Web Application Attacks Detection: A Survey and Classification

Nadya ElBachir El Moussaid, Ahmed Toumanari
ESSI, National School of Applied Sciences, Ibno Zohr University, Agadir, Morocco

ABSTRACT
The number of attacks is increasing day by day, especially the web attacks due to the shift of the majority of companies towards web applications. Therefore, the security of their sensitive data against attackers becomes a crucial matter for all organization and companies. Thus the necessity to use intrusion detection systems are required in order to increases the protection and prevent attackers from exploiting these data in illegal way. In this paper we begin by giving a survey of web application attacks and vulnerabilities, also approaches to improve the web application security using intrusion detection systems and scanners based on machine learning and artificial intelligence. When it comes to vulnerability, it is also an attack which exploits this vulnerability; therefore our paper presents web intrusion detection system based on detection of web vulnerabilities. Experimental results have been acquired from HTTP simulations in our network and from responses of HTTP requests sent to a bunch of websites and applications to test the efficiency of our intrusion detection system. This efficiency can be noticed from a High detection rate which is greater than 90%.

Keywords
Web Application Security, Web Application Vulnerabilities, Intrusion Detection System (IDS), Classification, Machine learning, Weka.

1. INTRODUCTION
As a result of the increasing use of the web applications by companies, these companies insist to secure their sensitive data (such as password, bank card numbers, etc...) by using Firewalls and DeMilitarized Zone (DMZ) as first line of defense, even so it still not enough to provide a decent level of security, however does not guarantee a high level of security. For that reason, comes the second line of defense by using Intrusion Detection Systems (IDS) and Web Application Vulnerability Scanners to ensure sensitive data security from attempted attack and exploitation of data in an illegal manner.

Intrusion detection systems can be classified into two major categories:

- Signature-based intrusion detection systems,
- Anomaly-based intrusion detection systems.

However, they suffer from many drawbacks which make traditional intrusion detection systems inadequate to detect new attacks or applicable in web application.

Thus, recently, many researches adopt data mining, machine learning and artificial intelligence techniques and algorithms to profit from it advantages in order to improve the intrusion detection systems performance which is measured by the rate of detection and the false alarm rate [1-6].

A Web application is not limited to a single Web server managing a set of static HTML pages. The architecture become more complex than before, therefore a web application is not only a single web server but a set of many servers and machines namely web server, application server and data base server.

The major security threats are threats that exploit the vulnerability of web applications inasmuch to the complexity of web applications and technologies also to the also the wide use of web application in almost all areas, as well as to the lack of security experiences and talents of the developers. Web application vulnerabilities are listed as follow, and will be described in detail in next section:

- Injection
- Cross Site Scripting (XSS)
- Broken Authentication and Session Management
- Insecure Direct Object References
- Cross Site Request Forgery (CSRF)
- Security Misconfiguration
- Insecure Cryptographic Storage
- Failure to Restrict URL Access
- Insufficient Transport Layer Protection
- Invalidated Redirects and Forwards

The SQL injection and cross-site scripting (XSS) are the two most famous vulnerabilities in Web application.

Various scanners have been created to detect and identify web application vulnerabilities such as open source scanners namely W3af, Skipfish and Wapiti, etc…, commercial scanners namely Acunetix, WebInspect, AppScan, etc... [7]. To identify vulnerabilities of a website, web application vulnerabilities scanners submit queries containing non-compliant data corresponding to potential attacks. The responses are then analyzed to identify the execution pages. If an execution page is identified, the corresponding page is considered vulnerable. There are two main classes of approaches adopted by web application vulnerability scanners such as:

- Recognition error message requests that the server returns
- Studying similarity of pages returned.

The main aim of this paper is to give a survey of web application vulnerabilities and how to secure it from attacks that exploits these vulnerabilities. The rest of paper is organized as follow: Section II describes the top ten web application vulnerabilities.

Section III presents detection of web application vulnerabilities. Section IV discusses the use of machine learning algorithms to detect web application vulnerabilities. In Section V proposed algorithm and then a conclusion in Section VI.

2. WEB APPLICATION VULNERABILITIES

2.1 Injection

Injection or SQL injection (SQLI) is a type of attacks that use application's vulnerabilities that interacts with a database, by injecting an unauthorized SQL query by an attacker in order to compromise its security.

SQLI is one of the most famous and popular web applications vulnerability where the attackers exploits input vulnerability and attempts to send incorrect or unsanitized command or SQL query to the application. This hostile command cans fraud the interpreter to display unauthorized data to attacker [8, 21]. A successful SQL injection allows the attacker to perform his / her intent namely:

- Read sensitive and unauthorized data from the database.
- Modify database data using Insert, Update and Delete query.
- Execute administration operations on the database.
- There are several sub-classes of SQL injection namely:
- Classic SQLI
- Blind or Inference SQL injection
- Database management system-specific SQLI
- Compounded SQLI

2.2 Cross-Site Scripting (XSS)

Attacks Cross-Site Scripting (also called XSS or CSS) are also one of the famous web applications vulnerabilities like SQLI. XSS is a type of web application vulnerabilities which the attacker injects a script in trusted web pages. These pages are returned to clients, may therefore include malicious executable code that will execute in the browser of the client. This attack therefore attempts indirectly the user of a web site through the exploitation of the vulnerabilities of this web site [8].

XSS is an attack against web application that display dynamically it content to users without checking or encoding the information entered by these users.

In addition, most browsers have the ability to interpret the scripts in web pages written in different languages, such as JavaScript, VBScript, Java, ActiveX or Flash. The following HTML tags allow incorporating executable scripts in a web page: <SCRIPT>, <OBJECT>, <APPLET>, and <EMBED>.

Attackers can use XSS to send harmful script considered trustful script by a user's browser. This harmful script can:

- Access to any cookies.
- Access to session tokens.
- Access to sensitive information retained by your browser.
- Rewrite the content of the HTML page.

We can classify XSS vulnerabilities into two classes: server and client XSS. And there are generally tree types of XSS vulnerabilities:

Stored XSS: Stored XSS is also called Persistent or Type-I XSS. This kind of XSS vulnerability occurs when the attacker stores the injected malicious scripts permanently in the target server such as:

- A database,
- A message forum,
- Visitor log,
- Comment field, etc.

Therefore the victim recuperate the malicious script when demands the stored information from the target server.

Reflected XSS: The reflected XSS is also called Non-Persistent XSS or Type-II XSS. This XSS vulnerability is occur when the attacker creates an injected malicious scripts where reflect the input user. i.e. the reflected XSS occurs when a victim user clicks on malicious link without ensuring that the link is trusted, or submit a non-trustful form, or even do any non-safe action which leads to executing a malicious script, thus as response the web application reflects the user input.

DOM based XSS: Document Object Model (DOM) based XSS is also called type-0 XSS. The DOM based XSS was published first time by Amit Klein in 2005 [9], the first two types of XSS vulnerability mentioned above exploit server-side code, whereas the DOM based XSS vulnerability have an impact on the script code being executed in the client's browser.

The most famous ways that trigger the XSS attack are:

- By clicking on an URL in e-mail
- By clicking on an URL in a website

2.3 Broken Authentication and Session Management

Broken Authentication and Session Management attacks exploits all vulnerabilities concerning the authentication and session managing which are very important features of any web application. Various action can broke the authentication even a solid one, if the account management of the web application doesn't take in consideration the re-authentication in every actions although in the case of a valid session ID like:

- Changing the account password,
- Updating the account information,
- Lack of security passwords such as weak encrypted passwords or even not stored properly,
- Also not well protected session may emerge to exploit this vulnerability by the attacker.

2.4 Insecure Direct Object References

An Insecure Direct Object References occurs when web application can afford a direct access to object and resources such as a file, directory or database key as a URL. This access is provided without any check and control or protection for example by changing and manipulating references in the URL.

2.5 Cross Site Request Forgery (CSRF)

The CSRF attack is similar to XSS but with some difference. The CSRF is an attack which forces the victim user to execute a malicious action in the web application in which he is authenticated such as submitting a forged HTTP requests through image tags, XSS or with other techniques.

2.6 Security Misconfiguration

The security misconfiguration attack exploits configuration weakness in a web application. It occurs when the system administrator and the developers ignore the lack and problems of security of the configuration of the web application. Web application security misconfiguration can be embodied in unnecessary features used by the attacker to gain unauthorized access to sensitive information even with elevated privileges if the victim user is an administrator; these features are the attacker access:

- Default accounts of a web applications provided during installation,
- Unused pages,
- Unpatched flows
- Unprotected files and directories.

2.7 Insecure Cryptographic Storage

Insecure Cryptographic Storage occur when the web application store sensitive information such as login and password or credit card number and personal information needs without encrypted stored data or using weak encryption key and algorithms, Without forgetting the weak key storage and management.

2.8 Failure to Restrict URL Access

The failure to restrict URL access occurs when the web application does not manage the access authorization of a user in safe and proper manner. These vulnerabilities in the web application can display unauthorized files or even afford privileges functions to unauthorized user. Therefore this kind of vulnerability can lead to session management problems cited above. Access control and verification must be executed to ensure the authorization of the user before any request to a sensitive function or sensitive information.

2.9 Insufficient Transport Layer Protection

The Insufficient Transport Layer Protection vulnerability is due to the fact that Web applications do not protect or poorly protected the network traffic. SSL/TSL has been used by web application only during the authentication phase. data and session IDs are in clear-text in application network flows. Expired certificates or misconfigured can also be the cause of this type of vulnerability. Improper configuration of SSL can facilitate such attacks "phishing" type "man in the middle", etc… An attacker can:

- Intercept sensitive data such as the login and password,
- Inject malicious scripting
- Redirect data flows between the original extremity
- Delete the contents of packets

2.10 Invalidated Redirects and Forwards

Web applications frequently use redirects and references to redirect users to other pages. Sometimes the target page specified in a non-validated parameter allows an attacker to choose the redirection page. An attacker can therefore try to make sure that a victim clicks on a link on a trusted site, but containing a redirection setting pointing to a malicious site, a copy of the trusted site. The victim will probably click on the malicious link, as pointing to a valid website link. Such redirects can allow for example to install malicious software to capture sensitive user information or bypass access controls.

3. WEB APPLICATION VULNERABILITIES DETECTION

Different methods and techniques can be implemented by developers and administrators in charge of IT security to deal with various threats against web applications such as firewalls, DMZ, IDS… etc.

Intrusion detection aims to identify actions and attempts that try to bypass the security policy to compromise the confidentiality, integrity or availability of a resource, and raises alerts in case of detection. To evaluate the efficiency of the IDS, following metrics are used:

$$\text{Detection rate} = (TP) / (TP + TN) \quad (1)$$

$$\text{False alarm rate} = (FP) / (FP + TN) \quad (2)$$

Where:

- **TP:** True Positive
- **FP:** False Positive
- **TN:** True Negative

Intrusion detection systems are used to detect intrusions against Web applications are classified into two principal categories mentioned above.

3.1 Signature-based Intrusion Detection Systems

The IDS of this category is based on learning techniques on known attacks so as to define their signatures [10-13].

3.2 Anomaly-based Intrusion Detection Systems

For this category of IDS, IDS is distinguished into three approaches that have different levels of analysis: a “Black box”, “Gray box” and “White box” approach. Each one is based on the type of information available to construct the reference model of application [13-19].

- **Black Box:**

The approach of "Black Box" type uses no internal information program. The model reference behavior to be defined in these approaches can be obtained from the specification of the application or deducted by learning based on the execution of the application [12-14, 16].

- **Gray Box:**

Like the "Black Box" approach, the “Gray Box” approach is based on sequences of system calls. However, it extracts additional information from the process, memory, etc… [14, 16-17]

- **White Box:**

In “White Box” approaches, the information in the code source of the program is used to build a pattern or a model of intrusion detection at the application level. This approach can be used to detect both attacks against the control flow of the application and attacks against the data [18-19].

In the following, some examples of ways to ensure the security of web applications from some vulnerability.

4. MACHINE LEARNING AND WEB APPLICATION: DISCUSSION

Machine learning is the one of artificial intelligence study which is concerned with the development, analysis and implementation of automated methods that allow systems to evolve through a process of learning, in order to fulfill the difficult tasks in reasonable time.

This section discuss how machine learning techniques aims to improve the performance of web application in detecting and preventing attackers from taking advantages of web applications vulnerabilities which are mentioned above.

Various techniques have been proposed are based on machine learning, artificial intelligence and data mining methods. Below some proposed researches:

Authors of [20] adopt user input sanitization and data mining methods. They classify several input sanitization techniques into different types, and then they apply data mining methods to predict SQL injection and cross site scripting vulnerabilities in web applications. They implement several classification algorithms such as C4.5/J48, Naïve Bayes (NB), and MultiLayer Perceptron (MLP) in WEKA tool.

In this paper [22], A. Dessiatnikoff and colleagues present an approach based on the responses of the web pages returned by the web server (execution or rejection page), these pages are analyzed using machine learning algorithms. Their approach begins by identifying injection point (i.e. vulnerable pages where the malicious code can be injected) by applying three sets of requests namely:

- **Rr:** Set of requests generated randomly to generate rejection pages.
- **Rii:** Set of SQL injection requests unsuitable for the given injection point to generate rejection pages.
- **Rvi:** Set of SQL injection requests that generate execution pages or even the rejection pages in case of non-presence of SQLI vulnerabilities.

Afterward, the next phase is classification phase by classifying response pages in clusters using the distance technique expounded below:

$$\mathrm{diff}(a_i, b_j) = \begin{cases} n - i + m - j & i = 0 \text{ or } j = m \\ \mathrm{diff}(a_{i+1}, b_{j+1}) & a_i = b_j, i < n, j < m \\ 1 + \min(\mathrm{diff}(a_{i+1}, b_j), \mathrm{diff}(a_i, b_{j+1}) & a_i \neq b_j, i < n, j < m \end{cases} \quad (3)$$

$$\mathrm{d(a,b)} = \frac{\mathrm{diff}\,(a_1, b_1)}{(n+m)} \quad (4)$$

In addition, a threshold is used to determine clusters of similar responses in combination with the hierarchical clustering technique.

However this approach has limits in identifying XSS vulnerabilities in web application.

Many researches about web scanners are done such as mentioned above based on identifying injection points which are used by attackers to exploit it in order to get full access to sensitive data. However, web scanners couldn't recognize all injection point in web application. Furthermore scanners can be used to identify these injection points by attackers themselves. Therefore the use of intrusion detection system is necessary which allows you to analyze web traffic in the case of exploiting an injection point not identified by a web scanner or even identified one, also protect web application against scanning. For that purpose, the proposed intrusion detection system is based on analyzing HTTP requests using machine learning algorithms.

Table 1. Training and Test Dataset

Datastes	Training	Test
Dmoz	60470	5738
Xssed	40606	4369
Total	101076	10107

5. PROPOSED ALGORITHM

The proposed approach adopts the black box concept, where it will analyze HTTP requests and responses, therefore the proposed IDS is able to detect web attacks without knowing the internal information such as the log files or the code analysis.

5.1Features selection

- **Source and Destination:** these two features are used to identify the sender and the receiver address which are used to detect DoS.
- **Type of Protocol:** type of protocol is also necessary to detect some attacks.
- **Length of URL:** length of the URL is one of the features that will be used to detect any attempts to insert or inject any code or script into a URL such as SQL injection or XSS.
- **Date:** this property concerns the date and time of transfer.
- **Number of Login:** this feature is useful to identify password guess attack.
- **Request Methods:** request methods are GET, POST, HEAD, PUT and DELETE.
- **SQL Injection:** this feature indicates whether the HTTP request contains an injection of a SQL query in the URL or even in the HTTP header such as cookies, User_Agent …etc.
- **Script Tags:** such as SQL injection, this feature indicates whether the HTTP request contains a Script tags even in hexadecimal code, where attackers often uses the hexadecimal code %3C%73%63%72%69%70%74%3E instead of <script>.
- **Status code**: the status code indicates that the browser is unable to provide the requested page.
- **Content length:** content length feature provide the Body length of the response.

Features selection allows the extraction of useful properties which provide a high level of detection, the following table shows the features has been used to detect web vulnerabilities:

5.2 Experiments

The experiments focus on detecting the SQL injection and XSS attacks in view of the fact that they are the most used against web applications; however that doesn't mean neglecting other attacks. Nevertheless, the same reasoning is used to detect other types of attacks.

To study the reliability of the proposed approach, the training and testing data set are built from Dmoz [23] database which

TABLE I. Results (Detection Rate)

Attacks	Proposed Approach	Snort	Dessiatnikoff approach
SQL Injection	98%	60%	95,73%
XSS attack	95,75%	57%	82%
Other attacks	90%	31,50%	80%

contain several web pages and applications, some of these pages are vulnerable, also from XSS database (xssed) [24]. The classification algorithm has been used is the improved k-means algorithm [1] and support vector machine (SVM), the reason of using it is the simplicity, detection efficiency of new attacks and optimization.

TABLE II. Selected Features

Name	Type
Source and destination	Continuous
Type of protocol	Symbolic
Length of URL	Continuous
Date	Continuous
Number of login	Continuous
Request methods	Symbolic
Sql injection	Boolean
Script tag	Boolean
Status code	Symbolic
Content length	Continuous

Our approach consists of using the improved k-means to classify the HTTP requests and webpages, than the SVM algorithm is applied to delete outliers.

The tools used for detecting web vulnerabilities and classify the web pages and application into vulnerable and non-vulnerable is the open source tool: WEKA [25], also classify the HTTP request into attack or non-attack. The cross validation is N = 10 which mean that WEKA divides data into 10 dataset where 9 dataset are used in the training phase and 1 dataset in the testing phase.

Dmoz and Xssed databases contain more than 50000 of web applications and web sites.

The next table provides a global view on the results of our experiments:
It's noticeable from the table above that our approach is more reliable than other approaches in detection rate.

The proposed approach detects 98% of SQL Injection and 95% of XSS Attacks which are the most famous web application attacks.

6. CONCLUSION

This paper presents an overview about the different top ten web application vulnerabilities especially the SQL injection and XSS vulnerabilities. Many techniques have been proposed to secure web application and browsers from attackers but even so this techniques still poor and insufficient. Thus, the use of machine learning methods and techniques make intrusion detection systems more intelligent and autonomous in detecting new attacks since the static techniques may be able to be bypassed by attackers.

Future work will be on the intrusion detection systems in the cloud environment, because cloud computing is the trend, and is a major paradigm shift of computer systems.

7. REFERENCES

[1] N. El Moussaid, A. Toumanari, M. Elazhari, "Intrusion detection based on clustering algorithm", International Journal of Electronics and Computer Science Engineering, Vol.2, p. 1059 – 1064, 2013.

[2] M. Moorthy, S. Sathiyabama, "A study of intrusion detection using data mining", IEEE-International Conference On Advances In Engineering, Sciense And Management (ICAE-2012).

[3] Y. Qing, W. Xiaoping, H. Gaofeng, " An intrusion detection system based on data mining", 2nd International Conference on Future Computer and Communication, Vol.1, p. 695-698, 2010.

[4] S. Naiping, Z.Genyuan, "A study on intrusin detection based on data mining", International Conference of Information Science and Management Engineering 2010.

[5] M. Ektefa, S. Memar, F. Sidi, L. Suriani Affendy, "Intrusion detection using data mining Techniques", International Conference on Information Retrieval & Knowledge Management, (CAMP), p. 200 – 203, 2010.

[6] C. Miao, W. Chen, "A study of intrusion detection system based on data- mining", IEEE International Conference on Information Theory and Information Security (ICITIS), p. 186 – 189, 2010.

[7] http://projects.webappsec.org/w/page/13246988/Web%20Application%20Security%20Scanner%20List

[8] R.Johari, p.Sharma, "A survey on web application vulnerabilities (SQLIA,XSS) exploitation and security engine for SQL injection", International Conference on Communication Systems and Network Technologies, 2012.

[9] A. Klein,"DOM based cross site scripting or XSS of the third kind", (WASC writeup), July 2005.

[10] J. McHugh, "Testing intrusion detection systems: A critique of the 1998 and 1999 DARPA intrusion detection system evaluations as performed by Lincoln laboratory", ACM Transactions on Information and System Security, Vol. 3, No. 4, November 2000.

[11] P.Proctor, "Practical intrusion detection handbook". Upper Saddle River, NJ, USA, 2000.

[12] G.Vigna, W.Robertson, V.Kher, and R. A. Kemmerer, "A stateful intrusion detection system for world-wide web servers", 19th Annual Computer Security Applications Conference, 2003. Proceedings. (ACSAC 2003), p. 34-43, Las Vegas, NV, December 2003.

[13] M.Almgren, Ulf Lindqvist. "Application-integrated data collection for security monitoring". Proceedings of the fourth International Symposium on Recent Advances in Intrusion Detection (RAID 2001), Vol. 2212, p. 22-36, Davis, California, Oct. 10,12, 2001

[14] S.-A. Hofmeyr, S.Forrest, A.Somayaji, "Intrusion detection using sequences of system calls", Journal of Computer Security, V:6, p: 151-180, August 18, 1998.

[15] C. Ko, G. Fink and K.N Levitt, "Automated detection of vulnerabilities in privileged programs by execution monitoring", Proceedings of the 10th Annual Computer Security Applications Conference (ACSAC'94), p:134-144, 1994.

[16] D.Gao, M.-K. Reiter, D.Song, "Gray-box extraction of execution graphs for anomaly detection". In Proceedings of the 11th ACM conference on Computer and communications security, p.318-329, 2004.

[17] C.Kruegel, D.Mutz, F.Valeur, G.Vigna, "On the detection of anomalous system call arguments", In 8th European Symposium on Research in Computer Security (ESORICS 2003), Vol. 2808, p: 326-343, October 2003.

[18] M. Cova, D. Balzarotti, V. Felmetsger, and G. Vigna. "Swaddler: An approach for the anomaly-based detection of state violations in web applications", RAID'07 Proceedings of the 10th international conference on Recent advances in intrusion detection, p. 63-86, 2007.

[19] R. Ludinard, E. Totel, F. Tronel, V. Nicomette, M. Kaaniche, E. Alata, R. Akrout, Y. Bachy, "Detecting attacks against data in web applications", 7th International Conference on Risk and Security of Internet and Systems (CRiSIS), p:1-8, 2012.

[20] L. K. Shar, H. B. Kuan Tan, "Mining input aanitization patterns for predicting SQL injection and cross site scripting vulnerabilities", 34th International Conference on Software Engineering (ICSE), p: 1293 – 1296, 2012.

[21] W. G.J. Halfond, J. Viegas, and A. Orso, "A classification of SQL injection attacks and countermeasures", In Proceedings of the international Symposium on secure Software Engineering (ISSSE), 2006.

[22] A. Dessiatnikoff R. Akrout E. Alata M. Kaˆ aniche V. Nicomette, "A clustering approach for web vulnerabilities detection", 17th IEEE Pacific Rim International Symposium on Dependable Computing, p: 194 – 203, 2011.

[23] http://www.dmoz.org

[24] http://www.xssed.com

[25] http://www.cs.waikato.ac.nz/ml/weka/