g. Gupta and Modak (2021)) cannot assess the attack detection capacity unless a different attack dataset is used.
- Assessing the detectors requires a *ground truth*, i.e., a labeled dataset. Except for a few proposals, the training of the normality model requires attack-free traffic. But, in any case, evaluating the detector's capabilities requires both normal and attack traffic to be correctly classified. Synthetic datasets (Table 2) include labels during the automatic generation procedure (automatic). But the need to label traffic from a real scenario creates a recursive problem, as it is impossible to detect all attacks automatically Díaz-Verdejo et al. (2020), and, at the same time, it is not plausible to do it manually unless the size of the dataset is small, which would affect its ability to train models.

Indirectly related to the datasets, there is a crucial problem of comparability of different proposals. We cannot compare the results obtained with different datasets. Furthermore, the limitations above make the results obtained in most works of low relevance for modern systems. In fact, it is common for authors to use in-house datasets Tang et al. (2020), which prevents direct comparison with other works unless the same techniques are implemented. Sometimes, even though using the same techniques, the comparison is often unfair or inadequate if the authors fail to provide enough information about the settings. This impairs the advance of the research.

Challenge 4. *It is necessary to count with labeled datasets that are representative and large enough to train and support the temporal evolution of the service, which is a major challenge. On the one hand, having new datasets from real traffic would be desirable. But this encompasses a costly process of anonymization and labeling for large volumes of data, leading to the use of sanitization with semi-supervised methodologies Díaz-Verdejo et al. (2020). On the other hand, attacks from a real-life capture are expected to be less than desirable and will probably not include the required attack variability. Thus, it is necessary to develop suitable approaches to generate artificial attacks adapted to real scenarios that complement or substitute existing attacks while considering potential biases in the balance among classes. Repositories of vulnerabilities and existing attacks should be used to cover most types of known attacks.*

5.5. Methodology or inference limitations

Another subject of concern is the use of flawed methodologies to evaluate the systems. In this sense, we can find several limitations:

- Misdirected training/testing/validation. Numerous works apply an incorrect experimental methodology to assess the performance of their proposals concerning to the data used for each step. In fact, some authors even use the same dataset for training and evaluation Yu et al. (2018). This deficiency prevents one

from obtaining accurate information on the detection capability versus actual data. It is also common to use two disjoint partitions for training and evaluation, including techniques for cross-validation to improve the significance of results Mimura (2020). Unfortunately, this is only adequate when different approaches are not compared and model parameters are not fitted after testing, seeking a better result for the technique, as it would produce overfitting for the evaluation dataset. Few works use a third disjoint partition to validate the results. We believe this could be due to the scarcity of databases of adequate size.

- Blind reue of the technique. Some works improve previous proposals by suggesting a modification or variant of an existing scheme. This, although initially seems adequate, ends up, in many cases, in overfitting. First, due to the blind use of the classification technique without a previous study on its adequacy Gupta and Modak (2021), that is, not all methods are suitable to separate normal/attack classes from the parametrization used. Second, these variations tend to use the same data sets as the original work in the hope of an incremental improvement in some parameters Liu et al. (2020).
- Use of insufficient performance metrics. The problem of detecting attacks is by nature a strongly unbalanced classification problem (e.g., our experience suggests that attacks tend to be less than 0.02% of requests). Therefore, traditional indicators such as precision, recall, or the f1 score should not be considered the only ones, as many proposals do Wang et al. (2019). Specifically when the point of operation of the classifier cannot be set, such is the case of neural networks, kNN, or SVM. It is particularly important that metrics outweigh achieving a low rate of false positives so that the system's usability is plausible. This implies analyzing the trade-off between true and false positives by adjusting some meta-parameters, as commonly done when using Receiver Operating Characteristics curves, ROCMimura (2020).
- Biased evaluation. Some proposals address the unbalanced nature of the problem by changing the ratio of normal and attack samples in the test phase. Usually, a 50/50 proportion is used de la Torre-Abaitua et al. (2021) in these approaches by acquiring a similar number of samples, if an in-home dataset is used, or by dismissing some normal data, if external datasets are used. Anyway, preserving the natural proportions of attacks is almost unfeasible for synthetic datasets and even for real datasets due to attack scarcity and lack of representativeness (see Section 5.4).
- The classifier does not provide additional information. Most systems use a binary classification (i.e., normal or attack), and, in the best case, a third category is used for doubtful bases. This is the case for the most commonly used classifiers (neural networks Park et al. (2018), SVM Wang et al. (2019), etc.). However, the interpretation of alarms by the CSO and the integration of alarms in SIEMs would benefit from additional information besides a classification between normal / attack / anomaly. This additional information could include not only the quality of the classification but also the reason why the request has been marked as an attack Duessel et al. (2017). Unfortunately, few proposals include additional information that allows the CSO to know the degree of deviation from normality Salazar-Hernández and Díaz-Verdejo (2010) and consequently facilitate the adjustment of the optimal operation point or a confidence interval in the result Tang et al. (2020). Some systems even classify the type of attack, although they have limited capabilities.
- Classification subject. The detection of web attacks, at least those based on individual requests, should consider in some way the offensive parts of the requests and, consequently classify individual requests. But some proposals classify the request based on parameters at the flow level or even consider all the requests from an offending IP as attack An et al. (2021). In this regard, some of the existing public datasets directly provide flow-based data as input (KDD99) or are labeled per flow (e.g., IDS-ICICS2017).
- Use of a specific server/service that prevents generalization. Due to the lack of suitable datasets, most works assess only the proposed system over a single dataset/server. Thus, it is not possible to evaluate their generalizability.

Challenge 5. *Overcoming some of the previous limitations, besides applying good practices, strongly depends on the availability of quality datasets that allow: verification of the capability of generalization of the proposals, the use of a third partition that is only used for the final assessment of the system, and an unbiased assessment. The evaluation, and, mostly, the ability to compare different proposals, requires using metrics that gauge the balance between the detection capability and the false positive rate in strongly unbalanced environments. The interpretability of the classification, which is an essential requirement for the actual usability of the detectors, requires novel approaches capable of enriching the classification by identifying the cause of the threat or categorizing them.*

6. Conclusions

We have defined a framework with common steps for detecting anomalies in HTTP messages. This framework articulates a taxonomy of techniques and facilitates comparison between different approaches.

The literature on anomaly detection in the context of detecting HTTP-based web attacks suffers from some limitations that partially impair the advance of this research field. Most research proposals are presented as isolated contributions (i.e., new approaches are designed for niches without comparing with previous research) and are biased to datasets from specific scenarios. This makes it difficult to compare different solutions. Furthermore, it remains unclear to what extent advances in detecting cyber-attacks based on HTTP request elements can or cannot be generalized and transferred from isolated niches to a broader scale endorsed by the industry.

Real-life deployments that include web services are likely to exhibit some issues (such as scalability or obsolescence) that have not been satisfactorily covered in the existing literature and require further analysis and solutions from this research field.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

This work has been partly funded by the research grant PID2020-115199RB-I00 provided by the Spanish ministry of Industry under the contract MICIN/AEI/10.13039/501100011033, and also by FEDER/Junta de Andalucía-Consejería de Transformacin Econmica, Industria, Conocimiento y Universidades under projects PYC20-RE-087-USE and A-TIC-224-UGR20.

References

Watchguard, Watchguard launches 2016 q4 internet security report. https://www.northamber.com/sites/default/files/marketing/solutionsSite/PDFs/WatchGuard,Accessed: 2021-11-27; 2016.

news, December 2021 web server survey. https://news.netcraft.com/archives/2021/12/22/december-2021-web-server-survey.html,Accessed: 2010-09-30; 2021a.

securitybrief, December 2021 web server survey. https://securitybrief.asia/story/malicious-web-application-attacks-climb-88-report,Accessed: 2010-09-30; 2021b.
Agarwal, N., Hussain, S.Z., 2018. A closer look at intrusion detection system for web applications. Secur. Commun. Netw. 2018, 1–27. doi:10.1155/2018/9601357.
Ahmed, M., Naser Mahmood, A., Hu, J., 2016. A survey of network anomaly detection techniques. J. Netw. Comput. Appl. 60, 19–31. doi:10.1016/j.jnca.2015.11.016.
Akrout, R., Alata, E., Kaaniche, M., Nicomette, V., 2014. An automated black box approach for web vulnerability identification and attack scenario generation. J. Brazil. Soc. 20 (1), 1–16.
Alaoui, R.L., Nfaoui, E.H., 2022. Deep learning for vulnerability and attack detection on web applications: a systematic literature review. Future Internet 14 (4), 118. doi:10.3390/fi14040118.
An, Y., Yu, F.R., Li, J., Chen, J., Leung, V.C.M., et al., 2021. Edge Intelligence (EI)-Enabled HTTP anomaly detection framework for the internet of things (IoT). IEEE IoT J. 8 (5), 3554–3566. doi:10.1109/JIOT.2020.3024645.
Ariu, D., Tronci, R., Giacinto, G., 2011. HMMPayl: an intrusion detection system based on hidden markov models. Comput. Secur. 30 (4), 221–241.
Babiker, M., Karaarslan, E., Hoscan, Y., 2018. Web application attack detection and forensics: a survey. In: 2018 6th International Symposium on Digital Forensic and Security (ISDFS). IEEE, pp. 1–6. doi:10.1109/ISDFS.2018.8355378.
Betarte, G., Gimenez, E., Martinez, R., Pardo, A., 2019. Improving web application firewalls through anomaly detection. In: Proceedings - 17th IEEE International Conference on Machine Learning and Applications, ICMLA 2018. Institute of Electrical and Electronics Engineers Inc., pp. 779–784.
Blázquez-García, A., Conde, A., Mori, U., Lozano, J.A., 2022. A review on outlier/anomaly detection in time series data. ACM Comput. Surv. 54 (3), 1–33. doi:10.1145/3444690.
Chalapathy, R., Chawla, S., 2019. Deep learning for anomaly detection:. A Survey.
Cook, A.A., Misirli, G., Fan, Z., 2020. Anomaly detection for iot time-series data: a survey. IEEE IoT J. 7 (7), 6481–6494. doi:10.1109/JIOT.2019.2958185.
Corona, I., Ariu, D., Giacinto, G., 2009. HMM-web: a framework for the detection of attacks against web applications. In: IEEE International Conference on Communications, pp. 1–6.
Criscione, C., Salvaneschi, G., Maggi, F., Zanero, S., 2010. Integrated detection of attacks against browsers, web applications and databases. EC2ND 2009 - Eur. Conf. Comput. Netw. Def. 37–45.
Díaz-Verdejo, J.E., Estepa, A., Estepa, R., Madinabeitia, G., Muñoz Calle, F.J., 2020. A methodology for conducting efficient sanitization of HTTP training datasets. Future Generat. Comput. Syst. 109, 67–82.
Duessel, P., Gehl, C., Flegel, U., Dietrich, S., Meier, M., 2017. Detecting zero-day attacks using context-aware anomaly detection at the application-layer. Int. J. Inf. Secur. 16 (5), 475–490.
Erlacher, F., Dressler, F., 2018. How to test an IDS? GENESIDS: an automated system for generating attack traffic. WTMC 2018 - Proceedings of the 2018 Workshop on Traffic Measurements for Cybersecurity, Part of SIGCOMM 2018 46–51.
Estepa, R., Diaz-Verdejo, J.E., Estepa, A., Madinabeitia, G., 2020. How much training data is enough? A case study for HTTP anomaly-based intrusion detection. IEEE Access 8, 44410–44425.
Estévez-Tapiador, J.M., García-Teodoro, P., Díaz-Verdejo, J.E., 2005. Detection of web-based attacks through Markovian protocol parsing. In: Proceedings - IEEE Symposium on Computers and Communications, pp. 457–462.
Ezeife, C.I., Dong, J., Aggarwal, A.K., 2008. SensorWebIDS: a web mining intrusion detection system. In: International Journal of Web Information Systems, volume 4, pp. 97–120.
Fdez-Riverola, F., Borrajo, L., Laza, R., Rodríguez, F.J., Martínez, D., 2006. HTTPHunting: an IBR approach to filtering dangerous HTTP traffic. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), volume 4065 LNAI, pp. 91–105.
García, V.H., Monroy, R., Quintana, M., 2006. Web attack detection using ID3. IFIP Adv. Inf. Commun. Technol. 218, 323–332.
García Adeva, J.J., Pikatza Atxa, J.M., 2007. Intrusion detection in web applications using text mining. Eng. Appl. Artif. Intell. 20 (4), 555–566.
García-Teodoro, P., Díaz-Verdejo, J., Maciá-Fernández, G., Vázquez, E., 2009. Anomaly-based network intrusion detection: techniques, systems and challenges. Comput. Secur. 28 (1-2), 18–28.
Geraily, M., Jahan, M.V., 2012. Fuzzy detection of malicious attacks on web applications based on hidden Markov model ensemble. In: Proceedings - 3rd International Conference on Intelligent Systems Modelling and Simulation, ISMS 2012, pp. 102–108.
Gupta, A., Modak, A., 2021. Anomaly detection in HTTP requests using machine learning. In: Advances in Intelligent Systems and Computing, volume 1311 AISC, pp. 445–455.
Hajj, S., El Sibai, R., Bou Abdo, J., Demerjian, J., Makhoul, A., Guyeux, C., 2021. Anomaly-based intrusion detection systems: the requirements, methods, measurements, and datasets. Trans. Emerg. Telecommun. Technol. 32 (4), e4240.
Hao, S., Long, J., Yang, Y., 2019. BL-IDS: detecting web attacks using Bi-LSTM model based on deep learning. In: Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST, volume 284. Springer Verlag, pp. 551–563.
Husák, M., Apruzzese, G., Yang, S.J., Werner, G., 2021. Towards an efficient detection of pivoting activity. In: 2021 IFIP/IEEE International Symposium on Integrated Network Management (IM). IEEE, pp. 980–985.
Jaafar, G.A., Abdullah, S.M., Ismail, S., 2019. Review of recent detection methods for HTTP DDoS attack. J. Comput. Netw. Commun. 2019, 1–10. doi:10.1155/2019/1283472.
Kakavand, M., Mustapha, A., Tan, Z., Yazdani, S.F., Arulsamy, L., 2019. O-ADPI: online adaptive deep-packet inspector using mahalanobis distance map for web service attacks classification. IEEE Access 7, 167141–167156.
Kenyon, A., Deka, L., Elizondo, D., 2020. Are public intrusion datasets fit for purpose characterising the state of the art in intrusion event datasets. Comput. Secur. 99, 102022.
Khraisat, A., Gondal, I., Vamplew, P., Kamruzzaman, J., 2019. Survey of intrusion detection systems: techniques, datasets and challenges. Cybersecurity 2 (1), 1–22.
Kim, A., Park, M., Lee, D.H., 2020. AI-IDS: application of deep learning to real-time web intrusion detection. IEEE Access 8, 70245–70261.
Kirchner, M., 2010. A framework for detecting anomalies in HTTP traffic using instance-based learning and k-nearest neighbor classification. In: 2010 2nd International Workshop on Security and Communication Networks, IWSCN 2010, pp. 1–8.
Kozi, R., Choraś, M., Renk, R., Hołubowicz, W., 2015. Patterns extraction method for anomaly detection in HTTP traffic. In: Advances in Intelligent Systems and Computing, volume 369, pp. 227–236.
Kruegel, C., Vigna, G., 2003. Anomaly detection of Web-based attacks. In: Proceedings of the ACM Conference on Computer and Communications Security. Association for Computing Machinery, pp. 251–261.
Kruegel, C., Vigna, G., Robertson, W., 2005. A multi-model approach to the detection of web-based attacks. Comput. Netw. 48 (5), 717–738.
Li, J., Zhang, H., Wei, Z., 2020. The weighted Word2vec paragraph vectors for anomaly detection over HTTP traffic. IEEE Access 8, 141787–141798.
Li, K., Chen, R., Gu, L., Liu, C., Yin, J., 2018. A method based on statistical characteristics for detection malware requests in network traffic. In: Proceedings - 2018 IEEE 3rd International Conference on Data Science in Cyberspace, DSC 2018. Institute of Electrical and Electronics Engineers Inc., pp. 527–532.
Liang, J., Zhao, W., Ye, W., 2017. Anomaly-based web attack detection: a deep learning approach. In: Proceedings of the 2017 VI International Conference on Network, Communication and Computing - ICNCC 2017. ACM Press, pp. 80–85.
Liu, C., Yang, J., Wu, J., 2020. Web intrusion detection system combined with feature analysis and SVM optimization. Eurasip J. Wirel. Commun. Netw. 2020 (1), 1–9.
Mac, H., Truong, D., Nguyen, L., Nguyen, H., Tran, H.A., Tran, D., 2018. Detecting attacks on web applications using autoencoder. In: ACM International Conference Proceeding Series. Association for Computing Machinery, pp. 416–421.
Maestre Vidal, J., Sotelo Monge, M.A., Monterrubio, S.M.M., 2020. EsPADA: enhanced payload analyzer for malware detection robust against adversarial threats. Future Generat. Comput. Syst. 104, 159–173.
Maggi, F., Robertson, W., Kruegel, C., Vigna, G., 2009. Protecting a moving target: addressing web application concept drift. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), volume 5758 LNCS, pp. 21–40.
Mchugh, J., 2000. 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. 3 (4), 262–294.
Mimura, M., 2020. Adjusting lexical features of actual proxy logs for intrusion detection. J. Inf. Secur. Appl. 50, 102408.
Moreno-Torres, J.G., Raeder, T., Alaiz-Rodríguez, R., Chawla, N.V., Herrera, F., 2012. A unifying view on dataset shift in classification. Pattern Recognit. 45 (1), 521–530.
Moustafa, N., Hu, J., Slay, J., 2019. A holistic review of network anomaly detection systems: a comprehensive survey. J. Netw. Comput. Appl. 128, 33–55. doi:10.1016/j.jnca.2018.12.006.
Nasereddin, M., ALKhamaiseh, A., Qasaimeh, M., Al-Qassas, R., 2021. A systematic review of detection and prevention techniques of SQL injection attacks. Inf. Secur. J. 1–14. doi:10.1080/19393555.2021.1995537.
Nguyen, H.T., Torrano-Gimenez, C., Alvarez, G., Petrović, S., Franke, K., 2011. Application of the generic feature selection measure in detection of web attacks. In: Lecture Notes in Computer Science, volume 6694 LNCS, pp. 25–32.
Pałka, D., Zachara, M., 2011. Learning web application firewall - benefits and caveats. In: Tjoa, A.M., Quirchmayr, G., You, I., Xu, L. (Eds.), Availability, Reliability and Security for Business, Enterprise and Health Information Systems, pp. 295–308.
Pang, G., Shen, C., Cao, L., Hengel, A.V.D., 2022. Deep learning for anomaly detection: a review. ACM Comput. Surv. 54 (2), 1–38. doi:10.1145/3439950.
Park, S., Kim, M., Lee, S., 2018. Anomaly detection for HTTP using convolutional autoencoders. IEEE Access 6, 70884–70901.
Perdisci, R., Ariu, D., Fogla, P., Giacinto, G., Lee, W., 2009. McPAD: a multiple classifier system for accurate payload-based anomaly detection. Comput. Netw. 53 (6), 864–881.
Prajapati, P., Patel, N., Shah, P., 2019. A review of recent detection methods for HTTP ddos attacks. Int. J. Sci. Technol. Res. 8 (12), 1693–1696.
Ren, X., Hu, Y., Kuang, W., Souleymanou, M.B., 2018. A web attack detection technology based on bag of words and hidden markov model. In: Proceedings - 15th IEEE International Conference on Mobile Ad Hoc and Sensor Systems, MASS 2018. Institute of Electrical and Electronics Engineers Inc., pp. 526–531.
Riera, T.S., Higuera, J.R.B., Higuera, J.B., Herraiz, J.J.M., Montalvo, J.A.S., 2020. Prevention and fighting against web attacks through anomaly detection technology. A systematic review. Sustainability (Switzerland) 12 (12).
Ring, M., Wunderlich, S., Scheuring, D., Landes, D., Hotho, A., 2019. A survey of network-based intrusion detection data sets. Comput. Secur. 86, 147–167.
Salazar-Hernández, R., Díaz-Verdejo, J.E., 2010. Hybrid detection of application layer attacks using markov models for normality and attacks. In: Proceedings of the 12th International Conference on Information and Communications Security, pp. 416–429.

Sarmah, U., Bhattacharyya, D.K., Kalita, J.K., 2018. A survey of detection methods for XSS attacks. J. Netw. Comput. Appl. 118, 113–143. doi:10.1016/j.jnca.2018.06.004.
Sommer, R., Paxson, V., 2010. Outside the closed world: on using machine learning for network intrusion detection. In: 2010 IEEE symposium on security and privacy. IEEE, pp. 305–316.
Song, Y., Keromytis, A.D., Stolfo, S.J., 2009. Spectrogram: a mixture-of-markov-chains model for anomaly detection in web traffic. Technical Report. NDSS Symposium 2009.
Sureda Riera, T., Bermejo Higuera, J.R., Bermejo Higuera, J., Martínez Herraiz, J.J., Sicilia Montalvo, J.A., 2020. Prevention and fighting against web attacks through anomaly detection technology. A systematic review. Sustainability 12 (12), 4945. doi:10.3390/su12124945.
Tang, R., Yang, Z., Li, Z., Meng, W., Wang, H., Li, Q., Sun, Y., Pei, D., Wei, T., Xu, Y., Liu, Y., 2020. ZeroWall: detecting zero-day web attacks through encoder-decoder recurrent neural networks. In: Proceedings - IEEE INFOCOM, volume 2020-July. Institute of Electrical and Electronics Engineers Inc., pp. 2479–2488.
Tavallaee, M., Stakhanova, N., Ghorbani, A.A., 2010. Toward credible evaluation of anomaly-based intrusion-detection methods. IEEE Trans. Syst. Man Cybernetic. Part C: Appl. Rev. 40 (5), 516–524.
Tekerek, A., 2021. A novel architecture for web-based attack detection using convolutional neural network. Comput. Secur. 100, 102096.
Torrano-Gimenez, C., Perez-Villegas, A., Alvarez, G., 2009. A self-learning anomaly-based web application firewall. In: Advances in Intelligent and Soft Computing. Springer, Berlin, Heidelberg, pp. 85–92.
de la Torre-Abaitua, G., Lago-Fernández, L.F., Arroyo, D., 2021. On the application of compression-based metrics to identifying anomalous behaviour in web traffic. Logic J. IGPL 28 (4), 546–557.
Tufan, E., Tezcan, C., Acarturk, C., 2021. Anomaly-based intrusion detection by machine learning: a case study on probing attacks to an institutional network. IEEE Access.
Vartouni, A.M., Teshnehlab, M., Kashi, S.S., 2019. Leveraging deep neural networks for anomaly-based web application firewall. IET Inf. Secur. 13 (4), 352–361.
Wang, S., Song, J., Guo, R., 2019. Char-level neural network for network anomaly behavior detection. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), volume 11354 LNCS, pp. 60–68.
Wu, K., Chen, Z., Li, W., 2018. A novel intrusion detection model for a massive network using convolutional neural networks. IEEE Access 6, 50850–50859.
Yu, Y., Yan, H., Guan, H., Zhou, H., 2018. DeepHTTP: semantics-structure model with attention for anomalous HTTP traffic detection and pattern mining. arXiv.
Zhang, S., Li, B., Li, J., Zhang, M., Chen, Y., 2016. A novel anomaly detection approach for mitigating web-based attacks against clouds. In: Proceedings - 2nd IEEE International Conference on Cyber Security and Cloud Computing, CSCloud 2015 - IEEE International Symposium of Smart Cloud, IEEE SSC 2015, pp. 289–294.
Zolotukhin, M., Hamalainen, T., Juvonen, A., 2012. Online anomaly detection by using N-gram model and growing hierarchical self-organizing maps. In: IWCMC 2012 - 8th International Wireless Communications and Mobile Computing Conference, pp. 47–52.
Zuech, R., Khoshgoftaar, T.M., Wald, R., 2015. Intrusion detection and big heterogeneous data: a survey. J. Big Data 2 (1).

JESÚS E. DÍAZ-VERDEJO received the MS and Ph.D. in Physics from the University of Granada in 1989 and 1995 respectively. He is a professor in the Department of Signal Theory, Telematics and Communications at the University of Granada. His initial research interest was focused on speech technologies. Currently, his research and teaching are in the areas of networking and cybersecurity, especially in intrusion detection systems, network security monitoring and traffic engineering.

RAFAEL ESTEPA received the M.S. and Ph.D. degrees in telecommunication engineering from the University of Seville, in 1998 and 2002, respectively, where he is currently an Associate Professor with the Department of Telematics Engineering. In the past, he was working for two years as a Product Engineer with Alcatel, Spain. He has also been a Visitor with the Department of Applied Mathematics, Instituto Superior Tecnico (IST), Lisbon, and the Dublin Institute of Technology (DIT). His research interests include the areas of networking, voice over IP (VoIP) quality of service, wireless networks, unmanned aerial vehicles (UAVs), and cybersecurity.

ANTONIO ESTEPA received the M.S. and Ph.D. degrees in telecommunication engineering from the University of Seville, in 1998 and 2004, respectively. In 2004, he was also a Visitor with the Department of Electrical Engineering and Computer Science, University of Minnesota, USA. He is currently an Associate Professor with the Department of Telematics Engineering, University of Seville. From 1998 to 2000, he was a software and network engineer with a software development company. He has authored or coauthored several conferences or journal papers. His research interests include the areas of telecommunication networks, with particular emphasis in networking protocols, wireless networks, and cybersecurity.

GERMAN MADINABEITIA received the M.S. and Ph.D. degrees in telecommunication engineer- ing from the Universidad Politecnica de Madrid, in 1986 and 2004, respectively. In the past, he was working for ten years as a product engineer in the industry. He is currently an Assistant Profes- sor with the Department of Telematics Engineering, University of Seville. His research interests include the areas of networking, the Internet of Things, cybersecurity, and traffic engineering.