Journal of Computer Science

Original Research Paper

Machine Learning-Based Technique to Detect SQL Injection Attack

[1]Muhammad Amirulluqman Azman, [1]Mohd Fadzli Marhusin and [2]Rossilawati Sulaiman

[1]Faculty of Science and Technology, Universiti Sains Islam Malaysia, Nilai, Malaysia
[2]Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia, Bangi, Malaysia

Article history
Received: 31-12-2020
Revised: 25-01-2021
Accepted: 05-02-2021

Corresponding Author:
Mohd Fadzli Marhusin
Faculty of Science and Technology, Universiti Sains Islam Malaysia, Nilai, Malaysia
Email: fadzli@usim.edu.my

Abstract: Lack of secure codes implemented in the web apps will lead to cyber-attack because of vulnerabilities. The statistic shows that highest record on the data theft related cyber-attacks are through the SQL injection technique. Hence, an effective SQL injection detection is needed in any web system to combat this threat. In this research, machine learning technique is used where training is provided to the SQL injection detector using a training data and then is evaluated against a testing data. The research relies on the preparation of the training and testing datasets. Training sets are used by the detector to establish the knowledge base and the test set is used to evaluate the performance of the detector. The result of the detection shows that the proposed technique produces high accuracy in recognizing malicious and benign web requests.

Keywords: Machine Learning, Signature-Based, Knowledge-Based, SQL Injection, SQL Injection Tools

Introduction

As data is the most critical asset to any organisation nowadays, the rise of cyber threat and cyberattack to the organisation's database is increasing. Hackers are the culprit and threat to data privacy and as an example, they could launch an SQL Injection Attack (SQLIA) against vulnerable websites. Furthermore, there are many existing tools that can be used to check a website's vulnerabilities and execute hacking activities automatically. These tools give an attacker more chance of getting into the web system database. If a web developer has no proper security knowledge, he/she will likely develop codes that contain vulnerabilities. It is hard to implement secure codes to defend websites against such attacks. Because of that, the systems they developed are vulnerable to SQL injection attacks. SQL injection is a type of attack to manipulate the website to disclose sensitive data by injecting malicious SQL queries to the database. Hence, if the SQL injection can be recognised earlier, it can help security officer or security analyst to terminate the attack.

Intrusion Detection System (IDS) and Intrusion Prevention System (IPS) can be used to detect SQL injections (Patil *et al*., 2017). In this research, a machine learning technique is used to detect an SQL injection attack by comparing the website access log file with the knowledge-based of malicious features. Machine learning part will undergo some training and which will then be used to scan the log for classifying where the log is being injected or not. The classification will result in a malicious or benign web request.

Problem Statement

OWASP top 10 web attacks states that SQL injection is still in the first place of the most website type attack from 2013-2017 (OWASP, 2017). These reports show that the SQL injection is a dangerous type of cyber-attack and demand continuous research effort for a better defence of our web systems. A reliable detection mechanism can be incorporated into a web system so that we can secure the data from being stolen by those attackers.

In the era of 4.0 IR, many researchers moving towards the application of the artificial intelligence and machine learning as its ability to decide like a human but with better accuracy, capacity and durability (Steve, 2019). In the case of SQL injection defence mechanism, the use of machine learning will enhance the detection in the future. But there is still lack of a white box detection technique based on machine learning as many researchers preferably using a predefined algorithm in built-in tools.

As hackers becoming more sophisticated with readily available as well as customisable tools, any vulnerable web applications are exposed to hacking activity. Hence, a countermeasure must be designed in

such a way that the system is adaptive and dependable. The higher the accuracy of the detection, the more secure the web application.

Literature Review

SQL Injections

SQL injection is a technique of exploitation of database where a hacker can insert an SQL query to retrieve records from an information system, by manipulation of data input in an application (Yekta and Yekta 2008). There are many types of SQL injection techniques such as blind SQL injection, In-band SQL injection and Out-band SQL injection.

Blind SQL Injection

In Blind SQL injection, the hacker does not know whether the website is vulnerable to SQL injection or not (Kumar and Binu, 2018). Data are not shown to the web page as the administrator has implemented some security features on the website. Therefore, the attacker needs to send a payload to be able to reconstruct the database structure, observing the response from the web application to see the behaviour of the database server. There are two types of blind SQL injections, which are the Boolean and Time-based.

Boolean based: Boolean-based SQLIA is a type of blind SQL injection which the website will only show the result, only after the correct query is entered as the parameter of the request statements.

Time-based: Time-based SQLIA is a type of blind SQL injection where it operates by sending SQL query to the database, which made the database to wait for some time before responding. So, if the HTTP response shows direct or delayed response, the attacker could infer from this result.

Union Based SQL Injection

The union-based attack is one of the in-band SQL injections that uses the UNION SQL query to combine two or more SQL statement results into a single result for the HTTP response (Halfond *et al*., 2006). The attacker can easily trick an application system by making union statements like UNION SELECT<rest of injected query>.

Error Based SQL Injection

Error based SQLIA is a type of attack where the attacker can obtain the information from the database by reading the error message appearing on the web application site. This type of SQL injection attack allows an attacker to access the whole database in the server. Initially, error messages are purposely implemented to ease the developers for troubleshooting and detecting bugs. Therefore, before the website is published, the function of error handling needs to be cleared and disabled (Chaturvedi *et al*., 2016).

Related Automated SQL Injection Tools

SQL injection tools are useful for testing purposes, which is usually used by penetration testers to examine the level of security of any organisation's websites. Rather than trying from scratch, using the automated tools helps security professionals to execute SQL injection attacks. Kalilinuxtutorials.com shows the best open source SQL injection tools (Ranjith, 2018):

- **SQLMap** – Automatic SQL injection and database takeover tool
- **jSQL Injection** – Java tool for automatic SQL database injection
- **BBQSQL** – A blind SQL -injection exploitation tool
- **NoSQLMap** – automated NoSQL database pwnage
- **Whitewidow** – SQL vulnerability scanner
- **DSSS** – Damn small SQLi scanner
- **Explo** – Human and machine-readable web vulnerability testing format
- **Blind-SQL-Bitshifting** – Blind SQL-injection via bit-shifting
- **Leviathan** – Wide range mass audit toolkit
- **Blisqy** – Exploit Time-based blind-SQL-injection in HTTP-headers (MySQL/MariaDB)

Each of the tools offers different functionalities. For SQLMap, it gives user automatic SQL injections without knowing the statements' detailed codes which are used to break into the database. SQLMap has the command switches that can be used to retrieve the whole database or specific data column of any table. Hence, with successful access to data such as username and password, the attacker can easily get into the system and access the information via the front door. Wheeler (2015) uses the SQLMap in his testing because it is open source and it supports all type of SQL injections.

Machine Learning

"Machine learning is based on algorithms that can learn from data without relying on rules-based programming" (Rao, 2015). This definition means that machine learning is a technique of giving the machine permission to make its own decision by the implementation of machine learning algorithms without using programmable codes. There are two categories of Machine learning.

Unsupervised Learning

This type of machine learning learns from information which is neither classified nor labelled. The machine gives answers without any guidance from the provided information. The machine groups raw data based on patterns, similarities and differences without any training process.

This technique makes prediction without unsupervised. For example, if there are pictures of dogs and cats, the machine cannot categorise whether a picture is a cat or a dog. However, it can categorise them based on the patterns, similarities and differences of dogs and cats.

Clustering is a type of unsupervised technique that divides similar data into the same group and dissimilar data into the other group. Clustering is the basic concept of data collection in the unsupervised learning (GeeksforGeeks, 2020). This method is essential to determine the intrinsic grouping for the unlabelled data (Greene *et al*., 2008).

Anomaly detection is detecting behaviours that are deviating from normal behaviours. This method detects rare items, incidents, or findings that can raise suspicion by varying substantially from most data (Zimek and Schubert, 2017).

Supervised Learning

This type of machine learning learns from trainings, which is done by supervising the input data. The training provides the machine sufficient labelled data with the correct answer for the machine to learns and give the predictions in the future (GeeksforGeeks, 2019). This research will be using this type of machine learning to see how it provides predictions towards the detection of SQL injections. Supervised learning can be categorised into two:

1. Regression: Regression analysis is predicting the next value, based on the statistic of the previous data that had been collected. The regression method evaluates the test data a_0 and a_1 by observing the sample (x_i, y_i) (Rong and Bao-Wen, 2018). The idea is that the data is trained and scattered on the linear graph, to draw a threshold value from the training activities to distinguish the result which it will be categorised
2. Classification: Classification technique is a type of supervised method to classify the attribute of data during the training phase so that it can classify an attribute for the next decision. This method is a well-known technique and most widely used by researchers (Singh *et al*., 2013). This research is using this approach as the classifier for the detector to classify the URL logs, whether they are malicious or benign

Log File Analysis over SQL Injection

Log file analysis can be used for an intrusion detection system to observe any transactions which have been made by users in an organization. The log files often record all access by the users to any websites. Thus, it can control the access (log files can control access?) of authorised and unauthorised users to access the website to control the SQL injection attacks or any kind of attacks.

SQL injections can be detected by analysing the attribute value of queries in an access log file, by searching unusual keywords in queries that are appended into the HTTP web request such as "#, or, 1 = 1--, WHERE, WHEN" and many more inside the log which shows unauthorised user trying to access into database (Kim, 2011).

Related Work

There are several researches that have been done to detect the SQL injection attacks. Multi-stage log analysis is a method that can be used to detect web attacks, which is not limited to SQL injection attacks. The main methods used are pattern matching and machine learning. Pattern matching is used to match the log with the pattern which has been made inside the library. Machine learning is another method to use for detection. However, the advantage of machine learning is that it depends on the probability and the threshold value that have been set earlier based on the dataset certain attribute the target may have. Therefore, multi-stage log analysis is set to use both log analysis pattern matching and machine learning.

AMNESIA (Halfond and Orso, 2012) is a fully automated tools for detecting SQL injection to protect web applications. The approaches used in this method are the combination of static analysis with runtime monitoring, which will monitor the system even while the system is running. AMNESIA is known as a tool that can detect all type of SQLIA. Static analysis is used to build a model of various type of queries that have been used by the attacker to exploit the web application. This tool is able the detect SQLIA runtime, as all the queries to database will be intercepted and compared with the model.

Analysing and detecting SQL injections from the user accessed files based on behaviour and response analysis can solve the problem of existing methods such as static and dynamic analysis (Xiao *et al*., 2017). This method is claimed to have high accuracy because it considers the response of URLs and the behaviour of the user.

Proposed Architecture

Our proposed system as shown in Fig. 1 consists of mainly three main functions to complete the process which are extraction, training and detection. The access log is extracted and separated into test set and training set. Then the training set undergoes learning process by the detector and creates a Knowledge Base (KB). Lastly, the test set is scanned by the detector based on KB and classifies the result into benign or malicious web requests.

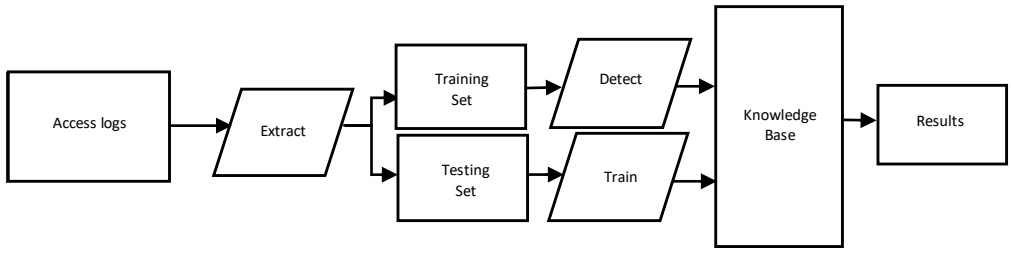

Fig. 1: The architecture of SQL injection detection system

For the overall works flow, when the attacker attacks a system, he/she injects the query command such as SELECT or UNION inside the website application system (Joshi and Geetha, 2014). The web system responds by sending the request to the application server and records any transactions in the access log file (AltexSoft, 2019).

The developed tool extracts the URL and queries which appear inside the access.log files from the application server, (Apache server) and converts it into five sets of signatures as *k*-fold cross validation (Wong and Yang, 2017). The four out of five (4/5) signature sets of benign and malicious log undergo the training phase to create KB for good and malicious signatures. The remaining set (1/5) is used in the testing phase with KB. The trained features will be the KB for the scanner to be detected against the test set. Figures 2-6 shows each of the test sets that is extracted inside the detector.

The Data Collection Technique

For this research, there is a limited amount of dataset that are available online, because company or organisation are not willing to share their access log files. So, the datasets are obtained by the testing that is done to buggy website such as Damn Vulnerability Web Application (DVWA) and bWapp. DVWA is used to get the datasets for the testing phase, which is the SQL injection queries that are appeared on the access log file provided by the web server.

DVWA can run on a localhost as it provides a full package file that contains all features of a real web application. It also provides an example database for the tester to run SQL injection attacks and try to get any information from the provided database. The site runs on Apache server and MySQL database which give the feeling of a real website application. Table 1 shows examples of commands and the corresponding log records.

SELECT, ORDER BY, INSERT and other query statements that are appeared inside the access log file are collected and documented after the testing is done from the result collected for making the knowledge based for comparison for the signature-based detections of the SQL injections (Jansen, 2006).

Fig. 2: Extracted test set 1

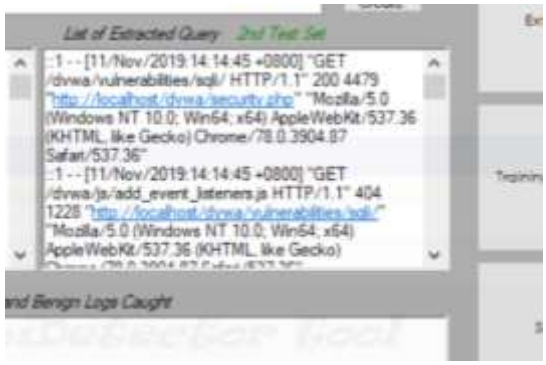

Fig. 3: Extracted test set 2

Fig. 4: Extracted test set 3

Fig. 5: Extracted test set 4

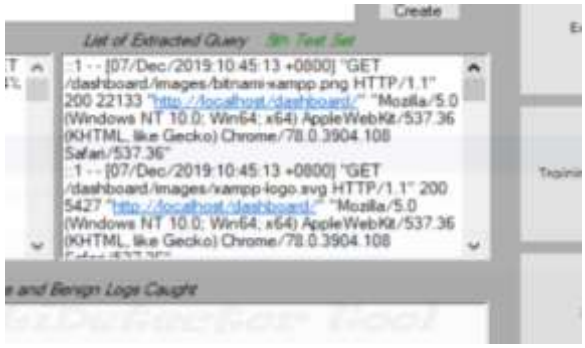

Fig. 6: Extracted test set 5

Table 1: Examples of commands used and the log records appeared

Commands	Log records
1=1#	1%3D1%23
' union all select 1, version()#	%27+union+all+select+1%2Cversion%28%29%23
' union all select 1, concat(table_schema,0x3a,table_name) from information_schema.tables#	%27+union+all+select+1%2Cconcat%28table_schema%2C0x3a%2Ctable_name%29+from+information_schema.tables%23
' union all select 1, concat(user,0x3a,password),3,4,5,6 from mysql.user#	%27+union+all+select+1%2Cconcat%28user%2C0x3a%2Cpassword%29+from+mysql.user%23

The Signature-Based Detector

Signature-based detection is generally used for malware detection over malicious files (Mujumdar *et al*., 2013). This type of intrusion detection can perform well against attacks that are already known or discovered (Kaur and Kaur, 2013). Signature based detection compares the test log file with the list of KB of malicious features. The signature is easy to implement and can perform pattern matching very fast on modern systems with minimal power.

The Classifier

Classification in machine learning is a technique for classifying test data in a particular class (Choi *et al*., 2011). The classification of access log into benign and malicious web requests is made by comparing the web requests against the malicious KB and if it is match, the detector will classify it as malicious web requests, or otherwise as benign.

String matching is used by the detector to compare and match the malicious features inside the log string. Boyer's Moore algorithm is used as it is one the best algorithm in term of performance. This algorithm arranges the texts and keywords to be compared and then test the keyword and the text from left to right. The search starts with the last keyword character and finishes with the first character (Rahman *et al*., 2017).

Evaluation on Accuracy

The result from the scanning of data samples are evaluated by comparing the expected output with detection output. If the detection output is equivalent to as expected, the results is either True Positive (TP) or True Negative (TN).

True positive is obtained if the result is expected and detected as malicious, or otherwise the result is true negative. The detection accuracy is calculated as in Equation (1) (Zhu *et al*., 2010):

$$TP + TN / TP + FP + TN + FN \quad (1)$$

True Positive and True Negative Evaluation

False alarm can occur if the expected and the detected outputs are not synchronised. False Positive (FP) can occur if the expected output is malicious but it is detected as benign. For False Negative (FN), the log is expected to be benign, but is detected as malicious.

Results

Table 2 shows an example of test logs from the first test set, which have series of test logs that are extracted from the system log file. The process rotates the data sample until the fifth test set as the final data sample.

Malicious web requests and benign web requests are identified by the SQL injection detector. With the supervised machine learning technique, the number of matched web requests are extracted and visualised in graphs to identify the accuracy of the detection, as seen in Fig. 7. The accuracy of the detection is calculated using the Equation (1). The web requests which are at the point 0 of y-axis is true positive and point 1 is false positive.

Table 2: Example of extracted web requests in the first test set

Line no	Test Set 1 (1/5 from the original access log File) Test log
1	::1 - - [24/Oct/2019:12:48:23 +0800] "GET/bWapp-master/bWapp/sqli_1.php?title=%27order+by+4%23&action=search HTTP/1.1" 200 24390 "http://localhost/bWapp-master/bWapp/sqli_1.php?title=%27or+1%3D2%23&action=search" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:69.0) Gecko/20100101 Firefox/69.0"
2	::1 - - [24/Oct/2019:12:34:57+0800] "GET/bWapp-master/bWapp/sqli_1.php?title= HTTP/1.1" 200 24390 "-" "sqlmap/1.3.3.66#dev (http://sqlmap.org)"
3	127.0.0.1 - - [24/Oct/2019:12:59:09+0800] "GET /bWapp-master/bWapp/sqli_1.php?title=%27order+by+3%23&action=search HTTP/1.1" 200 24390 "http://localhost/bWapp-master/bWapp/sqli_1.php?title=%27order+by+4%23&action=search" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:69.0) Gecko/20100101 Firefox/69.0"
4	::1 - - [24/Oct/2019:12:47:07 +0800] "GET /bWapp-master/bWapp/sqli_1.php?title=%27or+1%3D2%23&action=search HTTP/1.1" 200 24390 "http://localhost/bWapp-master/bWapp/sqli_1.php?title=sql%27+or+1%3D1%23&action=search" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:69.0) Gecko/20100101 Firefox/69.0"
5	127.0.0.1 - - [24/Oct/2019:13:02:25 +0800] "GET/dvwa/vulnerabilities/sqli/?id=%27+union+all+select+1%2Cconcat%28table_schema%2C0x3a%2Ctable_name%29+from+information_schema.tables%23&Submit=Submit HTTP/1.1" 200 44773 "http://localhost/dvwa/vulnerabilities/sqli/?id=%27+union+all+select+1%2Ctable_name+from+information_schema.tables%23&Submit=Submit" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:69.0) Gecko/20100101 Firefox/69.0"

Table 3: Evaluation result for all test sets

	Malicious web request		Benign web request	
	TP	FP	TN	FN
Test 1	32	4	22	0
Test 2	7	0	50	0
Test 3	24	0	34	0
Test 4	0	0	57	0
Test 5	0	0	56	0

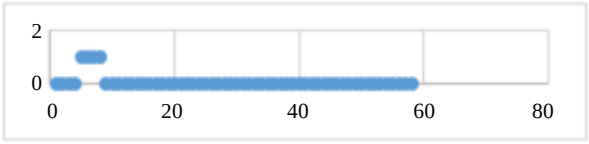

Fig. 7: Result of detection for test set 1

The results show a few of false positive detections, because the evaluation is per log detection and there are not enough signatures in the training set. Therefore, a better result is expected if the detection can be done per session detection which evaluation is by IP addresses. Also, the detector can train various malicious log if there are enough signatures for more accurate result.

Evaluation on Accuracy

From Table 3, all web requests were classified into TP, FP, TN and FN. From the result of evaluation above, the accuracy of each test can be calculated using Equation (1). Within the range of datasets that are collected, the result of detections for five test sets have high detection accuracy. With the result of 93% in the first test set and perfect 100% to the other 4 test set, we can conclude that the signature-based detection is capable of recognising for SQL injection attacks.

Discussion

In spite of promising results, the proposed technique is far from perfection for a production deployment. This research can be enhanced and improved further by having more tests against a larger size of dataset and a more rigorous fine-tuning not only on the aspect of detection accuracy but also on the aspects such as real-time detection and prevention, pre-emptive detection and detection speed without compromising the performances of the web server ability to serve the web requests are required.

Implementing a real-time detection in the system. With a real-time detection, the detection of SQL injection can be discovered and stop faster before any damages inflicted to the system. The challenge of implementing this idea is that to access the log files in real time is quite costly. This is because while the detection program trying to read the log content in real time, the web server continuously logging more access records as visitors are requesting for web content.

The solution for a real-time detection system is by extending the existing open source-based code of the web server to include the detector. Alternatively, if developed as a separate detector program, the solution can be like a near real-time where for every n-time window, the system makes attempt to retrieve the log data to draw its decision on the currently active sessions or source IP addresses. Any signs of SQL injection attack, the system can terminate or block the web requests originating from the source IP addresses.

Further research could be aimed to pre-emptively terminate and block web requests associated with SQL injection attack. This would avoid the system from receiving subsequent web requests from the attack source IP addresses.

The use of real-time ability, having huge size of knowledge base and complicated detection algorithm could cause lag in web server performances. Hence, pre-emptively stopping further web requests could help to increase the web server's speed. Further optimisation on the knowledge base and on the detection algorithm could help to increase the web server's speed too.

Conclusion

This research uses machine learning to detect malicious and benign web requests derived from the access log files, which has successfully detected malicious log files. In addition, string matching is used to match the features in the classification phase.

The main constraint of SQLI research is to acquire reputable and suitable dataset online. Therefore, data collection is developed in-house, by setting up a simple login website and perform SQL Injection attacks. Fortunately, there are platform such as DVWA that can be used to perform injections to creates datasets. As a result, only a few samples of SQL injection dataset can be used for training and testing. This research can be enhanced and improved further by implementing detection in real time, where detection of SQL injections can be discovered and stop faster before any damage inflicts the system. In addition, detecting the web request by session can improve the accuracy of the detector.

Acknowledgment

The authors are thankful to the Faculty of Science and Technology, Universiti Sains Islam Malaysia for the publication of this article.

Author's Contributions

Muhammad Amirulluqman Azman: Implemented the suggested methods and conducted the experiment and writing the paper.

Mohd Fadzli Marhusin: Defined the research problem, formulation of methods and proposed solution.

Rossilawati Sulaiman: Assisted in writing the paper.

Ethics

This article is original and contains unpublished material. The corresponding author confirms that all the other authors have read and approved the manuscript and no ethical issues involved.

References

AltexSoft. (2019). Web Application Architecture: How the Web Works. AltexSoft. https://www.altexsoft.com/blog/engineering/web-application-architecture-how-the-web-works/

Chaturvedi, V. A., Bagdi, S., & Choudhary, V. (2016). Analysis of SQL Injections Attacks and Vulnerabilities. International Journal of Advanced Research in Computer Science and Software Engineering, 6(3), 106-110.

Choi, J., Kim, H., Choi, C., & Kim, P. (2011, September). Efficient malicious code detection using n-gram analysis and SVM. In 2011 14th International Conference on Network-Based Information Systems (pp. 618-621). IEEE. https://doi.org/10.1109/NBiS.2011.104

GeeksforGeeks. (2019). Supervised and Unsupervised learning. GeeksforGeeks. https://www.geeksforgeeks.org/supervised-unsupervised-learning/

GeeksforGeeks. (2020). Clustering in Machine Learning – GeeksforGeeks. GeeksforGeeks. https://www.geeksforgeeks.org/clustering-in-machine-learning/

Greene, D., Cunningham, P., & Mayer, R. (2008). Unsupervised learning and clustering. In Machine learning techniques for multimedia (pp. 51-90). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-75171-7_3

Halfond, W. G. J., & Orso, A. (2012). Amnesia. In: Ramachandran, V. (Ed.), Encyclopedia of Human Behavior, (pp. 116–124). Academic Press, ISBN: 9780080961804.

Halfond, W. G., Viegas, J., & Orso, A. (2006, March). A classification of SQL-injection attacks and countermeasures. In Proceedings of the IEEE international symposium on secure software engineering (Vol. 1, pp. 13-15). IEEE. https://www.cc.gatech.edu/~orso/papers/halfond.viegas.orso.ISSSE06.pdf

Jansen, B. J. (2006). Search log analysis: What it is, what's been done, how to do it. Library & Information Science Research, 28(3), 407-432. https://doi.org/10.1016/j.lisr.2006.06.005

Joshi, A., & Geetha, V. (2014, July). SQL Injection detection using machine learning. In 2014 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT) (pp. 1111-1115). IEEE. https://doi.org/10.1109/ICCICCT.2014.6993127

Kaur, T., & Kaur, S. (2013). Comparative Analysis of Anomaly Based and Signature Based Intrusion Detection Systems Using PHAD and Snort. https://www.semanticscholar.org/paper/Comparative-Analysis-of-Anomaly-Based-and-Signature-Kaur-Kaur/1d2035d0cb56018d84ced89b5d235b538ae85420#extracted

Kim, J. G. (2011, April). Injection attack detection using the removal of SQL query attribute values. In 2011 International Conference on Information Science and Applications (pp. 1-7). IEEE. https://doi.org/10.1109/ICISA.2011.5772411

Kumar, A., & Binu, S. (2018). Proposed method for SQL injection detection and its prevention. International Journal of Engineering and Technology, 7, 213. https://doi.org/10.14419/ijet.v7i2.6.10569

Mujumdar, A., Masiwal, G., & Meshram, B. B. (2013). Analysis of signature-based and behavior-based anti-malware approaches. International Journal of Advanced Research in Computer Engineering and Technology (IJARCET), 2(6), 2037-2039. file:///C:/Users/WindowS%2010/Downloads/analysis-of-signature-based-and-behavior-based-anti.pdf

OWASP. (2017). Top 10 Web Application Security Risks. https://owasp.org/www-project-top-ten

Patil, A., Laturkar, A., Athawale, S. V., Takale, R., & Tathawade, P. (2017, August). A multilevel system to mitigate DDOS, brute force and SQL injection attack for cloud security. In 2017 International Conference on Information, Communication, Instrumentation and Control (ICICIC) (pp. 1-7). IEEE. https://doi.org/10.1109/ICOMICON.2017.8279028

Rahman, T. F. A., Buja, A. G., Abd, K., & Ali, F. M. (2017). SQL Injection Attack Scanner Using Boyer-Moore String Matching Algorithm. JCP, 12(2), 183-189. https://doi.org/10.17706/jcp.12.2.183-189

Ranjith. (2018). List of Best Open Source SQL Injection Tools – 2019. kalilinuxtutorials.com. https://kalilinuxtutorials.com/sql-injection/

Rao, A. (2015). Demystifying machine learning. https://bigdata-madesimple.com/demystifying-machine-learning-part-2.

Yekta, M., & Yekta, A. R. (2008). Advanced SQL Injection in MySQL. hakin9 Nr 2/2008. pp. 26-37. http://www.alirecaiyekta.com/uploads/Advanced-SQL-Injection-in-MySQL-GERMAN.pdf

Rong, S., & Bao-Wen, Z. (2018). The research of regression model in machine learning field. In MATEC Web of Conferences (Vol. 176, p. 01033). EDP Sciences. https://doi.org/10.1051/matecconf/201817601033

Singh, M., Sharma, S., & Kaur, A. (2013). Performance Analysis of Decision Trees. International Journal of Computer Applications, 71(19). https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.402.5290&rep=rep1&type=pdf

Steve, M. (2019). "Introduction to Machine Learning Model Evaluation. Heartbeat. https://heartbeat.fritz.ai/introduction-to-machine-learning-model-evaluation-fa859e1b2d7f

Wheeler, R. (2015). BlindCanSeeQL: Improved Blind SQL Injection For DB Schema Discovery Using A Predictive Dictionary From Web Scraped Word Based Lists. https://scholarcommons.usf.edu/etd/6050/

Wong, T. T., & Yang, N. Y. (2017). Dependency analysis of accuracy estimates in k-fold cross validation. IEEE Transactions on Knowledge and Data Engineering, 29(11), 2417-2427. https://doi.org/10.1109/TKDE.2017.2740926

Xiao, Z., Zhou, Z., Yang, W., & Deng, C. (2017, May). An approach for SQL injection detection based on behavior and response analysis. In 2017 IEEE 9th International Conference on Communication Software and Networks (ICCSN) (pp. 1437-1442). IEEE. https://doi.org/10.1109/ICCSN.2017.8230346

Zhu, W., Zeng, N., & Wang, N. (2010). Sensitivity, specificity, accuracy, associated confidence interval and ROC analysis with practical SAS implementations. NESUG proceedings: health care and life sciences, Baltimore, Maryland, 19, 67. https://www.lexjansen.com/nesug/nesug10/hl/hl07.pdf

Zimek, A., & Schubert E. (2017) Outlier Detection. In: Liu L., Özsu M. (eds) Encyclopedia of Database Systems. Springer, New York, NY. https://doi.org/10.1007/978-1-4899-7993-3_80719-1