Detection, Prevention and Defense Approaches in Web Applications

Yunfeng Lu
40118242
bentleylu@outlook.com

Osezele Ehi-Douglas
40121803
osezelec@gmail.com

Yunus Emre Aydar
40110411
yunus.aydar314@gmail.com

Abstract
This survey aims to figure out which kind of detection, protection, defense and mechanism methods can be observed during cyber security attacks in web applications. Methodologies, categorizations, challenges, mechanisms, comparisons, solutions, evaluations, models and architectures will be analyzed to understand and define better defenses against web application attacks. Detection models like mapping-based models, machine learning approaches, categorizations like etiological and symptomatic approaches, various detection and defense tools and framework will be mentioned during classifications and comparisons. There will be comparison tables for protection, defense and mechanism parts will be shown and explained shortly. Also, some factors that are derived from comparison tables will be discussed during the paper.

Keywords*: SQL injection, Cross-site scripting, Defense in web application, Cyber attacks in web applications, Intrusion detection system, Machine learning, Attack prevention, Defensive tools and frameworks and mechanisms for web applications.*

I. INTRODUCTION

With the boosted use of web applications, modern life becomes more convenient than ever. However, not only the advanced technologies were raised to a new level, but the cyber concerns were also stealthier and more harmful than before. Especially new variants derived from well-known attacks such as Cross-site scripting, SQL injection and etc. Although several traditional mechanisms are applied to detect, prevent and defeat, it is still impossible to eliminate mutated attacks in a satisfying way. Therefore, to achieve the most feasible solutions to make sure web applications work perfectly, new countermeasures have been invented to avoid the new variants causing unaffordable damages.

The paper is organized in three parts. They are detection, prevention, and defense sections, respectively. In the detection part, two typical methods against SQLi and XSS will be demonstrated, mapping-based model detection and machine learning involved-intrusion detection system. In mapping-based model detection, requests will be divided into two scenarios based on the types of web application working mechanisms. Requests will be matched in each case to enhance the ability of preventing possible attacks. In machine learning involved-IDS, new genetic algorithms were applied to evolve the signature database with self-generated patterns. Additionally, a cross entropy algorithm was also used to determine whether the incoming request is benign or malicious based on the provided probabilities. In the prevention mechanisms section, we explain that the rising level of sophistication of attacks and attackers' techniques means that more effort is being put into researching new methods for detecting and preventing attacks. We then introduce some of these mechanisms, focusing on Hybrid Taint Inference for the detection and prevention of SQLI attacks. In the defense tools and frameworks part, 3 categorizations (Etiological, Symptomatic, and Hybrid) which are defence strategies against web application attacks, were classified for major tools and frameworks that the paper [10] was used to improve the part. The methods that are in all categorizations were guided to choose the best possible defense system in each category. Meticulously chosen the tools and frameworks were analyzed and adapted in the uniquely designed comparison tables. In the last sections of the part, conclusions, evaluations, and factors were discussed in accordance with categories and comparison table analysis.

II. DETECTION MECHANISMS AND TOOLS

A. Representative 1: Mapping-based model

Author proposed a mapping-based intrusion detection system to detect and prevent the SQL and XSS attack. They categorized the web application into two types, static and dynamic.

Static web application (SWA)

When the request is static, it will be forwarded to the SWA mapping model. Then inspect whether the particular query is called by corresponding request. If it is true, execute the query to retrieve it from the database. If it is not, drop the request and block the user.[1]

Dynamic web application (DWA)

When the request is dynamic, firstly, the request will be compared with approved requests in the past. If it is a match, the request will be processed. If it is not, the further validation will be starting. Query sets will be created with peeled input data from request. Then use the new generated query set to match the legitimate query set. The process will be executed as long as the peeled set has proved valid and determined safe to run. During the procedure of removing input data, it ensures the malicious commands will not be executed. If the set is still infeasible to find a match, an alarm will be triggered to warn the harmful request and block the client.

Other methods and comparisons

The above mapping-based model is representative of non-machine learning based methods. There are two extra approaches used to prevent SQLI and XSS attacks.

AMNESIA+JCrasher adopted in [2] where JCrasher is an automatic tool that generate test cases for same feature with unique input values and Enhanced AMNESIA has been used to generate SQL query model and each model will be attached with a different ID which is used to identify the sources of the queries. During the inspection period, comparison will be made to check whether it is in accordance with the model allocated with the same ID. If the result is true, the query will be executed. If it is not, it will be aborted and marked as malicious activities.

In [3], Author's developed IDS is a mutation integrated in the tripwire, which is used to gather regular expressions to identify all the executable parts in the client web pages would possibly be under attack as a victim. The default regular expressions are dedicated to prevent potential XSS vectors, for instance, <script> and <img> commands. Users could build any pattern based on their needs of protection requirements. Thus, any not executable contents with low security concerns that need refreshing often, are not recommended to include into the pattern in order to minimize the occurrence of false positives. Since any tiny harmless modification will trigger huge changes in hash. With the generated web page baseline, server administrators could routinely check any differences between the current hash and old baseline to identify any malicious tampering. Furthermore, it is important to update the baseline in time to ensure no false positives would hit the alarm.

A comparison table could be made after all the features and shortcomings are listed below.

Papers	SQL injection	Cross-site scripting	Features	Shortcomings
AMNESIA+JCrasher[2]	√		No false positive; Low false positive (based on the assumption)	Requires assumptions that queries are created by manipulating string in the application
Mapping-based model[1]	√	√	Low false positive; using container to mitigate overhead;	Depending on authenticated queries; increased processing time
XSSmon[3]		√	Low false positive(1 out of 27)	Need to update baseline timely; not all parts are hashed into baseline

Table I: Features and Shortcomings of Non-Machine Learning based methods.

B. Representative 2: Evolutionary algorithm integrated into IDS.

In [4] author proposed two approaches to countermeasure the SQL injection (SQLI), Cross-site Scripting (XSS) and Remote File Inclusion (RFI) by customizing the anomaly-based IDS with cross-entropy and the signature-based IDS with genetic algorithm.

Anomaly-based IDS with cross-entropy is adopting a series of algorithms to calculate the cross-entropy between the new and the stored requests. If the result surpasses the threshold, an alarm will be raised. The benign stored request will be collected and stored during the training process in which the testing requests are sanitized without any malicious content.

Signature-based IDS with genetic algorithms are developed to enhance the performance of the signature to independently recognize new signatures derived from already existing signatures. The essence is converting log files into binary string representation and applying genetic algorithms to generate new signatures of the unseen

attacks. In a word, new variants or unrevealed mutations will be highly possible to be detected by the upgraded IDS. Similarly, the initial signatures are fed from training logs of intrusion in the training phase.
A hybrid IDS of both methods would be a good solution to monitor and tackle the web application threats.

Other methods and comparisons
The above evolutionary algorithm is representative of machine learning based methods. There are two extra approaches used to prevent SQLi attacks.
Adaptive deep forest in [5] is adopted which aims to detect unknown types of SQLI. The data were collected thru honeypots to train the deep forest algorithm. Multi-grained scanning was invented to improve the feature selection process by using 100,200,300 sizes of sliding windows. And to solve the degradation problems, fine-tuning the framework of the algorithm by feeding input of each layer by remerging the raw features with previous outputs. Furthermore, AdaBoost was used to distribute the weights based on the classification results.
The CNN-based approach was developed with three padding convolutional layers, three max pooling layers and one full connectivity layer and one hidden layer[28]. 3X3,4X4 and 5X5 convolutional layers were used in the algorithm followed by a 2X2 max pooling layer. The training data were collected from famous dataset like KDD99, UNSW-NB15 and etc. And SQLI scripts captured from the real world were used for validation purposes. The results were outstanding with 99.5% of accuracy and 100% of recall. This is illustrated in table II below.

Papers	SQL injection	Cross-site scripting	Features	Shortcomings
Evolutionary algorithm[4]	√	√	Genetic algorithm; Cross-entropy; self-generation signature;	Algorithm selection; training and test accuracy;
Adaptive deep forest[5]	√		Adaptive deep forest algorithm; enhanced feature scanning; concatenated raw features; need less samples; low computational cost;	Accuracy is not higher than deep neural network; degradation due to the increased number of layers
CNN-based approach[6]	√		Convolutional neural network algorithm; avoid overfitting;	High accuracy (99.5%); high recall (100%)

Table II. Machine Learning techniques involved in IDS

C. Comparisons & Evaluations
Based on the two different categorizations, each of them has their own merits. In terms of the computational aspect, machine learning requires more calculations, predictions and hypotheses during the training process. While it is deployed, it is depending on analyzing the parameters to predict the genuine intention of the request which is less need of computational energy. However, mapping-based models or other methods mentioned above may consume more time and resources on matching previously authenticated ones or issuing hotspot tags or hashing the baselines. And at some points, they are relatively weak to confront the new variants of malicious requests compared to machine learning based systems. But in some known cases they have shown that they are more stable and dependable than their machine learning counterparts, since they are still suffering algorithm selection, algorithm augments adjusting, training and test data options and accuracy validations.
Noticeably, algorithm selection is still a new challenge for developers. Like, k-means, fuzzy, c-mean, and other association rules and clustering methods are not suitable to be used in detecting such types of attack [7]. Some classification algorithms also need more analyses to determine the best performance. Like the adaptive deep forest case, new created enhanced algorithm failed to give a better output than deep neural network when the training samples are huge [28]. Based on the pros and cons and our perspectives of the development of the machine learning technologies, machine learning based approaches are more promising in future as long as the gaps lying in algorithm selections, accuracy validations and stability have been filled. Thus, non machine learning technology looks more practicable nowadays.

Class	Pros	Cons
Not machine-learning based methods	lower false positive; overhead mitigation; more stable than machine learning methods;	Previously authenticated queries required; baseline required; needs more time and energy to distinguish the requests; relatively weak to the new variants; expert-required when facing new unknown requests
Machine-learning based IDS	Mutation self-generation; lower false positive; lower computation requirements during execution; expert-less when predicting new unknown requests	Algorithm selection; training and test accuracy; needs more calculation and predictions; huge training data; overfitting; underfitting

Table III. Comparison of ML and non ML-based methods

III. WEB APPLICATION PROTECTION MECHANISMS

As web applications play an increasing role in our lives today and thus process and store information about individuals including PII, it is vital that mechanisms be but in place to ensure for their secure operation. Traditional mechanisms such as anomaly, signature and behavior-based detection have worked well in the past but due to the increasing complexity of web applications as well as the increasing level of sophistication of web application attacks, more recent techniques are needed to properly protect applications [13]. In this section we report our findings from examining six papers about state of the art with regards to these recent web application protection mechanisms. We introduce some mechanisms that have been researched or currently being researched, and finally discuss in detail one of them (Hybrid Taint Interference for defeating SQLI attacks). Table IV below summarizes the mechanisms in the papers we surveyed. There are various mechanisms that have been and are still being researched with regards to web application security. This is because with the ubiquity of web applications today, there is much more reward for malicious attacks and thus we are witnessing an upturn with regards to the level of sophistication of attacks and attackers. Two very common attacks today are Code injection attacks such as Cross-site Scripting (XSS) and SQL injection attacks (SQLI). These attacks consistently rank at the top of lists like the OWASP Top 10 Vulnerabilities.

Protection Mechanism	**Summary**	**Deployment**	**Detected Attacks**	**Performance**
Survey [13]	Traditional and recent approaches to detect, prevent and remove web application attacks.	Client/Server Side	XSS injection	
Moving Target Defense technology [16]	Changing system configuration to reduce opportunities and increase cost for attacker.	Client Side	XSS injection	Little impact on system performance Only 10ms overhead
Automata-based scheme [17]	Server-side ingress filter that aims to protect vulnerable browsers	Server side	Content-Sniffing XSS Attack	Zero false positives
Hybrid Taint Interference [18]	Exploits the complementary nature of negative and positive taint inference to mitigate their respective weaknesses.	Server side	SQL Injection Attacks	No false positives Low performance overhead (4%)
Moving Target Defense technology [19]	Take the first steps of applying MTD concepts to web applications to create effective defensive layers	Server side	Code injection and SQL injection attacks	Less than 4% overhead on average
Machine Learning Predictive Analytics [20]	(ML) predictive analytics provides a functional and scalable approach to data mining to big data in detection and prevention of SQLIA	Server side	SQL Injection Attacks	Accuracy: 0.986 precision: 0.974 Recall: 0.997 F1 Score: 0.985

Table IV: Summary of Papers on Web Application Attack Prevention Mechanisms.

Proposed solutions for them, although effective, rely on developer awareness and the implementation of secure coding practices like prepared statements and inputs sanitizing/validation. These solutions are also often ignored or improperly implemented. For example, in web frameworks such as WordPress, which actively encourage their user/developer community to add new functionality to the framework via plugins, the quality of these plugins varies widely. It is thus paramount to research and implement mechanisms that can automatedly function securely and independent on user or developer input.

Moving Target Defense technology (MTD) is a novel technique that works by frequently switching up the configuration of a system in an attempt to keep the status of the system unknown to an attacker. This is done while still maintaining functionality of the underlying system and it can be implemented in different areas such as in browsers at the client layer to prevent XSS attacks like in [16] or at the server layer to protect SQLI attacks as in [19]. It is an effective technique, has very low impact on system performance and produces very low overhead (<10ms). In [17] the researchers introduce an automata-based scheme for protecting against Content-Sniffing XSS attacks. This mechanism is implemented as a server-side filter that examines files uploaded by users against a set of potentially dangerous elements before they are allowed to pass on to the server. It is a less intrusive mechanism than current signature-based approaches as it works by observing HTML tags of packets in the headers rather than the data contained in the packet. It is also very accurate with no false positives and exhibits an insignificant overhead of 60ms when scanning files of 2KB. Machine Learning (ML) predictive analysis [20] is another technique that provides a functional and scalable mechanism for SQLI prevention with respect to big data and data mining. It uses a trained classifier which is deployed as a web service and attempts to intercept and predict SQLIA in web requests and preventing them from reaching the backend database. It also shows promising performance with an accuracy of 98.6%, precision of 97.4%, recall of 99.7% and F1 score of 98.5%.

Taint tracking is a technique that involves tracking the flow of data during the execution of a program. This continuous tracking of data has made the technique to be rarely used in practice as it suffers from deployment issues such as high performance overhead and intrusive instrumentation [18]. Taint Inference techniques can address the shortcomings of taint tracking as they eliminate the need to track data flow by instead inferring markings based on either the inputs to the program as in the case of Negative Taint Inference (NTI), or on the program itself as in the case of Positive Taint Inference (PTI). Both these techniques do show promise in SQLI attacks, however …… show that attackers can circumvent them when used alone by more sophisticated techniques that are tailored to bypass the weaknesses of NTI and PTI. They then propose a system (Joza) that utilizes a novel hybrid approach -exploiting the complementary nature of PTI and NTI- to mitigate these weaknesses. It works by firstly carrying out PTI analysis on the query and then NTI analysis, and only the queries that are deemed safe by both components can reach the DBMS. When a query is flagged dangerous by either component, Joza allows for two options; Error Virtualization in which an error code is returned to be application logic to be handled as a failed query, or Termination in which the application is forced to exit.

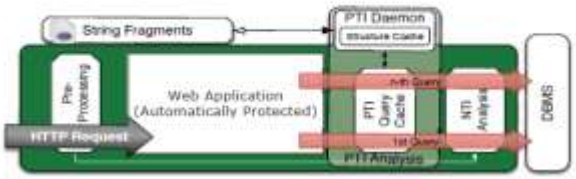

Fig I. Architecture of the Joza Plugin.

To evaluate the performance of the Joza system they implemented it as a plugin for the popular content management system Wordpress. The Wordpress plugin manager is then configured to intercept all requests to the DBMS being protected and first send them through the Joza plugin. They then created a testbed of 53 exploits, including exploits specifically adapted to bypass NTI or PTI when used alone. The results of their evaluation show that the performance of the plugin is dependent on the relative frequency of read and write requests. Write requests are seen to cause more overhead than read requests -A workload consisting of 10% write requests and 90% read requests will incur more overhead than one with 5% writes and 95% reads-. As Wordpress sites typically have request workloads that are made up of less than 1% of write requests, Joza exhibits an average overhead of less than 4%. The hybrid taint inference technique is also found to detect all attacks in the testbed, including those adapted to evade PTI and NTI successfully with no false positives. The imprecision of the taint inferencing process however did not allow for an accurate measurement of the false negative rate.

Writes	Reads	Plain Time	Protected Time	Overhead
50%	50%	0.2744	0.2990	8.96%
10%	90%	0.2284	0.2402	5.16%
5%	95%	0.2227	0.2328	4.53%
1%	99%	0.2181	0.2269	4.03%

Fig II: Overhead of Joza for different compositions of write and requests in a query.

Their experiment shows that Hybrid Taint Inference is extremely effective in preventing real-world SQL injection attacks, exhibits no false positives, and incurs a low performance overhead. It however is susceptible to false negatives due to the inherent imprecision of the inference process. HTI can also theoretically be circumvented using attacks that evade both PTI and NTI but such attacks would be very difficult to implement in practice.

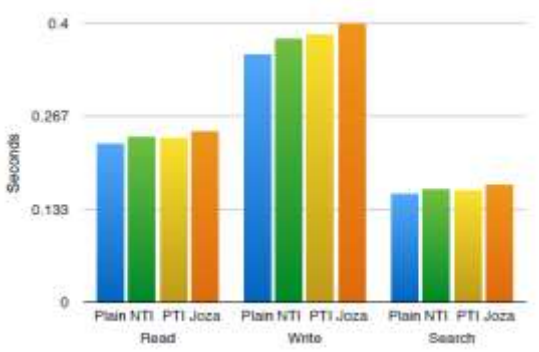

Fig III: Comparison of total overhead for Joza with that for plain queries, NTI and PTI

They conclude that as promising as Joza's overall performance is, it would be further improved with the use of content-caching engines that are often used by heavily trafficked websites. This way, only the first request to a URL would result in the execution of database queries to serve up the requested page. And subsequent requests would only require the retrieving of a static cached copy. This will greatly reduce the demand on Joza.

To summarize, there are different mechanisms that can be implemented for the prevention of web application attacks. These mechanisms are the major underlying architecture with which various protection tools and frameworks are built. The choice of mechanisms to be used in a protection tool depends on the type of attacks to be detected and protected against and the deployment location of the tool (client/server side). Techniques like MTD are also very effective when implemented as part of a Defense in Depth strategy as seen in [19].

IV. DEFENCE TOOLS AND FRAMEWORKS

According to Defence Tools and Frameworks, the papers of [8], [9], [10], [11], [12], [14], [15] guided our survey for learning defence tools and frameworks and analysing them. The paper of Mitropoulos et al. [10] led our survey forming unique tables for the tools and the frameworks by searching features of the tools and the frameworks in detail. The taxonomy of web application defenses against injection attacks which is mentioned in the paper [10] was used to develop our analysis for the tools and frameworks. According to the taxonomy that led our survey to understand the tools and frameworks, approaches for defense are separated into 3 categories which are Etiological approach, Symptomatic approach, and Hybrid approach. Etiological approach is separated into Parse-Tree Validation, Policy Enforcement, and Instruction Set Randomization (ISR). Symptomatic approach is separated into Taint Tracking and Training. Hybrid approach just utilizes both an Etiological and Symptomatic approach. All defense approaches are illustrated in the table below.

Fig. IV. Suggested categorizations for defence tools and frameworks in accordance with web application attacks [10]

The concept of taxonomy will be mentioned shortly as a hitherto representation of all the tools and the frameworks. The aim of the tools and the frameworks section will be about efficiency of using attack prevention tools and frameworks, finding best tools and frameworks, conclusion of evaluation all categories for defenses against web application attacks, and understanding major factors which are Client-Aware Usage/User-Friendliness, Cost, Performance, User Experience, and Adaptability that affect on usage of the tools and the frameworks. Those factors will be discussed in comparison part of the tools and the frameworks. Also, the tools and the frameworks for the paper [10] mostly were chosen from

Etiological and Hybrid categories. The tools and the frameworks under Symptomatic category were not practicable compared to other categories that any defence system was not picked from Symptomatic during forming tables and synthesizes. The category was only analyzed for general ideas and will be briefly mentioned for best tools, parts and frameworks. All other tools and frameworks that taken from the papers [8], [9], [11], [12], [14] were analyzed differently rather than classifying them in the categories[10].

Analysis Categorizations for Tools and Frameworks

A. Etiological Category

Parse-Tree Validation

Parse- Tree Validation method is used to compare the tree representation of the theoretical syntactic structure of the code that is approximately to be executed with the one that was initially aiming. In case the trees diverge, the application is likely under threat of attack. Parse-tree approval is a compelling approach for the discovery of Domain-specific language code injection attacks. SQLcheck can be proposed as a great system for Parse-Tree method. A drawback of frameworks of SQLcheck can be the application must be adjusted in each code part that sends an SQL inquiry to the database for execution [10].

Policy Enforcement

Policy Enforcement is utilized to avoid Cross Site Scripting and Cross-Site Request Forgery attacks. When employing a system that actualizes approach authorization, developers must characterize particular security policies on the server-side. Policies may be indicated through JavaScript extension, pattern matching, or syntax-specific settings. The policies can be implemented either within the user's browser at runtime, or on a server-side proxy that communicates with server responses. Noxes can be great tools in Policy Enforcement. Noxes limits clients for permission to define policies for the evasion of Cross Site Scripting attacks. The key thought behind Noxes is to parse the HTML response that comes to the browser and discover static URL references. At that point, rely on a set of approaches, Noxes permits or forbids any generated requests[10].

c) Instruction Set Randomization

Instruction Set Randomization is a method that can be utilized against various sorts of application attacks and was initially used for the prevention of binary code injection attacks. The best thought behind ISR is to alter the representation of code based on a chosen transformation at random and randomization of the execution environment. Thus, any malicious code injected as part of untrusted input data that is caused by the attacker. Then, the attacker is not able to calculate a randomized algorithm. In the end, malicious code will not be successful during execution. ISR is a viable method to avoid SQL injection attacks. SQLrand system is the one of best framework for Instruction Set Randomization that applies the concept of ISR for the prevention of SQL injection attacks. It permits software engineers to make SQL articulations utilizing randomized instructions rather than standard keywords. The changed queries are remade at runtime by utilizing the same key for randomization that causes interruption for malicious users[10].

B. Symptomatic Category

Taint Tracking

Taint Tracking method marks untrust data, another way to say tainted data that follows its engendering hitherto the program. Taint Tracking might be a feasible method against SQL injection and Buffer Overflow Attack. On the off chance that the variable is utilized in an expression that sets another variable, that variable is additionally checked as untrusted. On the off chance that any of these factors is utilized in a possibly unsafe operation may behave in like manner. Taint tracking can be facilitated and be seen in use of Perl and Ruby. CSSE(Context-Sensitive String Evaluation) framework can be chosen as a good defence in Taint Tracking method that helps associating tainted data with specific metadata easily . When tainted data comes to a sink, CSSE performs syntactic checks in accordance with metadata[10].

Training

Training method is a method that formed as the original intrusion detection framework. When training mechanism has been utilized, training mechanism helps learning all valid legitimate code statements during a training phase. Generally, Training methods detect Domain-specific language -driven injection attacks and generate and store valid code statements in several forms. Also, the method recognizes attacks in terms of the set of valid code statements. Amnesia tool can be the best for Training method that helps identify SQL injection attacks. To identify the attacks, a query model is used with the location of every SQL statement within the application. At the end, when SQL statements separate from the expected model, Amnesia monitors execution of application to distinguish them [10].

C. Hybrid Category

Hybrid approach includes mechanisms that borrow characteristics from both etiological and symptomatic approaches. Hybrid approach generally helps getting detection of Cross Site Scripting attacks and Domain-specific language code injection attacks with mixture approaches. XSS-GUARD can be the best framework for Hybrid category that XSS-GUARD is a training scheme method from etiological category that employs parse tree validation method from symptomatic category. During training, the scheme maps legitimate scripts to HTTP responses. XSS-GUARD recovers for each script contained in a response its parsed tree. Also, XSS-GUARD checks in case it is one of those already mapped to this response. When comparison of the parsed trees are excluded, XSS-GUARD verifies for a correct match of lexical entities [10].

	Attacks Analysis						
Tools	Cross-Site Scripting (XSS)	SQL Injections (SQLi)	Buffer Overflow	Remote Command Execution	Denial of Service (DDoS)	Performance	Tested Environment
SPAAS [14]	IP	I	I	I	I	Moderate	OpenEMR (open- source web application)
JSFlow [10][15]	P	I	I	I	I	Low	Browser extension(Firefox) and Snowfox(server side)
Noxes [10]	P	I	I	I	I	Unknown	Windows .NET application in C#
Project Shield [9]	I	I	I	I	P	Unknown	Google Server
OpenDaylight [9]	I	I	I	I	P	Unknown	Linux Server, OpenFlow, and Miniset
HAProxy[9][12]	I	I	I	I	P	High	Linux Server
ModSecurity (WAF) [8][11][12]	P	P	I	IP	P	High	Apache HTTP Server
CAPTCHA [9]	I	I	I	I	P	Unknown	Websites

Table V. Comparison Table for Defence Tools

IP (In Progress) is indicated under Attack Analysis category that the mentioned prevention system is partially usable that requires more testing in accordance with the papers(which are cited in the paper [10] and the other papers [8], [9], [11], [12], [14], [15] that show tools behaviour one by one. The paper[10] ignored the possibility of prevention against attacks which were not theoretically proposed in the paper[10].) P(Practicable) is indicated under Attack Analysis that the mentioned prevention system is usable against specific attack. I (Impracticable) is indicated under Attack Analysis that the mentioned prevention system is not usable against the specific attack. Evaluation of Performance category in accordance with the mentioned defence system that High represents satisfying performance of the system, Moderate represents the average performance of the system, Low represents unsatisfying performance of the system, and Unknown represents no information about performance of the system.

Tools	Usability Analysis	Descriptions		
SPAAS[14]	Advantages	Advanced in the security readiness of a software system.		
	Disadvantages	Usable in only PHP based web applications		
JSFlow[10][15]	Advantages	Possibility of deploying in browser extension,proxy,suffix proxy,security library, and server side		
	Disadvantages	Limitations of dynamic information-flow tracking		
Noxes[10]	Advantages	Protection against information leakage without minimal user interaction and customization effort	User friendly interface mechanism	Protection XSS attacks without relying on web application providers

	Disadvantages	Lack of SSL support	Lack of freeware utility	Performance is not clear during any filter-rule related operations
Project Shield[9]	Advantages	Invalidation of caches in accordance with needs of user of sites	Advanced site protection through control for JavaScript cookies and IP blacklists	Under the hood metric for a quick review of site traffic, error rates, and bandwidth savings
	Disadvantages	Unknown		
Opendaylight[9]	Advantages	Unknown		
	Disadvantages	SDI(Software-Defined Interconnection) impacts on work efficiency		
HAProxy[9][12]	Advantages	Plug-and-play load-balancing appliance that can be deployed in any environment		
	Disadvantages	HTTP cache feature is not supported	Lack of support for POP/SMTP/SPDY protocols	
ModSecurity (WAF)[8][11][12]	Advantages	Support for known standards of major threats(OWASP Top 10 and PCI DSS(Payment Card Industry Data Security Standard))	Support for SSL/TLS	Features can be implemented in software or hardware
	Disadvantages	System can cost higher for budget	Client-aware usage	
CAPTCHA[9]	Advantages	Easily solvable puzzles by users		
	Disadvantages	Breakable puzzles by bots through the use of suitable image processing algorithms	Negative impact on user experience	

Table VI. Usability Analysis demonstrates advantages and disadvantages of using the mentioned prevention system. The features were gotten through the papers which are cited in the paper[10] and the other papers [8], [9], [11], [12], [14], [15] to show behaviour of the tools.

	Attack Analysis						
Frameworks	Cross-Site Scripting(XSS)	SQL Injections (SQLi)	Buffer Overflow	Remote Command Execution	Denial of Service(DDoS)	Performance	Tested Environment
Noncespaces[10]	P	I	I	I	I	Moderate	PHP template engine(NSmarty)
Xjs[10]	P	I	I	I	I	High	Web browsers(Firefox, Webkit, and Chromium) and Apache web server
SMask[10]	P	IP	I	IP	I	Unknown	PHP-based applications and systems
SQLrand [10]	I	P	IP	I	I	Moderate	MySQL database
XSS-GUARD[10]	IP	I	I	I	I	Low	JSP/Java based applications and Firefox browser

Table VII. Comparison table of Defence Frameworks

IP (In Progress) is indicated under the Attack Analysis category that the mentioned prevention system is partially usable that requires more testing in accordance with the papers(which are cited in the paper[10] that show frameworks behaviour one by one. The paper[10] ignored the possibility of prevention against attacks which were not theoretically proposed in the paper[10].). P(Practicable) is indicated under Attack Analysis that the

mentioned prevention system is usable against specific attack. I (Impracticable) is indicated under Attack Analysis that the mentioned prevention system is not usable against the specific attack. Evaluation of Performance category in accordance with the mentioned defence system that High represents satisfying performance of the system, Moderate represents average performance of the system, Low represents unsatisfying performance of the system, and Unknown represents no information about performance of the system.

Frameworks	Usability Analysis	Descriptions	
Noncespaces[10]	Advantages	Server-side and client-side components	
	Disadvantages	Usable in only PHP template engine(through NSmarty)	Client-aware usage
Xjs [10]	Advantages	Adaptable to various web browsers	Developer-friendly solution
	Disadvantages	Defense against only JavaScript-based injections	
Smask [10]	Advantages	Applicable for interpreter of native language or automatic source to source instrumentation	No manual changes required for protection of applications
	Disadvantages	Advanced source-source translation is needed to improve performance	
SQLrand [10]	Advantages	Possibility to prevent high profile overflow attacks	
	Disadvantages	Usage in a particular CGI(Common Gateway Interface) application	Requirement of the integration of a proxy within database server
XSS-GUARD [10]	Advantages	Helping users to input content rich HTML input	
	Disadvantages	False negative cases during utilized non-Firefox attacks	

Table VIII. Usability Analysis demonstrates advantages and disadvantages of using the mentioned prevention system. The features were gotten through the papers which are cited in the paper[10] to show behaviour of the frameworks.

Analysis of All Defence Tools and Framework

A. Client-Aware Usage/User-Friendliness

When the use of defense systems is discussed, correct configuration of any system is a very important factor. When the tools and the frameworks are looked that are found in our survey, it can be noticed that some are automatically configured during setup and some need to be manually configured by the user. Separately, in some defense systems, there are options available for the user to quickly turn the system on and off, and the system has a user-specific simplified user interface. When the Noncespaces framework[10] is given as an example, the user should have knowledge about the system configuration because the system works in a client-aware manner. If the system is not configured well, it may not be able to prevent all kinds of XSS attacks. However, it can prevent the Node-splitting attack in accordance with Noxes, even when the system is not automatically configured well. Separately, when ModSecurity (WAF) tool[11],[12] is talked, it must be configured client-aware like Noncespaces. Unless ModSecurity (WAF) tool is configured with client-aware manner, it may not be able to implement its security mechanism well against attacks. When Noxes tool[10] is proposed, unlike client-aware requirements, it includes user-friendliness features. Less interaction with the user's system configuration is taken as a basis and the information leakage in the system has been minimized. It also has a user-friendly system activation interface. Likewise, the SMask framework[10] does not require manual changes for security. At last, when the XSS - Guard framework[10] is mentioned, its user-friendliness feature can be understood by helping the system to interact with the rich HTML content of the system. Thus, It can be understood that configuration of the system has a vital role during usage of the tools and the frameworks for client-aware usage/user-friendliness.

B. Cost

The amount of money that the defense system to be used will cost depends on the purpose and content of the web application. Companies make economic evaluations of their systems according to the value of the data they have and the cyber attacks that they will reach over time. As a result of these evaluations, they plan security systems in regards to web applications. According to the tables VIII above, the cost of the secure defense system varies according to the content of the web application. According to the paper [10], when any system failure occurs in a web application, the web application that is out of work can cost a lot of money to the organization.

Separately, fixing system failure that occurred in the web application may cost an additional amount. Although ModSecurity (WAF) is mentioned as an example to prevent this situation in the papers[8], [11], [12], ModSecurity(WAF) may cost more economically than HAProxy which is briefly mentioned in the paper[10]. HaProxy and ModSecurity (WAF) are effective defense tools against DDOS attacks. These tools are especially useful for protection of web applications against SYN Flood attacks. But overall ModSecurity is more powerful in protecting the web application against other different kinds of attacks which are Cross-Site Scripting and SQL Injection attacks). In this case, the possibilities of cyber attacks against the web application and the effect of the budget affect the security system to be used.

C. Performance

The performance of defense systems is a very important factor for both the client-side and the server-side during execution times and cyber attacks. Especially the loss of time during defense can create big problems. Moreover, it can even affect the user experience, which is another analysis factor in our survey. On the other hand, the internet connection infrastructure between client-side and the server-side may also be a factor affecting the performance of used defence tools and frameworks. As an example, the paper of Mitropoulos et al. [10]. According to the JSFlow tool, the runtime can take twice as long as normal [10]. This can easily be seen that the tool performs below expectations. A similar situation can also be seen in the XSS-Guard framework [10]. But in the case of XSS-Guard, the client and the server are communicating over the same ethernet network and the conditions created are designed considering the worst performance possibilities. This situation may improve under more realistic conditions. Separately, when SQLrand[10] is discussed, the proxy in the framework can also provide some performance drops. Unlikely, When ModSecurity(WAF) tool , which has a satisfying performance, is proposed, It can be seen that the paper[12] gives good results against XSS, SQLi, and Remote Command Execution. The paper[12] is also recommended to use the HAProxy tool as a performance enhancing factor. As stated in the table (), the HaProxy tool also has a good performance. Generally, performance has an important role in the security systems.

D. User Experience

When the experience of the web application at the user side is thought, the effect of the security system to be used on the user can be an important factor. The user's web application experience should not have negative effects and restrictions from the security system. Although these security effects and restrictions do not affect the security of the web application, negative effects can be observed economically. For example, when the paper of Praseed and Thilagam [9] is examined, It is mentioned that the use of the CAPTCHA security tool on the website may negatively affect the user experience. The user must solve the puzzle-challenges that are directed by the CAPTCHA system while entering the web sites consecutively. This situation can be both time-consuming and a burden for users. This can reduce the user demand for the website. Although CAPTCHA is a good security tool for DDOS attacks, choosing a different security tool that will help the users with time factor, will affect the user experience more positively.

E. Adaptability

Being able to adapt the designed security system to different hardware and software is also a very important feature. This feature plays a very critical role in some situations. Any tool and framework may adversely affect the security prevention system that will be created to work based on only one computer language or system. In fact, it can make the system vulnerable to security threats that may occur. For example, even though the Xjs framework [10] is adaptable to various web site technologies, the security system only keeps the system safe against JavaScript injections. Other known attacks such as iframe, drive-by downloads, non-JavaScript, and phishing have the potential to put the system protected with Xjs in danger. When the Noncespaces framework[10] was shown in our survey, It can be seen that the situation is similar to Xjs. Noncespaces do not show features adaptability with systems that do not support PHP. This weakens the general protection feature of the system. SPAAS tool[14] also works on PHP based web applications as well as Noncespaces. In addition, systems that do not support security protocols such as SSL/TLS may also pose a problem for the adaptability factor. As an example, we can show the Noxes tool[10]. Likewise, HAProxy tool does not support protocols such as POP, SMTP, and SPDY. Unlikely, ModSecurity(WAF) has SSL/TLS support. Generally, adaptability factor can be critical for evasion of several types of vulnerabilities

Analysis Evaluations and Conclusions for The Categorizations

A. Pros

-Etiological approaches are promising for defense against web applications attacks. Especially, approaches like Policy Enforcement and ISR

- Mechanism in different categories is getting familiar with different deployment environments.

- User/Developer may become more familiar in deployment process in all categorizations

B. Cons

- Hybrid and Symptomatic approaches can be improved by development of new languages and techniques.

- Poor testing, high overhead, lack of publicly available prototypes, deployment difficulties, compromised security can be gaps and challenges for evaluation of all approaches.

- Modification of mechanism has to be improved in accordance with both server and client side as general

Generally, all the tools and the frameworks that were analyzed in our survey for defense systems and the majority of them were not as practical as we expected. All defense systems need to be improved and to be tested more for the use in the real environment of web application defense. When the best defense system is chosen, ModSecurity (WAF) can be proposed in accordance with our survey. The system is practicable, and it has satisfying results in the launch of a real environment. To sum up, it can be said that the use of the system is mostly common in many web applications.

V. CONCLUSION

In this survey, we have reviewed 20 references and categorized those in three sections, detection, prevention, and defense, respectively. Every tool/mechanism mentioned in this paper has their own pros and cons. Also, it is infeasible to achieve security by deploying just one or two tools against all the threats. Corresponding tools and mechanisms should be selected depending on the deployment environments and types of attacks. Also some tools are applicable to real-world threats and are already being used in the industry field, while others are impractical or need some level of optimization to function properly in practice. There is also room for improvement with regards to the tools already been used in practice. With the development of new technology, web application security tools and mechanisms must also evolve to keep up with the pace of the evolving attacks and attackers. It is still too early to say who will dominate the endless conflicts, so it is important that there is continuous research to keep the arsenal up to date to make the virtual world safe.

REFERENCES

[1] R. M. Pandurang and D. C. Karia, "A mapping-based model for preventing Cross site scripting and sql injection attacks on web application and its impact analysis," 2015 1st International Conference on Next Generation Computing Technologies (NGCT), Dehradun, India, 2015, pp. 414-418, doi: 10.1109/NGCT.2015.7375152.

[2] M. Junjin, "An Approach for SQL Injection Vulnerability Detection," 2009 Sixth International Conference on Information Technology: New Generations, Las Vegas, NV, USA, 2009, pp. 1411-1414, doi: 10.1109/ITNG.2009.34.

[3] C. M. Frenz and J. P. Yoon, "XSSmon: A Perl based IDS for the detection of potential XSS attacks," 2012 IEEE Long Island Systems, Applications and Technology Conference (LISAT), Farmingdale, NY, USA, 2012, pp. 1-4, doi: 10.1109/LISAT.2012.6223107.

[4] S. Khan and D. Motwani, "Implementation of IDS for web application attack using evolutionary algorithm," 2017 International Conference on Intelligent Computing and Control (I2C2), Coimbatore, India, 2017, pp. 1-5, doi: 10.1109/I2C2.2017.8321956.

[5] Q. Li, W. Li, J. Wang and M. Cheng, "A SQL Injection Detection Method Based on Adaptive Deep Forest," in IEEE Access, vol. 7, pp. 145385-145394, 2019, doi: 10.1109/ACCESS.2019.2944951.

[6] A. Luo, W. Huang and W. Fan, "A CNN-based Approach to the Detection of SQL Injection Attacks," 2019 IEEE/ACIS 18th International Conference on Computer and Information Science (ICIS), 2019, pp. 320-324, doi: 10.1109/ICIS46139.2019.8940196.

[7] M. Babiker, E. Karaarslan and Y. Hoscan, "Web application attack detection and forensics: A survey," 2018 6th International Symposium on Digital Forensic and Security (ISDFS), Antalya, Turkey, 2018, pp. 1-6, doi: 10.1109/ISDFS.2018.8355378.

[8] Z. Ghanbari, Y. Rahmani, H. Ghaffarian and M. H. Ahmadzadegan, "Comparative approach to web application firewalls," 2015 2nd International Conference on Knowledge-Based Engineering and Innovation (KBEI), Tehran, Iran, 2015, pp. 808-812, doi: 10.1109/KBEI.2015.7436148.

[9] A. Praseed and P. S. Thilagam, "DDoS Attacks at the Application Layer: Challenges and Research Perspectives for Safeguarding Web Applications," in IEEE Communications Surveys & Tutorials, vol. 21, no. 1, pp. 661-685, Firstquarter 2019, doi: 10.1109/COMST.2018.2870658.

[10] D. Mitropoulos, P. Louridas, M. Polychronakis and A. D. Keromytis, "Defending Against Web Application Attacks: Approaches, Challenges and Implications," in IEEE Transactions on Dependable and Secure Computing, vol. 16, no. 2, pp. 188-203, 1 March-April 2019, doi: 10.1109/TDSC.2017.2665620.

[11] T. Jain and N. Jain, "Framework for Web Application Vulnerability Discovery and Mitigation by Customizing Rules Through ModSecurity," 2019 6th International Conference on Signal Processing and Integrated Networks (SPIN), Noida, India, 2019, pp. 643-648, doi: 10.1109/SPIN.2019.8711673.

[12] D. Arnaldy and T. S. Hati, "Performance Analysis of Reverse Proxy and Web Application Firewall with Telegram Bot as Attack Notification On Web Server," 2020 3rd International Conference on Computer and Informatics Engineering (IC2IE), Yogyakarta, Indonesia, 2020, pp. 455-459, doi: 10.1109/IC2IE50715.2020.9274592.

[13] G. P. Bherde and M. A. Pund, "Recent attack prevention techniques in web service applications," 2016 International Conference on Automatic Control and Dynamic Optimization

Techniques (ICACDOT), Pune, 2016, pp. 1174-1180, doi: 10.1109/ICACDOT.2016.7877771.
[14] P. Anand and J. Ryoo, "Security Patterns As Architectural Solution - Mitigating Cross-Site Scripting Attacks in Web Applications," 2017 International Conference on Software Security and Assurance (ICSSA), Altoona, PA, USA, 2017, pp. 25-31, doi: 10.1109/ICSSA.2017.30.
[15] D. Hedin and A. Sabelfeld, "Web Application Security Using JSFlow," 2015 17th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC), Timisoara, Romania, 2015, pp. 16-19, doi: 10.1109/SYNASC.2015.11.
[18] A. Naderi-Afooshteh, A. Nguyen-Tuong, M. Bagheri-Marzijarani, J. D. Hiser and J. W. Davidson, "Joza: Hybrid Taint Inference for Defeating Web Application SQL Injection Attacks," 2015 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, Rio de Janeiro, Brazil, 2015, pp. 172-183, doi: 10.1109/DSN.2015.13.
[19] M. Taguinod, A. Doupé, Z. Zhao and G. Ahn, "Toward a Moving Target Defense for Web Applications," 2015 IEEE International Conference on Information Reuse and Integration, San Francisco, CA, USA, 2015, pp. 510-517, doi: 10.1109/IRI.2015.84.
[20] Uwagbole, S. O., Buchanan, W. J., & Fan, L. (2017). Applied machine LEARNING predictive analytics to sql injection attack detection and prevention. 2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM). doi:10.23919/inm.2017.7987433
[21] D. Hedin, A. Birgisson, L. Bello, and A. Sabelfeld, “JSFlow: Tracking information flow in JavaScript and its APIs,” in Proc. 29th Annu. ACM Symp. Appl. Comput., 2014, pp. 1663–1671.
[22] M. V. Gundy and H. Chen, “Noncespaces: Using randomization to enforce information flow tracking and thwart cross-site scripting attacks,” in Proc. 16th Annu. Netw. Distrib. Syst. Secur. Symp.,2009.
[23] E. Kirda, N. Jovanovic, C. Kruegel, and G. Vigna, “Client-side cross-site scripting protection,” Comput. Secur., vol. 28, no. 7, pp. 592–604, 2009.
[24] M. Johns and C. Beyerlein, “SMASK: Preventing injection attacks in web applications by approximating automatic data/code separation,” in Proc. ACM Symp. Appl. Comput., 2007, pp. 284–291.
[25] S. Boyd and A. Keromytis, “SQLrand: Preventing SQL injection attacks,” in Proc. 2nd Appl. Cryptography Netw. Secur. Conf., 2004, pp. 292–304.
[26] E. Athanasopoulos, V. Pappas, A. Krithinakis, S. Ligouras, E. P. Markatos, and T. Karagiannis, “xJS: Practical XSS prevention for web application development,” in Proc. USENIX Conf. Web Appl. Develop., 2010, pp. 13–13.
[27] P. Bisht and V. N. Venkatakrishnan, “XSS-GUARD: Precise dynamic prevention of cross-site scripting attacks,” in Proc. 5th Int. Conf. Detection Intrusions Malware Vulnerability Assessment, 2008, pp. 23–43.
[28] Jianwei Hu, Wei Zhao, and Yanpeng Cui. 2020. A Survey on SQL Injection Attacks, Detection and Prevention. In Proceedings of the 2020 12th International Conference on Machine Learning and Computing (ICMLC 2020). Association for Computing Machinery, New York, NY, USA, 483–488.
[16] P. Chen, H. Yu, M. Zhao and J. Wang, "Research and Implementation of Cross-site Scripting Defense Method Based on Moving Target Defense Technology," 2018 5th International Conference on Systems and Informatics (ICSAI), Nanjing, China, 2018, pp. 818-822, doi: 10.1109/ICSAI.2018.8599463.
[17] Misganaw Tadesse Gebre, Kyung-Suk Lhee and ManPyo Hong, "A robust defense against Content-Sniffing XSS attacks," 6th International Conference on Digital Content, Multimedia Technology and its Applications, Seoul, Korea (South), 2010, pp. 315-320.

APPENDIX

While the defense tools and frameworks part was developed, the papers [21], [22], [23], [24], [25], [26], [27] which are cited in the paper [10], were consulted. Generally, these papers [21], [22], [23], [24], [25], [26], [27] were used to improve the tables and to analyze the chosen tools and frameworks for the part separately. Also, the paper [10] was advanced for showing more detailed behaviour of the tools and the frameworks in accordance with the papers [21], [22], [23], [24], [25], [26], [27].

TABLE OF CONTRIBUTION

Name	Papers Summarized	Section in charge
Yunfeng Lu	[1],[2],[3],[4],[5],[6],[7], [28]	Detection mechanism and tools
Osezele Ehi-Douglas	13,16,17,18,19,20	Web Application Prevention Mechanisms
Yunus Emre Aydar	8,9,10,11,12,14,15 (21,22,23,24,25,26, 27 are extensions in appendix)	Defence Tools and Frameworks