SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

March 2018, Vol. 61 032114:1–032114:16
https://doi.org/10.1007/s11432-017-9288-4

An adaptive system for detecting malicious queries in web attacks

Ying DONG[1], Yuqing ZHANG[1,2*], Hua MA[2,3], Qianru WU[4], Qixu LIU[1,2], Kai WANG[5] & Wenjie WANG[1]

[1]*National Computer Network Intrusion Protection Center, University of Chinese Academy of Sciences, Beijing* 101408*, China;*
[2]*State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing* 100093*, China;*
[3]*School of Mathematics and Statistics, Xidian University, Xi'an* 710071*, China;*
[4]*Security Department, Alibaba Group, Beijing* 100026*, China;*
[5]*Zhanlu Laboratory, Tencent Incorporation, Beijing* 100080*, China*

Received 1 August 2017/Accepted 27 October 2017/Published online 2 February 2018

Abstract Web request query strings (queries), which pass parameters to a referenced resource, are always manipulated by attackers to retrieve sensitive data and even take full control of victim web servers and web applications. However, existing malicious query detection approaches in the literature cannot cope with changing web attacks. In this paper, we introduce a novel adaptive system (AMOD) that can adaptively detect web-based code injection attacks, which are the majority of web attacks, by analyzing queries. We also present a new adaptive learning strategy, called SVM HYBRID, leveraged by our system to minimize manual work. In the evaluation, an up-to-date detection model is trained on a ten-day query dataset collected from an academic institute's web server logs. The evaluation shows our approach overwhelms existing approaches in two respects. Firstly, AMOD outperforms existing web attack detection methods with an F-value of 99.50% and FP rate of 0.001%. Secondly, the total number of malicious queries obtained by SVM HYBRID is 3.07 times that by the popular support vector machine adaptive learning (SVM AL) method. The malicious queries obtained can be used to update the web application firewall (WAF) signature library.

Keywords web attacks, adaptive learning, intrusion detection, anomaly detection, SVM

Citation Dong Y, Zhang Y Q, Ma H, et al. An adaptive system for detecting malicious queries in web attacks. Sci China Inf Sci, 2018, 61(3): 032114, https://doi.org/10.1007/s11432-017-9288-4

1 Introduction

Web attacks are attacks that exclusively use the HTTP/HTTPS protocol. Symantec internet security threat report (ISTR) [1] revealed that the number of web attacks had explosively increased, reaching 1.1 million every day in 2016, more than twice the rate in 2015 (0.783 million). Among web attacks, code injection attacks against web applications [2] increased each year and accounted for at least 96.15% of the entire web attacks in 2016, according to the Imperva web application attack report (WAAR) [3] and WhiteHat web application security statistic report (WASSR) [4]. In 2015, Team GhostShell claimed to have hacked numerous websites using SQL injection (SQLI) attacks [5], and disclosed thousands of compromised account details [6].

* Corresponding author (email: zhangyq@ucas.ac.cn)

This paper focuses on the most frequent types of web-based code injection attacks [7] in 2016, namely, cross-site scripting (XSS) [3], SQLI, directory traversal (DT) [3], and remote file inclusion (RFI) [3], each accounting for 49.09%, 28.32%, 9.82% and 8.92% of the entire web attacks [1, 4]. In the following, web attacks refer to web-based code injection attacks. Attack vectors of web attacks exist in user input [8]. Whenever user input is improperly handled, these attacks can succeed. Since most user input data exists in queries[1)], detecting malicious queries is the core of detecting web attacks. For web applications whose source code is unavailable, web intrusion detection systems (IDSs) are the only option [9]. A web IDS acts as an intermediate layer between the protected web application and users, and analyzes web traffic to detect possibly malicious activities.

A considerable amount of effort has been made to detect malicious queries in web requests using web IDSs. Two major detection approaches are utilized: signature-based detection and anomaly-based detection. Signature-based approaches are effective in detecting known attacks with a low false positive (FP) rate, but unable to detect previously unknown attacks (e.g., zero-day attacks). A popular signature-based detection approach is WAF. In contrast to signature-based approaches, anomaly-based approaches [10–15] can detect unknown attacks, but with a high FP rate. The ability to detect unknown attacks has drawn wide academic attention to anomaly-based approaches.

However, most existing web attack detection models using anomaly detection are unaware of behavior changes of web attacks, which we call constant learning. Adaptive learning overcomes this shortcoming by frequently improving the detection ability to adapt to the changing attacks [16,17]. Previous adaptive attack detection methods [18–20] are designed for network intrusions [21, 22] and are not applicable to detecting malicious queries in web attacks. A practical solution for keeping the web attack detection model always updated is to incorporate both informative queries and representative malicious queries, which together we call important queries. An intuitive approach to obtaining important queries is randomly selecting requests from web traffic and then manually labeling them. Unfortunately, most randomly selected queries are similar benign ones, with a low probability of being important, leading to considerable manual labeling work. This challenge has motivated our research to build a malicious query detection system that can adaptively incorporate the latest important queries to update the detection model.

In this paper, we address the issues related to the adaptive detection of malicious queries in web attacks. We present a novel adaptive system (AMOD) for this purpose. AMOD leverages an efficient adaptive learning strategy, SVM HYBRID, which is a hybrid of suspicion selection (SS) and exemplar selection (ES). SS features in acquiring the most important informative queries, namely, suspicions, while ES specializes in harvesting representative malicious queries, exemplars. Suspicions and exemplars are called important queries. AMOD progressively enhances its detection ability by periodically checking the latest important queries from incoming traffic. The number of important queries is trivial, so manual labeling work is minimized. The important queries are then incorporated into the training pool to update the detection model, which is a meta-learning [23,24] based ensemble classifier [25], with SVM as its meta classifier. A ten-day query dataset collected from an academic institute website's web server logs is used for evaluation. Malicious queries obtained by our system can be used to update the WAF signature library.

The main contributions of this study are as follows:

- We introduce AMOD capable of detecting web-based code injection attacks, which are the majority of web attacks.
- We present SVM HYBRID, a new adaptive learning strategy leveraged by AMOD. SVM HYBRID composes of SS and ES, the contribution of SVM HYBRID is also a hybrid of theirs, namely improving detection performance and harvesting truly malicious queries respectively.
- Evaluation demonstrates that AMOD outperforms existing web attack detection methods on both a real-world ten-day query set and the HTTP dataset CSIC 2010 [26] (with F-values of 99.50% and 99.96%, FP rates of 0.001% and 0.03%, respectively). The total number of malicious queries obtained by SVM HYBRID is 3.07 times that by SVM AL during the ten-day detection using AMOD.

1) Web request query string is consistently termed as query in this paper for simplicity.

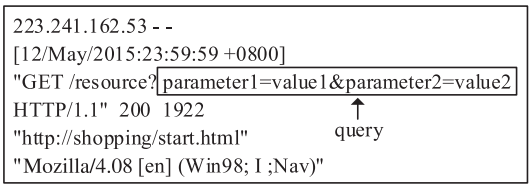

Figure 1 Sample entry in web server access logs.

Table 1 Comparison of normal queries and malicious queries in web-based code injection attacks

No.	Type	Query in a web request URL
(a)	Normal	http://site/index.php?postID=123
(b)	SQLI	http://site/index.php?postID='/**/union/**/select/**/0, concat(username,0x3a,password)/**/from/**/users/*'
(c)	XSS	http://site/index.php?postID=<script>document.location ="http://mal_site/stealcookie.cgi?"+document.cookie</script>
(d)	DT	http://site/index.php?postID=../../../../etc/passwd
(e)	RFI	http://site/index.php?postID=http://mal_site/hack.txt?ls

The remainder of the paper is organized as follows. Section 2 provides background and related work. Detailed design and implementation of the system will be presented in Section 3. Section 4 shows data preparation. Next, Section 5 introduces the process of determining the best configuration of AMOD. In addition, AMOD is compared with a constant stacking approach and five adaptive approaches, in terms of both the detection performance and the number of malicious queries obtained. Detection performance comparisons with related work are also demonstrated. Afterwards, we clarify some relevant issues of our approach in Section 6. Finally, Section 7 concludes this paper.

2 Background and related work

2.1 Background

Queries contain most user input and are thus closely related to web-based code injection attacks, which are the most threatening web attacks [1, 3]. Queries can be found in HTTP GET requests recorded in web server logs. Figure 1 shows a sample entry in web server access logs in common log format (CLF). The framed text in the request string is a query, which our system examines. A query is identified by a leading "?" character following the referenced resource, and lists pairs of parameter names and values.

Table 1 compares a normal request and four types of code injection attacks analyzed in this paper: SQLI, XSS, DT, and RFI. We include remote code execution (RCE) into XSS attacks, since RCE is essentially a special kind of XSS attacks [27]. The requests in Table 1 are forwarded to a vulnerable script named "index.php" in a forum website, and the script can extract posts from the database and present a post to a user. "postID" is a parameter name for this script and takes an integer input value as the post ID while recording the transaction. As shown in Table 1, (a) is a normal query, and (b)–(e) are malicious. It can be seen that malicious queries in web-based code injection attacks and normal queries are different in both character distribution and character sequences.

2.2 Related work

In the literature, most network intrusion detection approaches used HTTP datasets with very few web attacks [28]. Based on whether a detection method can adapt to the change of web attacks, we divide existing web attack and HTTP attack detection techniques that use anomaly detection schemes into two categories: adaptive detection and constant detection. Adaptive detection methods always incorporate new traffic data to induce an up-to-date detection model, while constant detection methods use a period of traffic data to build a detection model once and use it for all later detection.

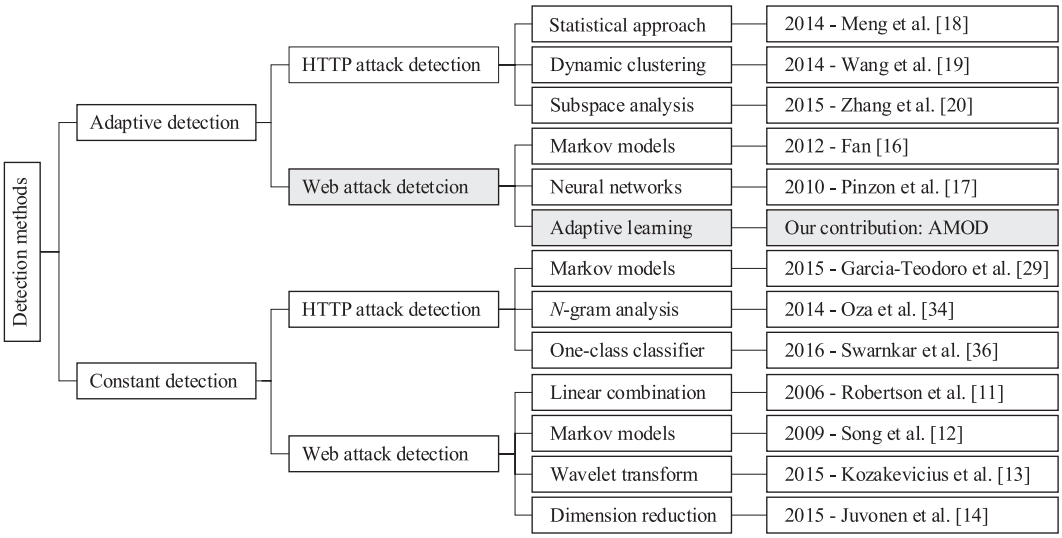

Figure 2 Taxonomy of the latest academic researches on anomaly detection methods of web attacks and HTTP attacks.

Unfortunately, most adaptive detection techniques [18–20] are designed for HTTP attacks, whose landscape differs from web attacks': malicious queries in web attacks have more subtle distinctions from legitimate queries. Existing adaptive detection approaches of web attacks either ignored parameter values in queries [16], or only focused on SQLI attacks [17]. The adaptive detection of various types of web attacks has been found deficient. In this paper, AMOD is proposed to bridge this gap. Figure 2 gives our taxonomy of the latest researches on anomaly detection of web attacks and HTTP attacks.

2.2.1 *Adaptive detection*

Adaptive detection can detect the latest unknown attacks by adapting to the changing behaviors of attacks, yet most adaptive detection methods to date only have been related to HTTP attack detection.

HTTP attack detection. These detection methods analyze packet payloads in network traffic of port 80, namely HTTP traffic. Three types of unsupervised learning techniques are utilized. (1) Statistic based. Meng et al. [18] used a statistic-based approach to develop an adaptive blacklist-based packet filter, which can help filter out network packets based on IP confidence. (2) Dynamic clustering. Wang et al. [19] employed affinity propagation (AP) in dynamical clustering to detect HTTP attacks adaptively. (3) Subspace analysis. Zhang et al. [20] proposed adaptive stream projected outlier deTector (A-SPOT), which used a novel adaptive subspace analysis approach to detect anomalies from network connections.

Web attack detection. Existing adaptive detection approaches of web attacks are as follows. (1) Markov Models. Fan [16] used multiple hidden Markov models (HMMs) as an adaptive model to detect web attacks. HMMs were built on benign requests and can identify attacks by calculating discreteness of requests. (2) Neural networks. AIIDA-SQL [17] was an adaptive detector of SQLI attacks, using case-based reasoning engine to achieve adaptability. Artificial neural networks and SVM were combined as the detection model to detect malicious SQL queries in HTTP requests.

2.2.2 *Constant detection*

In terms of constant detection methods for HTTP attacks, Markov models were used by Garcia-Teodoro et al. [29], Zhong et al. [30] and Ariu et al. [31] to represent the normal behaviors of a service in a network intrusion detection system (NIDS). N-gram technique is also widely used in NIDSs, such as PAYL [32], Anagram [33], RePIDS [28] and literature [34]. One-class classifiers were first used in NIDSs by McPAD [35] to detect HTTP attacks, and recently used by Swarnkar et al. [36] and Duessel et al. [37]. As for web attacks, linear combination was first used by Kruegel et al. [10], and then used by Robertson et al. [11] to compute the anomaly score of a web request. Besides, Markov models were also used to

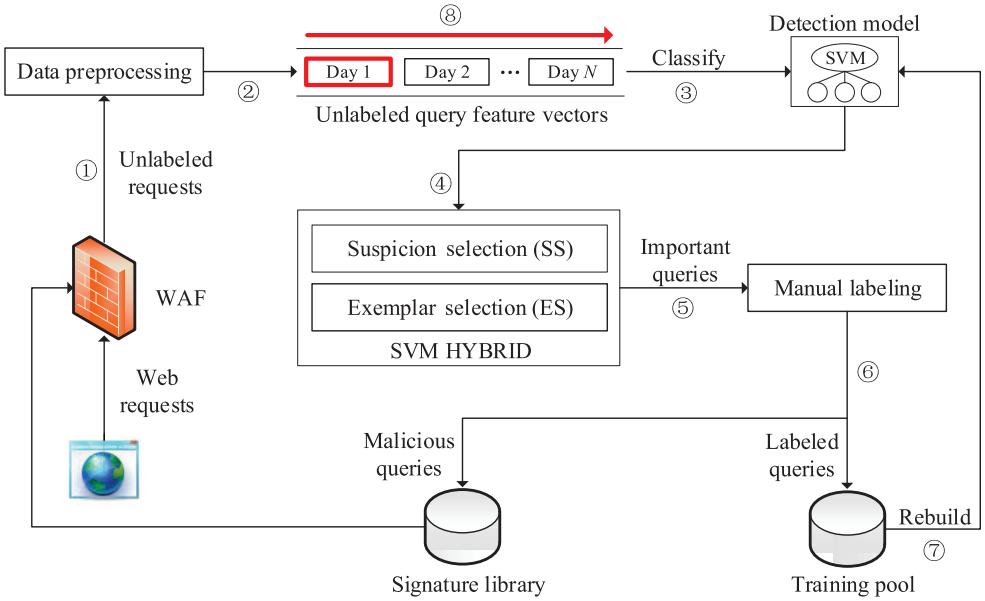

Figure 3 (Color online) System overview.

model queries [12]. Wavelet transform and dimension reduction were used as query anomaly sensor by Kozakevicius et al. [13] and Juvonen et al. [14], respectively.

3 Design and implementation

In this section, we first illustrate the design of our proposed system for adaptive malicious query detection (AMOD, "O" for "Q"), followed by the core adaptive learning technique, named SVM HYBRID. Finally, we introduce our stacking-based training and classification.

3.1 System design

In this paper, we introduce the concept of important queries, which consist of the most informative normal queries and the most representative malicious queries, as mentioned in Section 1. The goal of AMOD is to progressively enhance the ability to identify malicious queries by periodically checking the latest important queries. To reach this goal, our system leverages an adaptive learning approach called SVM HYBRID to obtain important queries from huge unlabeled traffic.

The system is initialized with a training pool composed of a small set of labeled queries, from which the initial detection model is trained. The detection model is based on stacking, and composed of three base classifiers and a meta classifier. SVM is used as its meta classifier, so that SVM HYBRID could exert the detection model to obtain important queries. To update the initial detection model, the system takes web server logs as its data source, and examines unlabeled queries in HTTP GET requests in batch mode. Classified unlabeled queries are analyzed by SVM HYBRID to obtain important queries for labeling. Thus, the manual work is greatly reduced. As shown in Figure 3, the system proceeds as follows.

(1) Collecting unlabeled requests. We consider it unbeneficial to detect attacks detectable by WAF, since there must be signatures in the WAF signature repository that can match these attacks. Hence, only unlabeled requests containing benign queries and malicious queries undetected by WAFs are collected.

(2) Data preparation (Section 4). Data preparation transforms requests into feature vectors of queries in batches of N days, including data preprocessing, feature construction and reduction.

(3) Classification (Subsection 3.3). Unlabeled queries on the 1st day are classified using the current detection model, and the classification results are recorded.

(4) Queries selection (Subsection 3.2). A fixed number (150) of important queries are obtained from classified unlabeled queries on the 1st day using SVM HYBRID.

(5) Label the obtained important queries manually to identify their exact labels.

(6) Add all the labeled important queries into the training pool[2)]. Update the WAF signature library using the obtained important queries that are labeled as malicious.

(7) Update the detection model (Subsection 3.3). Retrain the detection model on current training pool.

(8) Iterate steps (3)–(7) for unlabeled queries of the rest $N-1$ days.

3.2 SVM HYBRID

SVM HYBRID, our proposed adaptive learning strategy, operates in the kernel feature space of SVM, and relies on positions of queries in SVM kernel feature space to decide queries for labeling. Given training set $T=\{(x_1,y_1),\ldots,(x_n,y_n)\}$, where $x_i\in\mathbb{R}^N$ is a feature vector, and $y_i\in\{-1,1\}$ is a class label, the margin is defined as the maximal perpendicular distance between samples of the two classes [38]. SVM aims to find the optimal linear classifier whose separating hyperplane maximizes the margin. When original samples are non-linearly separable, they could be mapped to a new kernel feature space where they are linearly separable. Given a feature mapping process φ, the corresponding kernel is defined as

$$K(x,z)=\varphi(x)^{\mathrm{T}}\varphi(z). \tag{1}$$

For data that are non-linearly separable though mapped to infinite dimensional feature space, the optimization problem can be formulated as

$$\begin{aligned}&\max_{\alpha}\quad \sum_{i=1}^{n}\alpha_i-\frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j x_i^{\mathrm{T}}x_j,\\ &\text{s.t.}\quad 0\leqslant\alpha_i\leqslant C\ (i=1,\ldots,n),\ \sum_{i=1}^{n}\alpha_i y_i=0,\end{aligned} \tag{2}$$

where α_i is non-zero only for samples that are closest to the hyperplane, namely support vectors, the number of which is usually small. Therefore, the kernel distance is efficient to compute. Penalty parameter C controls the relative weighting of a large margin and low misclassification rate of training data.

The illustrative graph of SVM HYBRID is presented in Figure 4. The middle solid line is the separating hyperplane, while the two external solid lines indicate the margin. The upper side of the hyperplane is the malicious side. "+" represents a query in the positive class, namely, a malicious query, while "−" stands for a query in the negative class: a normal query. SVM HYBRID comprises of SS and ES. SS obtains the most important informative queries, which are suspected of being malicious and thus are called suspicions. Suspicions are within the prone-to-be-misclassified region, called the confusing region: the grey area delimited by two dashed lines in Figure 4. The two points surrounded by red circles within the confusing region are suspicions. Conversely, ES obtains the most representative malicious queries, which are called exemplars, since each of them is an exemplar of similar malicious queries. Exemplars lie in the malicious side and are far away from the hyperplane. As shown in Figure 4, the four points surrounded by red circles on the upper side are exemplars.

The main contribution of SS and ES is also different: the former contributes more to improving detection performance, while the latter harvesting truly malicious queries. The contribution of SVM HYBRID is a hybrid of these two strengths. Each unlabeled query set is partitioned into two disjointed subsets for SS and ES respectively. SVM HYBRID starts with SS on one subset and then switches to ES on the other subset. The partition ratio is an adjustable parameter in SVM HYBRID to address the tradeoff between detection performance improvement and the number of malicious queries obtained.

3.2.1 *Suspicion selection (SS)*

Adaptive learning with SVM uncertainty sampling is also known as SVM AL [39–41]. SVM AL and SS both select unlabeled samples within the margin. However, SVM AL simply selects samples the closest

2) In the following, the terms "training pool", "training set", and "labeled set" are used interchangeably.

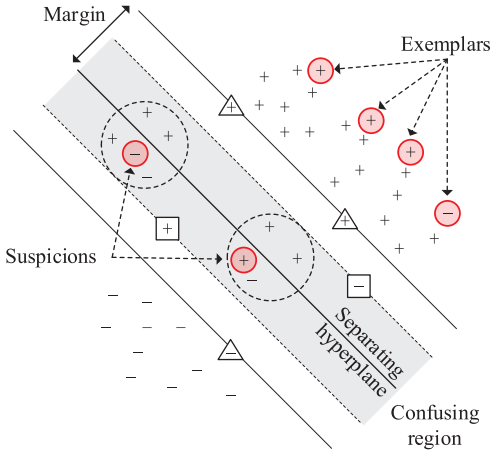

Figure 4 (Color online) Illustrative graph of SVM HYBRID.

to the hyperplane and does not consider their distribution. Consequently, similar samples might be selected, making SVM AL prone to sample bias. Labeling similar samples is also a waste of manual work. Therefore, given a large pool of unlabeled samples, it is desirable to preserve the density distribution [42] of samples in the pool during the learning process. Our proposed SS pursues the idea of preserving data distribution by using clustering to find potential informative queries from unlabeled queries within the confusing region. In brief, SVM AL concentrates on the most informative data, while SS focuses on the most important informative data so that data redundancy is reduced to the minimum. The procedure of SS is as follows.

Firstly, the confusing region is determined. The set of misclassified queries in a training set that fall into the margin during training is denoted as Q. Queries in Q are sorted in order according to their kernel distances computed using (3),

$$f(x) = \sum_{i=1}^{n} \alpha_i y_i K(x_i, x) + b. \tag{3}$$

The query in Q with the minimal kernel distance (the "+" point with the rectangular frame in Figure 4) determines the lower boundary of the confusing region (the lower dashed line). Oppositely, the query in Q with the maximal kernel distance (the "−" point with the rectangular frame) determines the upper boundary of the confusing region (the upper dashed line). That is,

$$f_{\text{lower}} = \min_{x \in Q} f(x), \quad f_{\text{upper}} = \max_{x \in Q} f(x). \tag{4}$$

To ensure the existence of the confusing region, the margin should be wide enough to allow misclassifications of training samples within the margin, which can be achieved by adjusting the penalty parameter C and SVM kernel parameter of the meta classifier SVM.

Secondly, K-medoids clustering [43] is performed on the unlabeled queries that fall into the confusing region. The final cluster centers are suspicions. K-medoids begins with K randomly selected samples as medoids (cluster centers). Then each remaining sample is assigned to the cluster whose medoid is the closest to it. That is, the partitioning method is based on minimizing the sum of the distances between each sample and its corresponding referenced medoid. This sum is denoted as E, and is defined as

$$E = \sum_{j=1}^{k} \sum_{x \in C_j} |\varphi(x) - \varphi(M_j)|, \tag{5}$$

where x is a given sample in cluster C_j, whose cluster center is M_j. $|\varphi(x) - \varphi(M_j)|$ is the Euclidean distance between x and M_j in kernel feature space, and using (1), it can be given by

$$|\varphi(x) - \varphi(M_j)| = \sqrt{K(x,x) + K(M_j, M_j) - 2K(x, M_j)}. \tag{6}$$

The K cluster centers with which E reaches its minimum value are optimal. The partitioning step is repeated until the number of unlabeled samples within the confusing region is less than K. When partitioning terminates, the final cluster centers are suspicions.

Figure 4 shows that the confusing region covers two clusters, each represented by a dashed circle. Their cluster centers are the two points surrounded by red circles in the two dashed circles. One cluster center is a normal query and another is a misclassified malicious one. In other words, suspicions might be misclassified and also might not be malicious. On the contrary, queries obtained by ES are highly likely malicious. ES is described in the following subsection.

3.2.2 *Exemplar selection (ES)*

ES serves to obtain representative malicious queries, called exemplars, from the unlabeled queries. New web attacks are very commonly variants of existing attacks. Queries in variant attacks might be similar to those in existing attacks. Labeling similar queries would unnecessarily increase manual labeling workload. As a result, we hope that each exemplar is the unlabeled malicious query that differs most from all the malicious queries in the current labeled set.

To attain this goal, we apply the kernel farthest-first (KFF) algorithm [44], a greedy heuristic, to obtain exemplars from the malicious side of the SVM hyperplane. KFF always chooses from the unlabeled query set the query that is the farthest from all the malicious queries in the labeled set. Given unlabeled query set U and labeled query set L, the farthest distance is defined as the maximum of the sum of distances between each malicious query y in L and a query x in U in kernel feature space. Formally, the optimal x is determined by

$$\arg\max_{x\in U} \Big\langle \sum_{x\in L} |\varphi(x) - \varphi(y)| \Big\rangle, \tag{7}$$

where $|\varphi(x) - \varphi(y)|$ is computed using (6). Each optimal x is an exemplar, and is added to the labeled set. As can be seen from Figure 4, ES obtains four exemplars, represented by four points surrounded by red circles. One of the exemplars is a misclassified normal query, yet most exemplars are truly malicious. At each iteration of the adaptive detection process, suspicions and exemplars are manually labeled to refine the decision boundary of the meta classifier SVM of the detection model.

To conclude, SS uses the kernel distance of unlabeled queries to the hyperplane, which is already solved when a query is classified, while ES uses the kernel distance of unlabeled ones to malicious ones in current training set, which could be computed using the efficient-to-compute kernel function.

3.3 Training and classification

In our approach, we employ stacked generalization [45], also named stacking, a well-known meta-learning technique, as our underlying detection model. The ability to learn from different learning biases makes meta-learning an appropriate approach for malicious query detection, due to the complex, dynamic, confrontational nature of its problem domain. The key idea of stacking is to train a meta classifier from the prediction results of base classifiers [45]. During training, stacking tries to induce which base classifiers are reliable and which are not by using a meta classifier. During classification, a test sample is classified by each of the base classifiers; then these classifications are fed into the meta classifier that makes the final decision. The base classifiers in our stacking detection model are chosen from three different classifier families. This combination scheme of base classifiers serves to obtain learning diversity and overcome the learning bias of classifiers in the same category. The meta classifier in our stacking detection model is SVM so that SVM HYBRID can operate on our model by exerting its meta classifier.

4 Data preparation

The web server access logs of an academic institute web application during the period of July 1–July 10, 2016 are used to evaluate our system. The institute uses an Apache web server, whose logs are in

Table 2 Description of web server logs and data preprocessing

Item	Raw logs		Data preprocessing			
	Duration (days)	Log size (GB)	Original requests	Cleaned queries	Normalized queries	Filtered queries
#	10	31.1	4764598	1882901	1189228	1123497

Table 3 Statistics of query dataset

#Queries	Benign	Malicious	Total
Initial set	900	100	1000
Ten unlabeled sets	990000	10000	1000000

CLF format as shown in Figure 1. In addition, the sever is equipped with a commercial WAF, called Hillstone [46], and only requests that pass through the WAF are recorded in its web server access logs. The institute website contains many dynamic pages which accept user input, and thus its web traffic contains a large number of the web attacks examined in our paper (see Subsection 2.1). During data preparation, data preprocessing is performed to obtain queries which are later used in feature construction and reduction.

4.1 Data preprocessing

Our system models queries in HTTP GET requests shown in Figure 1. These requests are preprocessed using following steps. (1) Data cleaning. Requests whose return code is equal to or greater than 200, and less than 300 are collected. Then, static requests are removed and the remaining requests are parsed to extract queries. (2) Data normalization. These steps are performed in order for the cleaned queries: decoding printable ASCII characters, un-escaping, transforming to lowercase and removing queries whose length is less than four. (3) Character filter. The general syntax of URI in RFC 2616 [47] defines the unsafe characters that should not appear in standard queries. We consider queries that contain any unsafe characters as malicious and use a character filter to remove these queries. Note that we consider queries with obfuscated code as malicious and do not make further analysis on them, because the goal of obfuscation in web requests is to evade detection; it is futile to obfuscate benign web requests.

A description of the raw logs and data preprocessing procedure is presented in Table 2. The number of queries after each step of data preprocessing is shown. As presented in Table 3, eleven disjointed query sets created from the remaining 1123497 queries are used in our experiments. The initial set is used to induce the initial detection model, while the ten unlabeled sets (each containing 1000 malicious queries and 99000 benign ones) are used to update the model.

4.2 Feature construction and reduction

Feature construction transforms each query into a numerical feature vector of N-grams, and is performed on the eleven sets used in our experiment. An N-gram is an N-character slice of a string, which is fully automatic and requires no prior knowledge of target web application and target attacks. As for the value of N, we choose N=2 to balance computational consumption and detection performance; it also captures both character distribution and the sequence information, which is necessary for attack detection in strings [28,32–34]. Since 63 unique characters exist in all the queries after data preparation, 63^2 (3969) distinct N-grams are used in feature construction. To capture the structure of queries more accurately, a frequency-based N-gram model instead of a binary-based model is applied. Each N-gram frequency is then divided by the most frequently appearing N-gram frequency in the same query.

5 Experiments and evaluation

In this section, we report and analyze the results of a series of experiments. Experiment 1 (Subsection 5.1) determines the best configuration of AMOD. Experiment 2 (Subsection 5.2) compares AMOD with a

Table 4 Determining the best setting

Feature reduction		Stacking model		SVM HYBRID	
IG	PCA	Base classifiers	Meta classifier	Partition	#Obtained queries (/day)
800	80	RF, Logistic, MLP	SVM-RBF	1:1	150

constant stacking approach and five adaptive approaches in terms of detection performance and the number of malicious queries obtained. Lastly, Experiment 3 (Subsection 5.3) compares AMOD with the most related work.

All the experiments are implemented in Java and on a machine with 32-GB memory and Inter® Core™ 2.93-GHz CPU. To allow SVM HYBRID to work well with the meta classifier SVM of the detection model, we implement stacking on our own. Single classifiers are applied using WEKA, except that SVM is applied using LibSVM. Primary metric employed for detection performance evaluation is F-value along with true positive (TP) rate and FP rate. As defined in (8), F-value is a combination of Precision and Recall. β corresponds to the relative importance of Precision versus Recall and is set to default 1.

$$F\text{-value} = (1+\beta)^2 * \frac{\text{Precision}}{\beta^2 * (\text{Precision} + \text{Recall})}. \tag{8}$$

5.1 Experiment 1: determining the best setting of AMOD

Experiment 1 aims to determine the best setting of AMOD as shown in Table 4, and is performed on the superset of the eleven query sets shown in Table 3 using 10-fold cross validation. Queries in the superset are represented by N-gram feature vectors in the 3969-dimensional feature space (see Subsection 4.2). The determination metric is the average F-value of each setting.

Feature reduction. In order to reduce the time in dimensionality reduction and training as well as improve classification performance [48], we apply feature selection before dimensionality reduction. Various combinations of feature selection methods (chi-square test, information gain (IG) and document frequency (DF)) and top selection (150, 200, 300, 500, 800, 1100, 1500 and 2000) are tested. IG with top 800 overwhelms other combinations, and thus is chosen. Then, different combinations of dimensionality reduction methods (principal component analysis (PCA) and random projection (RP)) and reservation quantities (10, 20, 30, 40, 60, 80, 150 and 300), as well as a full feature set, are also tested. PCA with top 80 reserved dimensionalities outperforms others and thus is chosen.

Classifier selection. Using the best setting of feature reduction, classifier selection is carried out in three steps. (1) Determine base classifier candidates. Twelve classifiers chosen from various classifier families are tested. MLP, random forest, JRip and simple logistic perform best in their respective family. Hence, they are chosen as base classifier candidates. (2) Determine the best combination of base classifiers. All combinations of base classifier candidates are tested in stacking with the meta classifier being SVM with RBF and polynomial kernel respectively. The combination of random forest, logistic and MLP outperforms other combinations, and thus is chosen. (3) Determine the best meta classifier SVM. In terms of meta classifier SVM, RBF kernel outperforms the polynomial kernel, thus SVM with RBF kernel is used as the meta classifier of the detection model. The optimal pair of SVM penalty parameter C and RBF kernel parameter γ is $(C, \gamma) = (0.05, 2)$, obtained using grid-search cross-validation [49].

SVM HYBRID parameters. SVM HYBRID impacts AMOD in both detection performance and the number of malicious queries obtained by adjusting partition ratio (the ratio of the size of query subset for SS to that for ES), which is set to 1:1 in our experiment, since we expect that our system could maintain a desirable detection performance and simultaneously obtain as many malicious queries as possible. The number of obtained important queries (suspicions and exemplars) each day is set to 150.

5.2 Experiment 2: comparisons of constant approach and adaptive approaches

Experiment 2 aims to compare AMOD with a constant stacking approach and five adaptive detection approaches, including SS, ES, random selection, SVM AL and adaptive SVM. All the approaches are implemented in the adaptive learning framework of AMOD (see Subsection 3.1) to carry out the ten-day

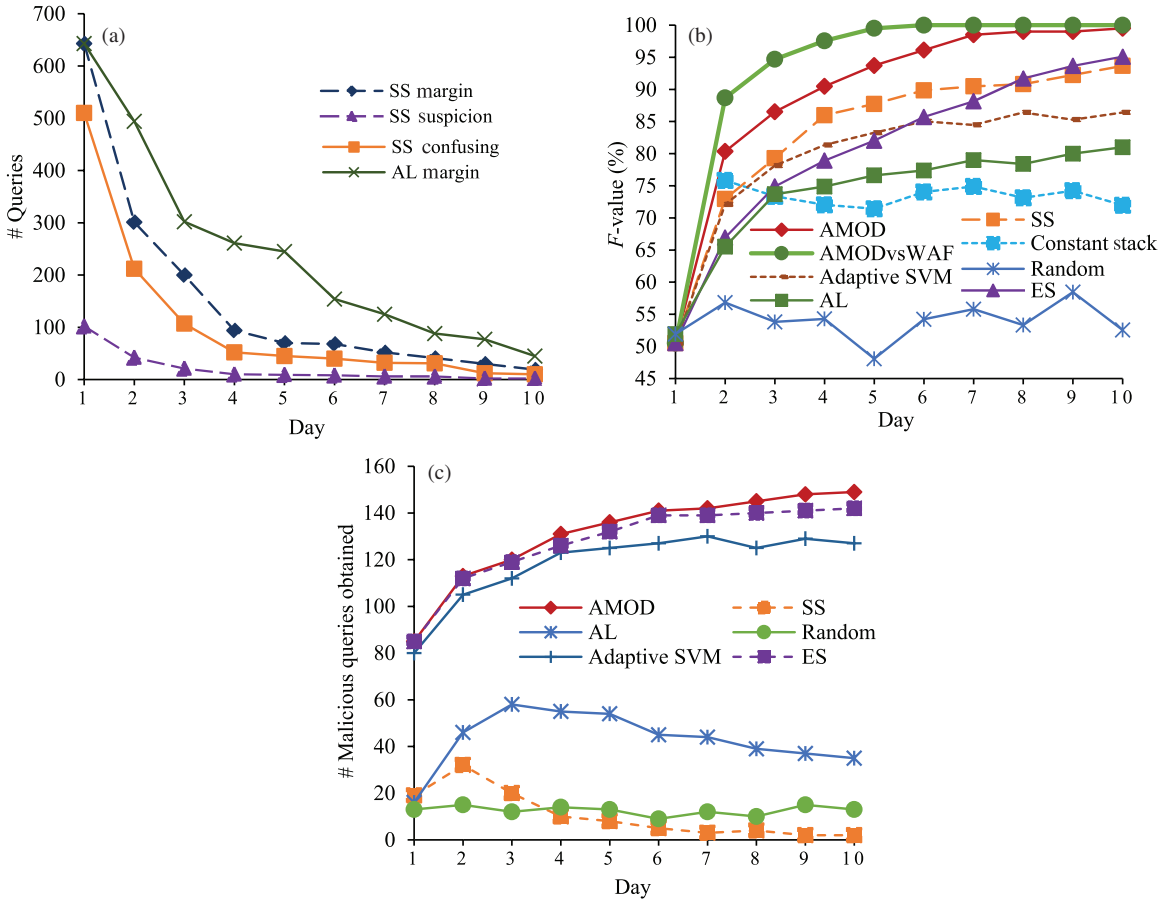

Figure 5 (Color online) (a) SS vs. SVM AL; (b) comparisons of F-values; (c) #Mal. queries obtained.

detection using the configuration obtained in Experiment 1, except that SS and SVM AL are required to obtain at most 150 important queries each day. As for detection model, except that adaptive SVM uses SVM, other approaches use stacking. The model in constant stacking approach is trained only once on the superset of the initial query set and the first-day unlabeled query set. The detection performance of the models on each day's unlabeled set (see step (3) in Subsection 3.1) is analyzed as follows.

Figure 5(a) depicts the process of SS in regard to the number of queries within the SVM margin (SS margin), within the confusing region (SS confusing), and obtained as suspicions (SS suspicion) separately. SVM AL is also shown about the number of queries within the margin (AL margin). They all present a decreasing trend. In comparison with SVM AL, important queries obtained by SS (SS suspicion) are constantly fewer than those obtained by SVM AL (the smaller one of AL margin and 150). However, queries within the margin in SS are always fewer than those in SVM AL, which implies that SS is more effective than SVM AL in reducing classifier uncertainty [50].

F-values of the seven detection approaches are exhibited in Figure 5(b). Among them, random selection shows the worst performance. Yet, for the five adaptive methods, a steep uptrend is shown for the first three days and then steady uptrend for the rest days. This is because at the beginning only a few queries are in the training pool, the decision boundary of the model is inaccurate. With more queries incorporated into the training pool, the detection performance is rapidly improved. The F-value trend of AMOD on datasets whose malicious queries are detectable by the WAF (AMODvsWAF) is always above all the others during the ten days, and stabilizes at 100% on the sixth day. This phenomenon shows that AMOD performs even better on malicious queries detectable by the WAF than on those undetectable by the WAF, with no false positives and false negatives after six days. Detection performance improvement of AMOD over the constant stacking approach suggests that AMOD effectively leverages knowledge gained during the adaptive detection process to enhance subsequent detection of queries. Moreover, AMOD

along with SS outperforms other adaptive methods. The improvement of AMOD over the adaptive SVM approach implies the stronger classification ability of stacking over single classifiers. The superiority of AMOD mainly lies in that it employs SVM HYBRID, which combines the contributions of SS and ES. Furthermore, SS outperforms ES, which confirms that suspicions contribute more to an accurate detection model. This is fair because the updated detection model would make a more accurate prediction of new queries, when the most important uncertain queries (suspicions) are incorporated into the training pool, rather than representative certain queries (exemplars). SS also outperforms SVM AL, even though SS obtains fewer important queries, as shown in Figure 5(a). This phenomenon reveals the advantage of SS over SVM AL: SS requires less manual labeling work and achieves better detection performance.

Figure 5(c) compares the number of truly malicious queries obtained by the six adaptive detection methods during the ten-day detection. Random selection still performs the worst. Generally, AMOD, ES and the adaptive SVM approach show an uptrend, while SS and SVM AL show a downtrend. This distinction is due to the difference in their preference for queries. AMOD, ES and the adaptive SVM approach prefer queries that are highly likely to be malicious, while SS and SVM AL prefer queries that the classifier is uncertain about, and thus might not be malicious; the reduction in uncertain queries leads to a decline in malicious queries obtained. AMOD, together with ES, outperforms other methods. The reason why ES falls short of AMOD lies in that SS benefits the detection performance of AMOD, which leads AMOD to obtain more correctly-classified exemplars. Moreover, the adaptive SVM approach lags behind AMOD and ES, which demonstrates the detection ability of the stacking-based detection model is stronger than single SVM's. The total number of malicious queries obtained by AMOD (1315) is 3.07 times and 10.44 times that by SVM AL (429) and random selection (126) respectively. This demonstrates the overwhelming ability of AMOD in harvesting malicious queries.

To sum up, Experiment 2 demonstrates the advantages of AMOD in two respects: improving detection performance and obtaining malicious queries. SS does well in the former respect, while ES excels in the latter. Both SS and ES defeat other methods except for AMOD in their advantageous respect.

5.3 Experiment 3: comparisons with related work

Experiment 3 aims at comparing AMOD with existing web attack detection methods, including linear combination [11], wavelet transform [13] and dimensionality reduction [14]. Existing web attack detection methods cannot adapt to the change of malicious behaviors, while our proposed adaptive detection method is capable of detecting the latest attacks by incrementally updating the detection model.

We use the publicly available HTTP dataset CSIC 2010 [26] for evaluation, whose traffic is targeted to an e-commerce web application. This dataset contains the four web attacks analyzed in this paper, with 36000 benign requests and 25000 malicious ones in total, making this dataset suitable for comparing malicious query detection systems. The dataset contains three subsets: normal traffic set for training, normal traffic set for test and malicious traffic set for test. Linear combination [11] uses normal traffic set for training to build the detection model, and uses the two test sets for the test phase. Its anomaly threshold is set to the default 10% larger than the maximum anomaly score seen during training. Wavelet transform [13] and dimensionality reduction [14] are unsupervised and thus do not require a training phase; they are performed on the two test sets. We set their anomaly threshold to 30.62%, the percentage of malicious queries in the dataset. TW2D-BA is used to perform wavelet transform [13], while PCA along with RP is used for dimensionality reduction [14]. For AMOD, we use 10% of the dataset as the initial set, and the rest as ten equal-size unlabeled sets. We also test these methods on the institute dataset, using the tenth-day set as the test set, and other ten sets as the training sets.

Table 5 compares these detection methods in terms of both detection performance and average runtime for 1000 web requests in test phase. As is shown, AMOD is the overall winner on both datasets, achieving the highest F-value (99.96%) and TP rate (99.95%), as well as the lowest FP rate (0.03%) among them on CSIC 2010 dataset. Among the three methods, linear combination [11] obtains the highest F-value and TP rate. We attribute the success of AMOD to the principle of SVM HYBRID, namely, choosing both the most important prone-to-be-misclassified queries (suspicions) and the most representative malicious

Table 5 Detection performance and runtime of web attack detection methods

Method	HTTP dataset CSIC 2010				Institute dataset			
	F-value (%)	TPR (%)	FPR (%)	Time (s)	F-value (%)	TPR (%)	FPR (%)	Time (s)
Linear combination [11]	97.12	95.57	1.23	16.2	91.18	93.00	0.01	26.7
Wavelet transform [13]	96.60	94.19	0.81	18.3	85.42	82.00	0.01	28.1
Dimension reduction [14]	96.73	94.13	0.49	7.9	88.89	84.00	0.005	12.2
Adaptive learning (AMOD)	**99.96**	**99.95**	**0.03**	**4.4**	**99.50**	**100.00**	**0.001**	**6.7**

queries (exemplars) from unlabeled queries. By incrementally incorporating correctly-labeled suspicions and exemplars into the training pool to update the detection model, its detection ability is progressively enhanced. AMOD also beats other methods in terms of test speed; it takes 4.4 and 6.7 s for AMOD to preprocess and classify 1000 web requests on average. Thus AMOD is applicable for real-time attack detection scenarios.

6 Discussion

In this paper, we propose an adaptive learning strategy called SVM HYBRID for efficient selection of important queries, and develop an adaptive system for malicious query detection. In this section, we discuss relevant issues of our approach.

Labeling the queries. We labeled the institute dataset to test the performance of AMOD. Manually labeling is time-consuming and error-prone. Therefore, we first filtered attacks using the rules of ModSecurity, which performs relatively better than other WAFs [51]. The rules of ModSecurity are provided by the OWASP ModSecurity core rule set (CRS) [52] project; the CRS version we used is 3.0.0. Then we manually confirmed the detected attacks, and concluded that the TP rate and FP rate of the detection results were 43.52% and 4.7%. To decipher the reason for the unsatisfactory detection results, we manually checked the ModSecurity rule set and found that some rules matched queries with simple malicious code keywords, which might appear in normal queries, resulting in false positives. Malicious queries that contain rare or complex malicious code snippets do not match any rules, leading to false negatives. This demonstrates the necessity of updating the WAF signature library, which can be achieved via our system.

Performance on the balanced dataset. It is rational to use a balanced dataset in most machine learning studies. However, since it is hard to collect a ten-day query set that is sufficiently large and balanced, we used random undersampling, which randomly removes benign queries from the original dataset, to create a decreased balanced dataset. The performance of AMOD on the decreased balanced dataset slightly falls behind with that on the original dataset before the seventh day, with lower F-value, lower TP rate and higher FP rate. This is reasonable, as removing a subset of benign queries may cause the classifier to miss important concepts related to the benign query class. Nevertheless, the performance on the decreased balanced dataset dramatically improves over time and catches up with the original dataset case on the seventh day. This confirms that our learning algorithm could handle data unbalance problem well by incorporating elaborately selected queries by SVM HYBRID to adjust classifier decision boundary.

Resistance to adversarial drift. Adaptive systems are intrinsically exposed to adversarial drift attacks, where attackers send a set of well-crafted requests which appear malicious; they do not intend to compromise the web server, but aim to mislead the classifier instead, so that real attacks might evade detection [53]. We confirm our system is resistant to adversarial drift. The reasons are as follows. (1) If the crafted queries appear similar to malicious queries in the training pool, the crafted ones would not be selected by our system since SVM HYBRID never select similar one to known malicious patterns but only selects representative unfamiliar ones. (2) If the crafted queries appear quite different from both benign and known malicious patterns, they pose great uncertainty to our classifier. Hence, the crafted one would be selected as a suspicion by SVM HYBRID. Then the selected crafted one might be labeled by the human experts and added to the training pool. The upcoming crafted queries that are the variants

of the selected crafted one would not be selected. (3) Even though the crafted query is selected as an exemplar by SVM HYBRID, its variants would not be selected again. Therefore, the adversarially crafted queries would not affect the detection ability of our classifier.

Comparisons with existing adaptive web attack detection methods. Both Fan [16] and Pinzón et al. [17] proposed adaptive web attack detection methods. Fan [16] analyzed request URL structure but ignored parameter values in queries, whereas we examined queries at character level to identify attacks in parameter values. Moreover, they classified normal behaviors by characteristics such as the number of visits and frequency. This is not reliable, since normal requests to web pages related to rarely used functions might be misclassified as malicious. We tackled this problem by only incorporating manually labeled queries selected by SVM HYBRID. Also, the model might be affected by adversarial drift, since attackers could send maliciously crafted traffic with the same characteristics of normal traffic; the crafted traffic might be misclassified as normal. In AIIDA-SQL [17], each SQL query with classification score between 0.35 and 0.6 was manually revised, leading to huge manual workload, while we proposed SVM HYBRID to select important queries for labeling and minimize manual work. Besides, AIIDA-SQL required domain knowledge to extract attack features, thus unknown attacks might evade detection, while AMOD uses N-grams for automatic feature modeling to identify unknown attacks.

7 Conclusion

This paper introduces a novel adaptive system for detecting malicious queries in web requests, called AMOD. The core of AMOD, named SVM HYBRID, an adaptive learning strategy, composes of SS and ES. Queries obtained by SVM HYBRID are incrementally incorporated into the training pool to update the detection model. Experiments show that AMOD outperforms existing web attack detection methods in terms of both the detection performance and the number of malicious queries obtained. For future research, we intend to develop explainable deep learning techniques to make the classifier explain the reasons behind its classification decisions for each incoming query, so that more previously unknown attack patterns would be discovered.

Acknowledgements This work was supported in part by National Key Reasearch and Development Program of China (Grant No. 2016YFB0800703), in part by National Natural Science Foundation of China (Grant Nos. 61272481, 61572460), in part by Open Project Program of the State Key Laboratory of Information Security (Grant Nos. 2017-ZD-01, 2016-MS-02), and in part by National Information Security Special Project of the National Development and Reform Commission of China (Grant No. (2012)1424). We would also like to thank Dr. Xinyu Xing in Pennsylvania State University for his help with this work.

References

1 Symantec. Internet security threat report. 2016. https://www.symantec.com/security-center/threat-report
2 Fonseca J, Vieira M, Madeira H. Evaluation of web security mechanisms using vulnerability & attack injection. IEEE Trans Depend Secure Comput, 2014, 11: 440–453
3 Imperva. Web application attack report. 2015. https://www.imperva.com/docs/HII_Web_Application_Attack_Report_Ed6.pdf
4 WhiteHat. Web application security statistic report. 2016. https://info.whitehatsec.com/rs/675-YBI-674/images/WH-2016-Stats-Report-FINAL.pdf
5 Lawal M, Sultan A B M, Shakiru A O. Systematic literature review on SQL injection attack. Int J Soft Comput, 2016, 11: 26–35
6 Symantec. Team ghostshell hacking group back with a bang. 2015. https://www.symantec.com/connect/blogs/team-ghostshell-hacking-group-back-bang
7 Aleroud A, Zhou L. Phishing environments, techniques, and countermeasures: a survey. Comput Secur, 2017, 68: 160–196
8 Fang Z J, Liu Q X, Zhang Y Q, et al. A static technique for detecting input validation vulnerabilities in Android apps. Sci China Inf Sci, 2017, 60: 052111
9 Prokhorenko V, Choo K K R, Ashman H. Web application protection techniques: a taxonomy. J Netw Comput Appl, 2016, 60: 95–112

10 Krugel C, Vigna G, Robertson W. A multi-model approach to the detection of web-based attacks. Comput Netw, 2005, 48: 717–738
11 Robertson W K, Vigna G, Kruegel C, et al. Using generalization and characterization techniques in the anomaly-based detection of web attacks. In: Proceedings of the 13th Annual Network and Distributed System Security Symposium (NDSS'06), San Diego, 2006
12 Song Y, Keromytis A D, Stolfo S J. Spectrogram: a mixture-of-markov-chains model for anomaly detection in web traffic. In: Proceedings of the 16th Annual Network and Distributed System Security Symposium (NDSS'09), San Diego, 2009. 121–135
13 Kozakevicius A, Cappo C, Mozzaquatro B A, et al. URL query string anomaly sensor designed with the bidimensional haar wavelet transform. Int J Inf Secur, 2015, 14: 561–581
14 Juvonen A, Sipola T, Inen T. Online anomaly detection using dimensionality reduction techniques for http log analysis. Comput Netw, 2015, 91: 46–56
15 Xie Y, Tang S, Huang X, et al. Detecting latent attack behavior from aggregated web traffic. Comput Commun, 2013, 36: 895–907
16 Fan W K G. An adaptive anomaly detection of web-based attacks. In: Proceedings of the 7th International Conference on Computer Science & Education (ICCSE'12), Melbourne, 2012. 690–694
17 Pinzón C, De Paz J F, Bajo J, et al. AIIDA-SQL: an adaptive intelligent intrusion detector agent for detecting SQL injection attacks. In: Proceedings of the 10th International Conference on Hybrid Intelligent Systems (HIS'10), Atlanta, 2010. 73–78
18 Meng Y, Kwok L F. Adaptive blacklist-based packet filter with a statistic-based approach in network intrusion detection. J Netw Comput Appl, 2014, 39: 83–92
19 Wang W, Guyet T, Quiniou R, et al. Autonomic intrusion detection: adaptively detecting anomalies over unlabeled audit data streams in computer networks. Knowledge-Based Syst, 2014, 70: 103–117
20 Zhang J, Li H Z, Gao Q G, et al. Detecting anomalies from big network traffic data using an adaptive detection approach. Inf Sci, 2015, 318: 91–110
21 AlEroud A, Karabatis G. Queryable semantics to detect cyber-attacks: a flow-based detection approach. IEEE Trans Syst Man Cybern Syst, 2016. doi: 10.1109/TSMC.2016.2600405
22 Aleroud A, Karabatis G. Contextual information fusion for intrusion detection: a survey and taxonomy. Knowl Inf Syst, 2017, 52: 563–619
23 Sousa A F, Prudencio R B, Ludermir T B, et al. Active learning and data manipulation techniques for generating training examples in meta-learning. Neurocomput, 2016, 194: 45–55
24 Rossi A L D, de Leon Ferreira A C P, Soares C, et al. MetaStream: a meta-learning based method for periodic algorithm selection in time-changing data. Neurocomput, 2014, 127: 52–64
25 Folino G, Sabatino P. Ensemble based collaborative and distributed intrusion detection systems: a survey. J Netw Comput Appl, 2016, 66: 1–16
26 The HTTP dataset CSIC 2010. http://www.isi.csic.es/dataset/
27 Zheng Y H, Zhang X Y. Path sensitive static analysis of web applications for remote code execution vulnerability detection. In: Proceedings of the 35th International Conference on Software Engineering (ICSE'13), San Francisco, 2013. 652–661
28 Jamdagni A, Tan Z Y, He X J, et al. RePIDS: a multi-tier real-time payload-based intrusion detection system. Comput Netw, 2013, 57: 811–824
29 Garcia-Teodoro P, Diaz-Verdejo J E, Tapiador J E, et al. Automatic generation of HTTP intrusion signatures by selective identification of anomalies. Comput Secur, 2015, 55: 159–174
30 Zhong Y, Asakura H, Takakura H, et al. Detecting malicious inputs of web application parameters using character class sequences. In: Proceedings of the 39th Annual Computer Software and Applications Conference (COMPSAC'15), Taichung, 2015. 525–532
31 Ariu D, Tronci R, Giacinto G. Hmmpayl: an intrusion detection system based on hidden markov models. Comput Secur, 2011, 30: 221–241
32 Wang K, Stolfo S J. Anomalous payload-based network intrusion detection. In: Proceedings of the 7th International Symposium on Recent Advances in Intrusion Detection (RAID'04), Sophia Antipolis, 2004. 203–222
33 Wang K, Parekh J J, Stolfo S J. Anagram: a Content Anomaly Detector Resistant to Mimicry Attack. Berlin: Springer, 2006
34 Oza A, Ross K, Low R M, et al. HTTP attack detection using n-gram analysis. Comput Secur, 2014, 45: 242–254
35 Perdisci R, Ariu D, Fogla P, et al. McPAD: a multiple classifier system for accurate payload-based anomaly detection. Comput Netw, 2009, 53: 864–881
36 Swarnkar M, Hubballi N. OCPAD: one class naive bayes classifier for payload based anomaly detection. Expert Syst Appl, 2016, 64: 330–339
37 Duessel P, Gehl C, Flegel U, et al. Detecting zero-day attacks using context-aware anomaly detection at the application-layer. Int J Inf Secur, 2016, 16: 475–490
38 Vapnik V, Kotz S. Estimation of Dependences Based on Empirical Data. New York: Springer-Verlag, 2006
39 Guo H S, Wang W J. An active learning-based SVM multi-class classification model. Pattern Recogn, 2015, 4: 1577–1597
40 Kremer J, Steenstrup P K, Igel C. Active learning with support vector machines. Data Min Knowl Disc, 2014, 4: 313–326

41 Gao F, Lv W C, Zhang Y T, et al. A novel semisupervised support vector machine classifier based on active learning and context information. Multidim Syst Signal Process, 2016, 27: 969–988
42 Wang M, Min F, Zhang Z H, et al. Active learning through density clustering. Expert Syst Appl, 2017, 85: 305–317
43 Aghaee A, Ghadiri M, Baghshah M S, et al. Active distance-based clustering using K-medoids. In: Proceedings of the 20th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'16), Auckland, 2016. 253–264
44 Baram Y, Ran E Y, Luz K. Online choice of active learning algorithms. J Mach Learn Res, 2012, 5: 255–291
45 Wolpert D H. Stacked generalization. Neural Netw, 1992, 5: 241–259
46 Hillstone Networks. Hillstone e-series next-generation firewalls. http://www.hillstonenet.com/our-products/next-gen-firewalls-e-series/
47 Fielding R, Gettys J, Mogul J, et al. RFC 2616: hypertext transfer protocol-HTTP/1.1. Comput Sci Commun Dict, 1999, 7: 3969–3973
48 Ambusaidi M A, He X J, Nanda P, et al. Building an intrusion detection system using a filter-based feature selection algorithm. IEEE Trans Comput, 2016, 65: 2986–2998
49 Ben-Hur A, Weston J. A user's guide to support vector machines. In: Data Mining Techniques for the Life Sciences. Berlin: Springer, 2010. 223–239
50 Xiong C, Johnson D M, Corso J J. Active clustering with model-based uncertainty reduction. IEEE Trans Pattern Anal Mach Intell, 2017, 39: 5–17
51 Prandl S, Lazarescu M, Pham D S. A study of web application firewall solutions. In: Proceedings of the 11th International Conference on Information Systems Security (ICISS'15), Kolkata, 2015. 501–510
52 Trustwave. Modsecurity core rule set. 2016. https://www.owasp.org/index.php/Category:OWASP_ModSecurity_Core_Rule_Set_Project
53 Kantchelian A, Afroz S, Huang L, et al. Approaches to adversarial drift. In: Proceedings of the 6th ACM Workshop on Artificial Intelligence and Security (AISec'13), Berlin, 2013. 99–110