SQL Injection Attack Over Web: A Survey

Teena Sahu[1], Prof. Lalit Kumar P. Bhaiya [2]
Department of Computer Science & Engineering
[1] M. Tech Scholar, BCET Durg (CG)
[2] Associate Professor, BCET Durg (CG)

Abstract- *These days by quick advancement of Internet, on the web administrations utilize web applications to introduce their administrations and utilize the web worldview are turning into an intriguing technique for application programming organizations. It permits the plan of inescapable applications which can be possibly utilized by a large number of clients from straightforward web customers.But simultaneously threats to wen application also increases. SQL injection attack is one of the much vulnerable web-based attack. IN this paper we study existing techniques which are meant for detection of SQL injection attack.*

Keywords- IDS;SQL,XSS;SQL;ML

I. INTRODUCTION

As stated by OWASP, attacks are procedures that muggers use to feat vulnerabilities in applications [OWASP, 2013]. They chase malicious intentions, such as getting illegal access, manipulating or disabling systems. In its report from 2009, the SANS Institute appraised that attacks in contradiction of web applications establish more than 60% of the entire attack challenges perceived on the Internet [Higgins, 2009]. Moreover, recent studies reveal that the number of web attacks is increasingly growing [Wood, 2014].

There are multiple types of web attacks, such as Structured Query Language (SQL) injection, content spoofing, cross-site scripting (XSS), buffer overflow, cross-site request forgery, denial of service, Operating System (OS) commanding, path traversal or Lightweight Directory Access Protocol (LDAP) injection. Web attacks can imply drastic consequences, such as gaining access to databases with sensible user data, modification of web pages, non-availability of service, execution of the attacker's code, privilege escalation or access to unauthorized files.

Static web attacks look for security vulnerabilities in a web application platform: web server, application server, database server, firewall, operating system and third-party components (such as cryptographic modules or payment gateways). These security threats comprise well-known vulnerabilities and erroneous configurations. A common characteristic of all these vulnerabilities is that they request pages, file extensions, or elements that do not form part of the web application as intended for the end user. Differently, dynamic web attacks only request legal pages of the application but they harm the expected parameters. Manipulation of input arguments can lead to different attacks of variable impact, such as errors in the application that disclose information about the platform, XSS attacks that steal information about other users or command execution by means of buffer overflows. Web attacks can be performed by different actors that can be divided into two types:

- Outsiders. They are external attackers, who do not belong to the organization.
- Insiders. They are inside the organization, therefore they can have easier access to certain systems. Ben Salem et al. distinguish between two types of insiders: traitors and masquerades [Salem et al., 2008]. These authors define a traitor as "a legitimate user within an organization who has been granted access to systems and information resources, but whose actions are counter to policy, and whose goal is to negatively affect confidentially, integrity, or availability of some information asset." Masqueraders are attackers who succeed in stealing a legitimate user's identity and impersonate another user for malicious purposes.

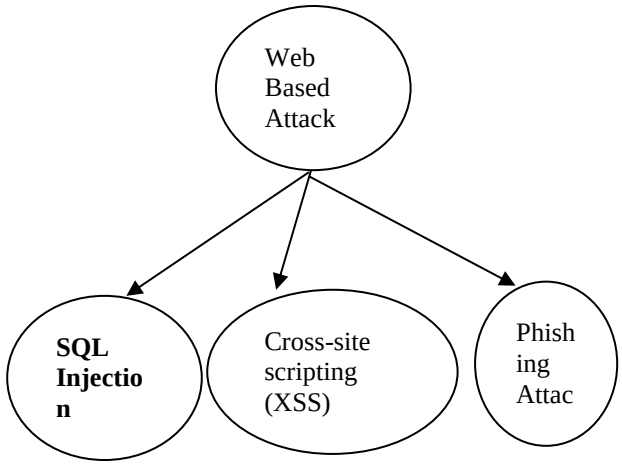

Fig.-1 Web Attack Classification

SQL Injection Attacks Structured Query Language (SQL) is an interpreted language used in database driven web applications which build SQL statements that include user supplied data or text. If this happened in a hazardous manner, then the web application become susceptible to SQL Injection Attack i.e.

when user supplied text is not correctly validated then user can alter malevolent SQL statements and can execute indiscriminate code on the target machine or alter the contents of database.

In order for an SQL Injection attack to take place, the vulnerable website needs to directly include user input within an SQL statement. An attacker can then insert a payload that will be included as part of the SQL query and run against the database server.

Types of SQL injection

- Tautology-based SQL Injection
- Piggy-backed Queries / Statement Injection
- Union Query
- Illegal/Logically Incorrect Queries
- Inference
- Stored Procedure Injection

In a tautology-based attack, the code is injected using the conditional OR operator such that the query always evaluates to TRUE. Tautology-based SQL injection attacks are usually bypass user authentication and extract data by inserting a tautology in the WHERE clause of a SQL query. The query transform the original condition into a tautology, causes all the rows in the database table are open to an unauthorized user. A typical SQL tautology has the form "or <comparison expression>", where the comparison expression uses one or more relational operators to compare operands and generate an always true condition.

Further in next section-II we will discuss different literature, in section-III we will provide a tabular comparison among literature, in section-IV we will discuss how we have motivated towards this study, finally at last section we will conclude our study.

II. LITERATURE SURVEY

Sayyed Mohammad Sadegh Sajjadi et. al. SQL injection is an attack technique that exploits a security vulnerability occurring in the database layer of an application and a service. This is most often found within web pages with dynamic content. This paper provides taxonomy on SQL injection prevention and detection approaches. Furthermore, for each type of vulnerability, we provide descriptions of how attacks of that type could take advantage of that vulnerability and perform attack. We also present and analysis some of existing detection and prevention techniques against SQL injection attacks. Finally, we compare different type of approaches and techniques and provide a list of their deployment requirements [IEEE 2013].

Anamika Joshi et. al. have proposed a method for detection of SQL injection attack based on Naïve Bayes Machine Learning Algorithm combined with Role Based Access control mechanism. Our approach detects malicious queries with the help of classifier. The addition of another parameter for RBAC has increased the accuracy of detection and also reduced number of false positives. The proposed classifier classifies the test set with 93.3% accuracy. We think that our method should be tested against larger datasets to evaluate the efficiency. The proposed method can be enhanced for detection of other types of SQL injection attacks also by extracting features appropriately [IEEE 2014].

Inyong Lee et. al. proposed a novel method for detecting SQL injection attacks by comparing static SQL queries with dynamically generated queries after removing the attribute values. Furthermore, author evaluated the performance of the proposed method by experimenting on vulnerable web applications. Author also compared our method with other detection methods and showed the efficiency of our proposed method. The proposed method simply removes the attribute values in SQL queries for analysis, which makes it independent of the DBMS. Complex operations such as parse trees or particular libraries are not needed in the proposed method. The proposed method cannot only be implemented on web applications but it can also be used on any applications connected to databases. Furthermore, it can be used for SQL query profiling, SQL query listing and modularization of detection programs [Elsevier 2011].

Yuji Kosuga et. al. said that with the recent rapid increase in interactive web applications that employ back-end database services, an SQL injection attack has become one of the most serious security threats. The SQL injection attack allows an attacker to access the underlying database, execute arbitrary commands at intent, and receive a dynamically generated output, such as HTML web pages. In this paper, we present our technique, Sania, for detecting SQL injection vulnerabilities in web applications during the development and debugging phases. Sania intercepts the SQL queries between a web application and a database, and automatically generates elaborate attacks according to the syntax and semantics of the potentially vulnerable spots in the SQL queries. In addition, Sania compares the parse trees of the intended SQL query and those resulting after an attack to assess the safety of these spots. We evaluated our technique using real-world web applications and found that our solution is efficient in comparison with a popular web application vulnerabilities scanner. We also found

vulnerability in a product that was just about to be released [IEEE 2009].

Dennis Appelt et. al. concluded that SQL injections have been ranked among the most common categories of vulnerabilities. Attacks that exploit such type of vulnerabilities increase rapidly over time. Automated testing techniques are important, not only to detect vulnerabilities in web services before they can be published, but also to reduce testing effort in contexts where the numbers of services and their input parameters are large. In particular, there is a need for black-box techniques that do not require access to the source code, as this is a common constraint when third party components are used or software development is (partly) outsourced. Existing techniques that have investigated this specific problem are bounded to known attack patterns that become out-dated very quickly, especially given the fast evolution of web services and their underpinning technologies. Their performance may also be limited by the presence of application protection mechanisms, such as firewalls, which may block known attacks. Our results confirm this problem by showing that state-of-practice, standard attacks do not, in most cases, make it through the firewall. In addition, the few that were not blocked by the firewall lead to non-executable SQL statements because of syntax errors [IEEE 2014].

Anurekh Kumar et. al. proposed a model which uses two databases one relational and other hierarchical to ensure about injection in a query, compare the results by applying tokenization technique on both databases. If the results are same, there is no injection, otherwise it is present. The proposed model uses a tokenization technique so; query containing Alias, Instances and Set operations can also be blocked at the entry point [IJSTE 2015].

Amjad Hussain Bhat et. al. said that Development of the cloud computing in recent years is increasing rapidly and gained great success, its security issues have got more and more attention. Many challenges in cloud computation increase the threat of data and service availability. There is need of many security services in order to improve cloud security for users as well as providers. In this paper, we propose a Anomaly Intrusion Detection System using machine learning approach for virtual machines on cloud computing. Our proposal is feature selection over events from Virtual Machine Monitor to detect anomaly in parallel to training the system so it will learn new threats and update the model. The experiment has been carried out on NSL-KDD'99 datasets using Naïve Bayes Tree (NB Tree) Classifier and hybrid approach of NB Tree and Random Forest[IJAIEM 2013].

III. COMPARISION

S. No.	Author/Title/Publication	Attack	Explanation	Future Bearing
1.	Anamika Joshi et. al./SQL Injection Detection using Machine Learning/IEEE 2014	SQL Injection	Author have devised a classifier for detection of SQL Injection attacks. The proposed classifier uses combination of Naïve Bayes machine learning algorithm and Role Based Access Control mechanism for detection. The proposed model is tested based on the test cases derived from the three SQLIA attacks: comments, union and tautology.	Proposed method can be enhanced for detection of other types of SQL injection attacks also by extracting features appropriately. Accuracy: - 93%
2.	Inyong Lee et. al./A novel method for SQL injection attack detection based on removing SQL query attribute values/Elsevier 2011	SQL Injection	Author removes the value of an SQL query attribute of web pages when parameters are submitted and then compares it with a predetermined one. This method uses combined static and dynamic analysis. The experiments show that the proposed method is very effective and simple than any other methods.	Future work is needed for not only SQL injection attacks but also for other web application attacks such as XSS, based on the proposed method and machine learning algorithms.
3.	Sayyed Mohammad Sadegh Sajjadi et. al./Study of SQL	SQL Injection	Author present and analysis some of existing detection and prevention techniques against SQL injection	Developer learning can be reduced.

	Injection Attacks and Countermeasures/ IEEE 2013		attacks. Finally, we compare different type of approaches and techniques and provide a list of their deployment requirements. Presented mechanisms will cover a wide range of SQL injection attacks which will culminate in a more secure and reliable database which is protected against SQL Injection Attacks	
4.	Dennis Appelt et. al./Automated Testing for SQL Injection Vulnerabilities: An Input Mutation Approach/ IEEE 2014	SQL injection	Author Presented an automated testing approach, namely µ4SQLi, and its underpinning set of mutation operators. µ4SQLi can produce effective inputs that lead to executable and harmful SQL statements.	NA
5.	Vipin Das et. al./Network Intrusion Detection System Based On Machine Learning Algorithms/IJCSIT 2010	Network IDS	This intrusion detection method using an SVM based system on a RST to reduce the number of features from 41 to 29. Author also compared the performance of RST with PCA.	Machine Learning (SVM) can improve performance
6.	Mahdi Zamani and Mahnush Movahedi/ Machine Learning Techniques for Intrusion Detection/ arXiv 2015	Single or a network of computers for malicious activities (attacks)	Characteristics of ML techniques makes it possible to design IDS that have high detection rates and low false positive rates while the system quickly adapts itself to changing malicious behaviors. Author divided algorithms into two types of ML-based schemes: Artificial Intelligence (AI) and Computational Intelligence (CI).	Noisy Training Data leads to decrease the performance of ML based IDS
7.	Anurekh Kumar et. al./Use of Query Tokenization to Detect and Prevent SQL Injection Attacks/IJSTE 2015	SQL Injection	Author proposed a model which uses two databases one relational and other hierarchical to ensure about injection in a query, compare the results by applying tokenization technique on both databases. If the results are same, there is no injection, otherwise it is present. The proposed model uses a tokenization technique so; query containing Alias, Instances and Set operations can also be blocked at the entry point.	NA

IV. MOTIVATION

On account of the across the board utilization of databases and their proclivity to contain delicate what's more, therefore lucrative information, they have turned out to be prime focuses for assault by anybody with the capacity. Tragically, this isn't usually reflected in the training of the individuals who will conceivably cooperate with databases and make applications to get to and control database information. The size of the issue is appeared by the quantity of organizations, colleges and government associations who have been bargained in the course of recent years. Most databases and programming dialects that associate with them have instruments set up to forestall assaults against those databases, however they are regularly unused in light of the fact that individuals don't know what they are and how to utilize them. After going through several literature we have identified some problem which are as:

- Developer learning is required.
- Source code adjustment is needed.

V. SQL INJECTION ATTACK AND MACHINE LEARNING

SQL injection attacks structured query language (sql) is an interpreted language used in database driven web applications which build sql statements that include user supplied data or text. if this happened in a hazardous manner, then the web application become susceptible to sql injection attack i.e. when user supplied text is not correctly validated then user can alter malevolent sql statements and can execute indiscriminate code on the target machine or alter the contents of database.

In order for an SQL Injection attack to take place, the vulnerable website needs to directly include user input within In Piggy-backed Queries type of attack is different than others because the hacker injects additional queries to the original query, as a result the database receives multiple SQL queries. The first query is valid and executed normally, the subsequent queries are the injected queries, which are executed in addition to the first. Due to misconfiguration, a system is vulnerable to piggy-backed queries and allows multiple statements in one query.

an SQL statement. An attacker can then insert a payload that will be included as part of the SQL query and run against the database server.

Types of SQL injection

- Tautology-based SQL Injection
- Piggy-backed Queries / Statement Injection
- Union Query
- Illegal/Logically Incorrect Queries
- Inference
- Stored Procedure Injection
-

In a tautology-based attack, the code is injected using the conditional OR operator such that the query always evaluates to TRUE. Tautology-based SQL injection attacks are usually bypass user authentication and extract data by inserting a tautology in the WHERE clause of a SQL query. The query transform the original condition into a tautology, causes all the rows in the database table are open to an unauthorized user. A typical SQL tautology has the form "or <comparison expression>", where the comparison expression uses one or more relational operators to compare operands and generate an always true condition.

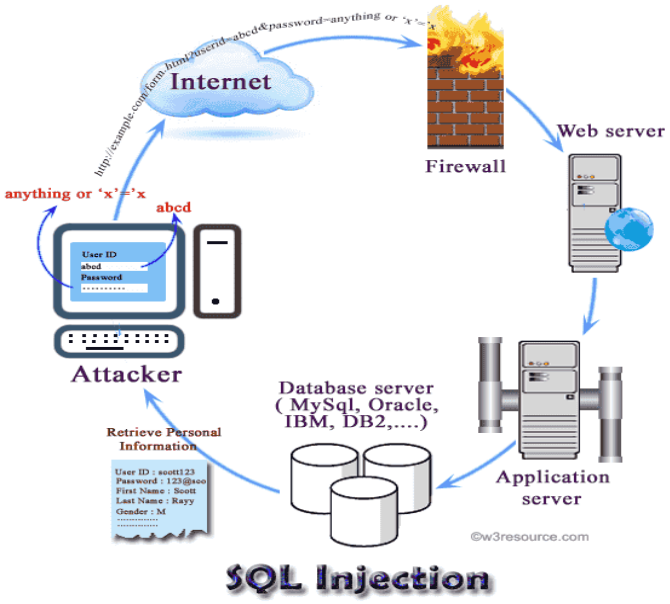

Fig.-2 SQL Injection Attack

Piggy-Backed Queries: This Type of attack inserts malevolent SQL queries into a normal SQL query. It is likely because many SQL queries can be processed if the operator ‘‘;’’ is added after each query. Following is an example. Note that the operator ‘‘;’’ is inserted at the end of query. Result of following query is to delete the user table.

```
SELECT * FROM user WHERE id='admin' AND password='1234'; DROP TABLE user; --';
```

Machine learning (ML) is a subfield of Artificial Intelligence that can help in the automation of processes. In 1959, Arthur Samuel defined it as a “field of study that gives computers the ability to learn without being explicitly programmed” [Samuel, 1959]. ML has been extensively applied in intrusion detection [Tsai et al., 2009]. Detection systems based on ML allow to quickly detect attacks while demanding much less manual work. Because of this reason, the approach is becoming increasingly important for computer security, especially when considering the huge amount of network data that IDSs need to analyse [Nguyen, 2012].

In order for an SQL Injection attack to take place, the vulnerable website needs to directly include user input within an SQL statement. An attacker can then insert a payload that will be included as part of the SQL query and run against the database server. The following server-side code is used for authenticating a user with a username and a password against a database with a table named users, and a username and password column.

```
uname = request.POST['username']
passwd = request.POST['password']
sql = "SELECT id FROM users WHERE username='" +
uname + "' AND password='" + passwd + "'"
# Execute the SQL statement
database.execute(sql)
```

The above script is vulnerable to SQL Injection because an attacker could submit malicious input in such a way that would alter the SQL statement being executed by the database server.

VI. CONCLUSION

Earlier conventional intrusion detection and anticipation techniques, alike firewalls, access control tools, and encryptions, have numerous boundaries in completely protecting networks and systems from progressively refined attacks like rejection of service. Furthermore, most systems constructed depend on such techniques agonize from high false positive and false negative detection rates (FNR/TPR) and the absence of unceasingly familiarizing to changing malevolent behaviors. Finally in this paper we have emphasize on SQL injection attacks how it works, from the comparison we can say that machine learning (ML) technique is efficient method for intrusion detection.

REFERENCES

[1] Anamika Joshi et. al. SQL Injection Detection using Machine Learning IEEE 2014.
[2] Inyong Lee et. al. A novel method for SQL injection attack detection based on removing SQL query attribute values Elsevier 2011.
[3] Sayyed Mohammad Sadegh Sajjadi et. al. Study of SQL Injection Attacks and Countermeasures IEEE 2013.
[4] Dennis Appelt et. al. Automated Testing for SQL Injection Vulnerabilities: An Input Mutation Approach IEEE 2014.
[5] Vipin Das et. al. Network Intrusion Detection System Based On Machine Learning Algorithms IJCSIT 2010.
[6] Mahdi Zamani and Mahnush Movahedi Machine Learning Techniques for Intrusion Detection arXiv 2015.
[7] Anurekh Kumar et. al. Use of Query Tokenization to Detect and Prevent SQL Injection Attacks IJSTE 2015.
[8] Y. Kosuga, K. Kernel, M. Hanaoka, M. Hishiyama, Y. Takahama, Sania: syntactic and semantic analysis for automated testing against SQL injection, in: Proceedings of the Computer Security Applications Conference 2007, 2007, pp. 107–117.
[9] Joshi Anamika ,V Geetha SQL Injection Detection using Machine Learning 2014 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT) IEEE.