DOI: http://dx.doi.org/10.26483/ijarcs.v9i1.5243

ISSN No. 0976-5697

Volume 9, No. 1, January-February 2018

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

'E-SPY': DETECTION AND PREDICTION OF WEBSITE ATTACKS

Falguni J., Serena L., Rashi B., Sanyuktha K.
Department of Computer Engineering
SVKM's NMIMS Mukesh Patel School of Technology Management and Engineering
Mumbai, India

Prof. Sanjay Deshmukh
Assistant Professor,
Department of Computer Engineering
SVKM's NMIMS Mukesh Patel School of Technology Management and Engineering Mumbai , India

Abstract—While making the world smaller , internet is experiencing an increase in cybercrimes like hacking, identity and data theft and people fall prey to malicious attacks because of lack of knowledge. SQLIA enables unauthorized users to make use of loopholes within any system to gain access to its databases and phish data , using tautologies, piggybacking, and union. These intrusions can be handled by an IDS which involves monitoring and analyzing: user activities , system activities , configurations and vulnerabilities , abnormal activity patterns , user policy violations, etc. IDS transforms the captured packets to a predefined data structure.Our proposed system's slow detection link combats the drawback of using a Snort plug-in which obtains only the port information of the source IP-address. An IDS can be integrated with a honeypot which allows the user to attack to learn the pattern of various attacks and prevent such attacks on the main system.

Keywords— Intrusion Detection System (IDS); Intrusion Prevention System (IPS); Security; Denial of Service (DoS), Structured Query Language Intrusion Detection (SQLIA)

I. INTRODUCTION

With the advent of technology the way we perceive things and perform different activities has changed, we find solutions to all our problems online. It is this blind faith in the network security which poses a threat to our information. Websites have been continuously targeted by highly motivated malicious users to acquire monetary gain and there had been an increase in cybercrimes over the years. Data thefts, data phishing, embedding malicious software's and unauthorized access to personal data are some of the most prominent threats the society faces. This paper proposes a solution to these problems with the help of "e-Spy", an Intrusion Detection System.

The main objective of our project is to detect and predict the violations and notify the administrator in order to prevent the user from gaining further access (using IDS). Our proposed system will make use of the HONEYPOT system to combat different types of SQLIA attacks.

In the succeeding sections we shall discuss the different types of SQLIA attacks, the methods to combat them and ourproposed model which shall make use of algorithms and the honey pot system to efficiently detect intrusions and alert the administrator regarding the same.

II. RELATED WORK

A. *Literature Cited*

TABLE 1: LITERATURE TABLE – HONEYPOT SYSTEM

Name of The Paper and Author's Name	Description	Inferences
Honey Pot Systems Explained [10]	A record of the intruder's activities is maintained to gain insight into attack methodologies and protect real production systems.	• The Honey Pot system should appear as generic as possible to lure the intruder into attacking it.
Honey Pot and Scan Detection System, by Chunmei YIN, Mingchu LI, Jianh MA, Jizhou SUN [8]	Monitors network traffic for a series pattern of malicious activities. Intrusion is caused by • Port-scanning • Intruding and residing in the system • Planting Trojans • Clearing trails	• Honeypot enabled IDS • Given system has very low false positives and false negatives, high real-time quality and good stability.

TABLE 2: LITERATURE TABLE – DETECTION OF SQL INJECTION ATTACKS

Name of The Paper and Author's Name	Description	Inferences
Detecting SQL Injection Attacks Using SNORT IDS,by Hussein Alnabulsi, MdRafiqul Islam ,QuaziMamun [7]	In SQL injection attack, the attackers take advantage of poorly coded web application software to introduce malicious code into the system to could retrieve important information.	• If the IDS finds a suspicious action, it creates an alert which contains information about the source, target, and preview type of the attack. • High success rate in detection with zero false alarm. • Slow processing due to use of PCRE
Detecting and Preventing SQL Injection Attacks: A formal Approach, by Mohammad Qbea'h, Mohammad Alshraideh, KhairEddinSabri [4]	SQL Injection attacks: There can be two types of attacks • Tautology • Alternate encodings which consists of six operations	• Prevent real attack expression and does not put constraints like restricting the usage of '=' in the password. • This

	namely char, ascii, unicode, nchar, convert and cast	technique can only detect attacks not predict attacks.
Detecting Various SQL Injection Vulnerabilities using String Matching and LCS Method, by Anitha V, SuphaLakshmi,RevathiM,Selvi K [6]	Various Techniques used in SQL injection attacks are : • Tautology • Piggy Back : • Union : LCS algorithm is applied for overcoming Piggy Backing and Banner Grabbing attacks while brute force are used for tautology and union	• The algorithm detects the attacks efficiently and it takes very less time to detect the web page
*Analysis & Detection of SQL Injection Vulnerabilities via Automatic Test Case Generation of Programs,*by Michelle Ruse, TanmoySarkar, SamikBasu [2]	SQL injection detection techniques rely on enforcing certain rules that must be satisfied by the user-inputs.	• Compiling SQL queries helps identify the dependencies between query conditions at various locations. • Does not produce any false positive or false negatives. • The rules for detection are platform dependent.
Improving Web Application Firewalls to Detect Advanced SQL Injection Attacks, by Abdelhamid MAKIOU, Youcef BEGRICHE ,Ahmed SERHROUCHNI [5]	• Management of Security Rules • Frequent attacks through various HTTP headers	• A novel approach to dissect HTTP requests in order to cover most evasion techniques and improve security rules management

TABLE 3: LITERATURE TABLE – IDPS

Design of Anomaly-based Intrusion Detection and Prevention System for Smart City Web Application using Rule-Growth Sequential Pattern Mining,[3]	• Sequential pattern analysis of web usage is done to to predict and prevent intrusion using Rule-Growth algorithms.	• Is able to detect new kinds of attacks. • The mechanism consumes resources process

B. Intrusion Detection System

Intrusion Detection systems work towards identifying attacks or any suspicious activity happening across the network and implement appropriate functions like notifying the administrator etc. It monitors the user access behavior, as the pattern that an authentic user will follow to gain access to his/her data will differ from that of a malicious user.

C. SQLIA

SQL Injection Attacks are when the intruder or hacker injects the system using malicious code based on SQL or Structured Query Language. SQL is a very commonly used for intrusion due to its flexibility and simplicity of syntax.

D. Honeypot

Honeypot is a virtual machine that contains information of all the possible attack scenarios, any suspicious activity from the users is analyzed, processed, and matched with the existing scenarios and suitable action is taken in case of an attack. It encourages hackers to continue with their attack in order to trace and explore the various possible attack scenarios, and adds information to the system in case any new scenarios are discovered.

III. SQL INJECTION ATTACKS

SQL injection attack represents a genuine security danger among the Internet users these days and its harmful defects are found in the Web applications. In SQL injection attack, the assailants can exploit ineffectively coded web application programming to bring malevolent code into the framework as well as could recover essential data. It is an attack methodology that targets the data residing in a database through the firewall that shields it. SQL Injection Attacks occur when an attacker is able to insert a series of SQL queries into a system by manipulating user input data into a web-based application.

SQLIA can be categorized into two forms and which can be converted from one form to another as both belong to regular languages: Finite automata and Regular expressions.

There are three categories in which SQL Injection attacks have been broadly divided into - In Band , Out Band , and Inferential.

There are five main classes of SQLI Attacks that intruders use, to break into a system, namely

A. Tautology - In tautology attacks, the intruder appends some conditional statement through the user parameters that are always true..
B. Piggybacking - The intruder sends the malicious query along with the main query with the help of connectors such as ';', Even in case of an error, the malicious query gets executed successfully.
C. Union - Similar to a tautology attack, the intruder sends the malicious query through the UNION command. Hacker gets to know any particular user's details through the error message.
D. Banner Grabbing – Even in this type of attack, information is revealed to the intruder through error messages. The difference between banner grabbing and union attacks is that, in a banner grabbing attack, the intruder gains information of the database used and can then attack the entire back end using brute force techniques.
E. Alternate encodings- This technique uses char ,ascii , Unicode , nchar , convert , cast to disguise the query and break into the system.

SQLIA attacks have the following ill-effects

- Loss of confidentiality
- Loss of integrity
- Loss of authentication
- Loss of Authorization

Websites are highly prone to SQLI attacks, as they are generally configured in a distributed environment, and also because any web application consists of three-tier architecture. There are two main techniques that can be used to break the HTTP protocol bypass the web application firewalls -

A. **SQL Injection attacks by Code Obfuscations** - Whenever an input is given from the clients' side, it passes through various checks based on the security rules of the HTTP Protocol. These security rules are such that they supposedly consist of every possible attack scenario and are therefore very complex.
Attackers take advantage of this fact and use SQL to attack the system, they disguise the query in such a way that it bypasses all the pattern matching algorithms or filters defined by the security rules of the HTTP Protocol.

The following are common code obfuscations observed in SQLI attacks

- Uppercase/lowercase blend: Pattern matching algorithms or filters often ignore mixed case words. For Eg. OR can be written as oR.
- Tautology: Refers to embedding a condition that is always true along with the actual query.
- SQL keywords: Manipulating the SQL keywords in such a way that they do not appear like keywords and are therefore ignored by the filters, and executed by the compiler.
- The use of transcoding functions: Refers to giving input in an encoded form that only the compiler can understand and execute.

B. **SQL Injection attacks using Headers**

SQL Injection attacks that are done using the headers break into the system in a very clean manner. Since applying pattern matching algorithms on headers can be a tedious task for the system, it often doesn't apply checks on each and every header coming in. Attacks can take place through the following headers -

- User-agent Header: E-commerce applications store information about the client side like web browser, audit application in this header for authorization of clients. An intruder can exploit this header using a tautology.
- Referer Header: Referer header is also highly vulnerable to SQL injection attacks as the application is storing it in database without sanitizing it. An intruder can exploit the referer header using a tautology.
- X-Forwarded- For Header: X-Forwarded header is used to gain the original IP address of a client who is connected to the server through a proxy or any other load-balancing device. The intruder can inject this header using a tautology attack.
- Cookie Header: Most of the times, the application does not perform the validation process itself, but passes the obtained information to the database using an SQL Query through the cookie header. At this stage, the intruder can inject his SQL code in this header and completely mess up the whole process .He/She can use tools such as cookie manager to forge a valid cookie as cookie variables sometimes are not checked and encoded in a proper manner before being used in the SQL query.

IV. SQLI DETECTION AND PREVENTION

The primary challenge in detecting and avoiding such injection attacks stems from the fact that it is difficult, if not impossible, to check for all possible forms of malicious user-inputs. Typically, SQL injection detection (we will refer as SQLID) techniques rely on enforcing certain rules that must be satisfied by the user-inputs. In the event, if the user inputs violate any one of these rules, the query is deemed malicious and not allowed to execute. While the existing ru1e-based techniques provide promising results, they suffer from the drawback that the rules for SQLID are constructed from the syntactic structure of the SQL query. SQLID techniques can be classified into two broad categories. One class of techniques is concerned with the embedding of SQL queries in Web programs. The other class of techniques works with the "raw" SQL query and tries to identify when one or more user inputs can exploit the query vulnerability.

There are various methods that we will be considering when we talk about SQLI Detection

A. TECHNIQUE USING CREST TOOL

This technique consists of three main steps: (a) compiling SQL queries to a target language (in our case C) to capture the dependencies correctly (b) applying an existing test generator (in our case CREST) to obtain the test cases corresponding to conditions at different locations leading to possible injection vulnerability exploitation; and finally (c) analyzing the test cases to identify the cause of the vulnerability that can be effectively used during run-time monitoring of the query-executions.

The primary objective of our translator is to generate a program which captures conditions at different locations in the query and their inter-dependencies that can maliciously affect the query result. In the event the 'WHERE' condition is atomic, thus, the query becomes vulnerable whenever the condition at that location becomes a tautology.

Using the CREST tool it is possible for us to generate a number of highly efficient test cases. It permits efficient monitoring of user inputs at runtime and removes all redundancies in the conditions. At runtime the user inputs provided are checked to see whether they conform to the requirements generated by the CREST Tool or not, a failure indicates that the inputs entered were intrusive in nature.

Decision trees are generated by mapping each node to a variable and the directed edges to their valuations. A path from the root node to the leaf node in the tree indicates the path of evaluation of a variable and its result. Removal of redundancies from a decision tree gives rise to decision diagram which is a Boolean operation. It is based on a recursive backward exploration of the decision tree and has a complexity of O(N log(N)), where N is the total number of nodes in the decision Tree.

The given technique has two main advantages, it does not produce any false positive or false negatives and it produces results that capture exactly the cause of SQL injection with respect to user inputs.

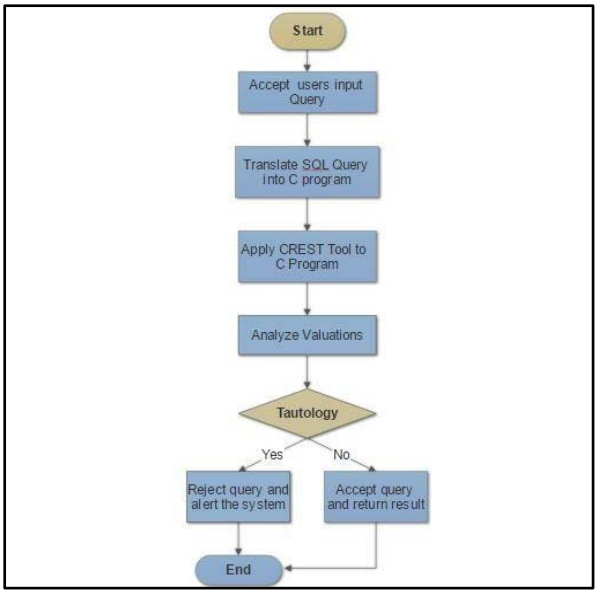

Fig1. Identification of tautologies with the CREST Tool

B. HYBRID INJECTION PREVENTION SYSTEM

HTTP PROTOCOL DISSECTION METHOD

A very efficient method used to dissect and thereby parse the HTTP protocol that makes security rules easier to manage, resulting in better results due to better organization.

In our project, the incoming HTTP stream is being segregated into two tables namely, HTTP-request and HTTP-response, each of these tables is attached to a hook of pattern matching algorithms or security rules.

The following is a mechanism using which we can implement this algorithm.

a) Typical HTTP request: The encoding technique used by headers for their requests is ASCII format. An HTTP request consists of three things - The method the URL and the protocol version. The delimiter between different headers is \r\n.
b) HTTP Request Dissection: The dissection module separates the headers and requests. Dissection is performed after the module has obtained complete information on the pattern matching algorithms or security rules.

A Hybrid Injection Prevention System (HIPS) consists of a supervised machine learning module known as 'classifier' – which monitors the HTTP stream strictly; the suspicious queries are predicted and forwarded for further checking.

The following are the steps, implementing which the HIPS system works

1. Request for URL: The system extracts the URL from the information received.
2. Check for legitimacy: The URL is considered legitimate if it does not contain any suspicious code. If not legitimate: The detection engine loads all the security rules hooked to the HTTP URL hook
3. Check for Pattern: If any of the patterns that are part of the HTTP request, happen to match with the scenarios mentioned in the security rules, the request is rejected.
4. No rule matched: If none of the rules match the pattern of the given HTTP request, then it is sent for further processing. This process continues till the last part of the data stream has been received and analysed.

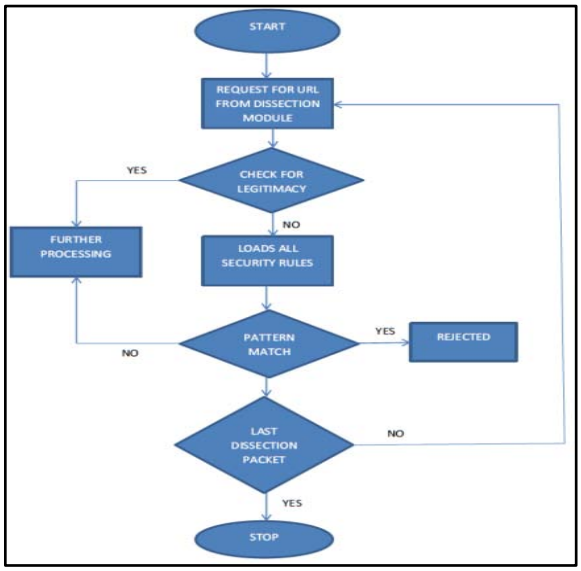

Fig 2: Working of HIPS

C. USE OF MULTIPLE ENCRYPTION

The users name and password are symmetrically encrypted with his/her private key and the query entered is encrypted using the public key of the server. However, this method fails to combat URL based SQL injection attacks and it is difficult to maintain every users private key at server side and client side.

D. USING NETWORK RECORDING,

This method uses network forensic techniques and tools to analyze the network packets containing get and post requests of a web application. It uses network based IDS to trigger network recording of suspected application attacks. Some disadvantages of this approach are, difficulty to record high traffic.

E. USING AUTOMATIC TEST CASE GENERATION,

The main idea behind this framework is based on creating a specific model that deals with SQL queries automatically. The CREST tool is an example of this framework.

F. USING THE COMBINATORIAL METHOD

This method uses both the static and dynamic approach to detect SQL injection. It is a signature based SQL injection detection technique. In this approach they generate hotspots for SQL queries in web application code and divide these hotspots into tokens and send it for validation.

G. USING AMNESIA

This method is a completely automated method consisting of two phases. The static phase which deals with analyzing the web code and generating the SQL query models and the dynamic phase which deals with generating dynamic SQL queries and checking them for similarity the statically generated queries. AMNeSIA tool makes use of Java String

Analyzer (JSA) library to statically analyze the source code and get the query models. Thus it can't be used for web applications other than those built on JSP such as PHP orASP.

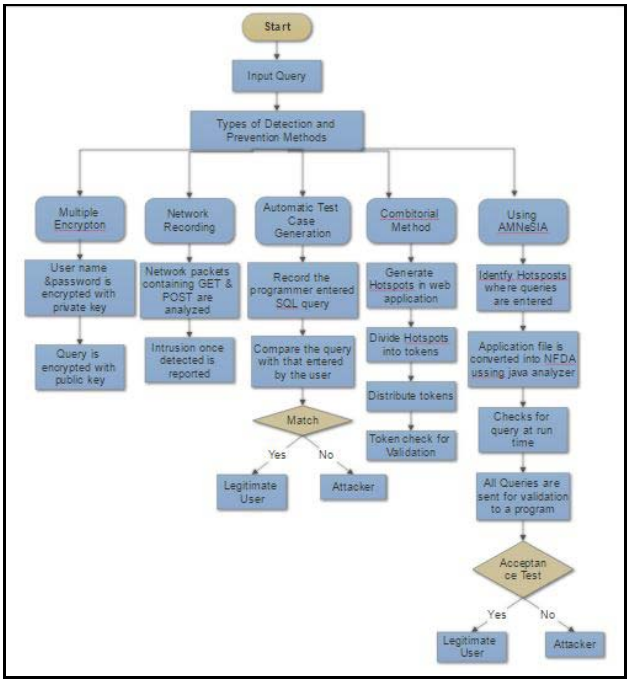

Fig. 3: SQLIA Detection Methods

H. USING AUTOMATIC CREATION OF SQL INJECTION AND CROSS-SITE SCRIPTING (XSS) ATTACKS,

I.

This method is used for finding vulnerabilities in Web Application such as SQL injection attacks and Cross site scripting. It uses static code analysis to find vulnerabilities. This technique works on the source code of the application to discover vulnerabilities in the code. It is based on input generation, taint propagation, and input mutation to find variants of an execution that exploit vulnerability.

J. USING SNORT IDS

This method utilizes SNORT device by increasing some of extra SNORT rules. SNORT is a very powerful and a mainstream control based Network Intrusion Detection System instrument, a parcel sniffer that screens organize activity continuously and bolsters conventions including TCP, UDP, IP and ICMP.

K. USING ASP.NET

Embedding a vulnerable code of ASP.net for input boxes of USERNAME and PASSWORD before the expressions with which the entries are to be matched checks for tautologies and on matching, detect an attack and also prevent it. This method helps to detect attacks in languages other than English as it covers all the 216 unicode characters.

L. USING ALTERNATE ENCODINGS

Alternate encodings with characters like char, Unicode, ascii , convert , cast , nchar can be prevented as this method compares the received input with predefined Regular Expressions of the above mentioned encoding.

M. USING SCULPTOR TOOL

This tool is used to test the module before and after the addition of prevention code. This technique does not prohibit the use of '=' or 'or'. Rather, it allows the use of expressions such as 'or 1', 'or 1=' etc It prohibits attacks in languages other than English, provides immunity against operators like 'cast' and tautologies which include the 'like' operator.

V. PROPOSED SYSTEM

A. PROPOSED ARCHITECTURE

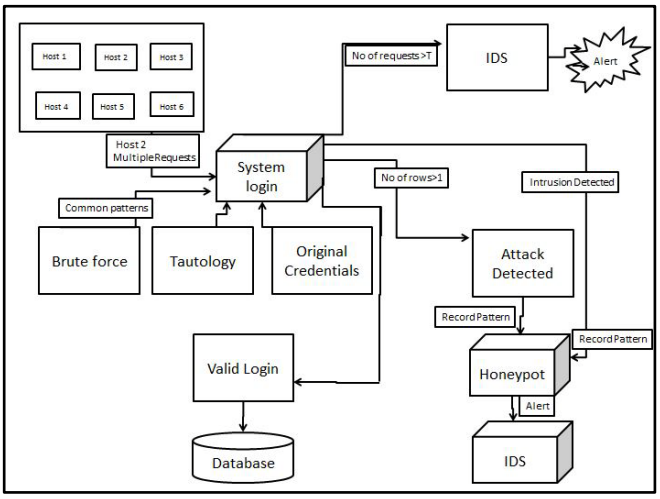

Fig 4: Architecture of IDS

The user will try logging in through the System Login portal. Assuming that the system can encounter four scenarios:

1. User logging in with the original credentials. In such a case , the system , the number of rows displayed will be just one and thus the user will be directed to his account ; or user logging in with the incorrect credentials. In such a case , the system , the number of rows displayed will be zero and thus the user will not be directed to his account.

2. User logging in with a tautology query. In such a case , the system , the number of rows displayed will be more than one and thus the user will get unauthorized access to all the accounts

3. A Bot trying to sneak in by putting in common passwords (Brute force attack).

4. A single host (or multiple hosts in Ddos) trying to flood multiple requests and block the website.

5. In the second scenario, an attack is detected and alert is sent to the IDS to take action and honeypot captures the pattern of the attack. In the third and the fourth scenarios , the user tying to attack the website is blocked as the requests and attempts exceed a certain threshold.

B. USE CASE DIAGRAM

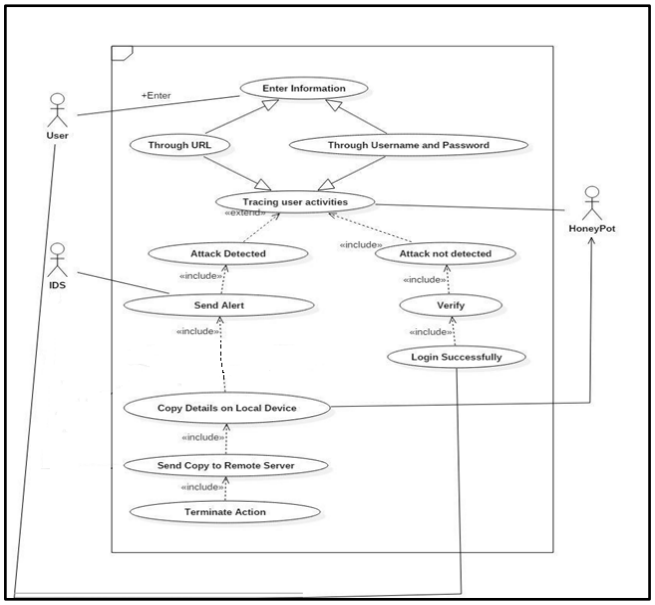

Fig 5: Use Case Diagram of Login Activity

C. ADVANTAGES OF PROPOSED SYSTEM

- Verifies success or failure of an attack: Since an IDS uses system logs containing events that have actually occurred, they can determine whether an attack occurred or not.
- Monitors System Activities: An IDS sensor monitors user and file access activity including file accesses, changes to file permissions, attempts to install new executables etc.
- Near real time detection and response: Although IDS does not offer true real-time response, it can come very close if implemented correctly.
- Low entry cost: Can be used by small firms that cannot afford to install an expensive IDS to ensure security of data.

VI. EXPERIMENTAL SURVEY

A short survey was conducted to know about how much knowledge people have about security of data and jargons that come with it. Almost cent percent people have a fear of losing their confidential data over the internet so they are aware of the fact that data can be misused if shared over the net. Considering the fact that the people participating in the survey were from an IT background, 48.5% people gave a positive response when asked if they knew about IDS and 66.7% said that they knew that data is prone to attacks through the application and the network layers.

When asked about SQLIA and tautologies, almost half the audience gave a negative reply and most of the remaining weren't sure of what it is but claimed to have heard of it. But inspite of less knowledge about these and other jargons like honeypot , almost everybody admitted to the fact they are not willing to provide their credit/debit card details to a lesstrusted website.

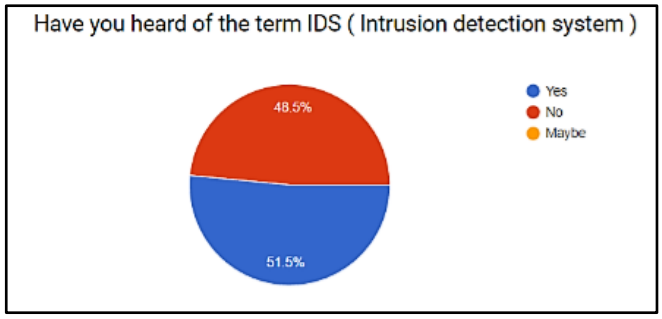

Fig 6: Graphical representation of survey on awareness of people about security and related jargons (1)

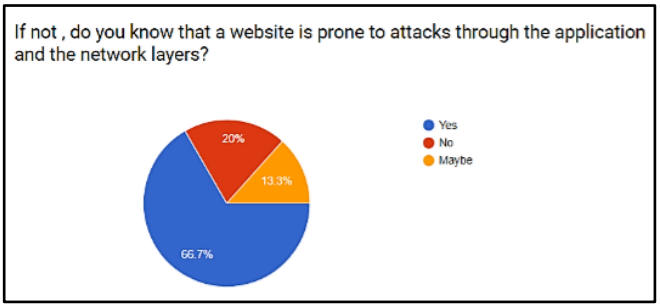

Fig 7:Graphical representation of survey on awareness of people about security and related jargons (2)

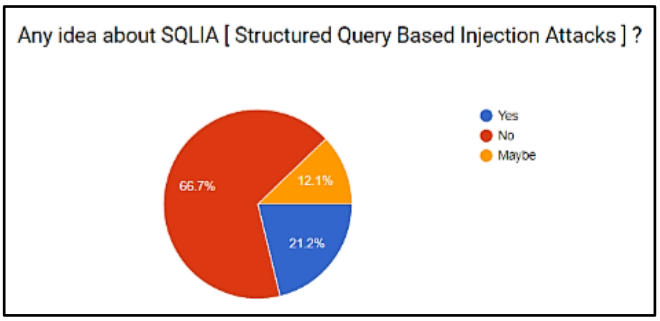

Fig 8:Graphical representation of survey on awareness of people about security and related jargons (3)

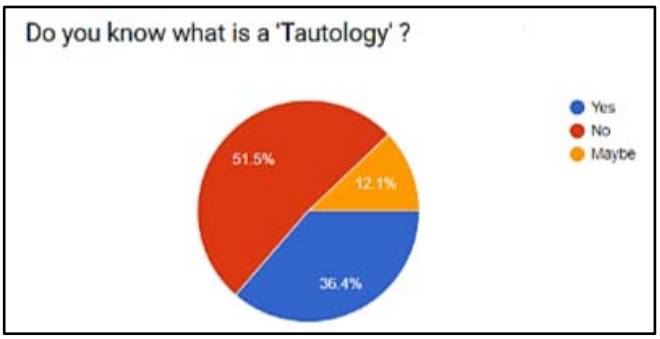

Fig 9:Graphical representation of survey on awareness of people about security and related jargons (4)

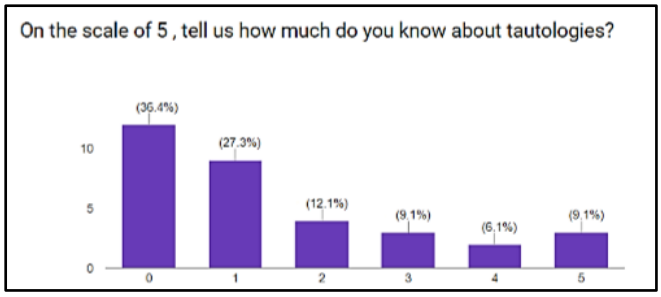

Fig 10:Graphical representation of survey on awareness of people about security and related jargons (5)

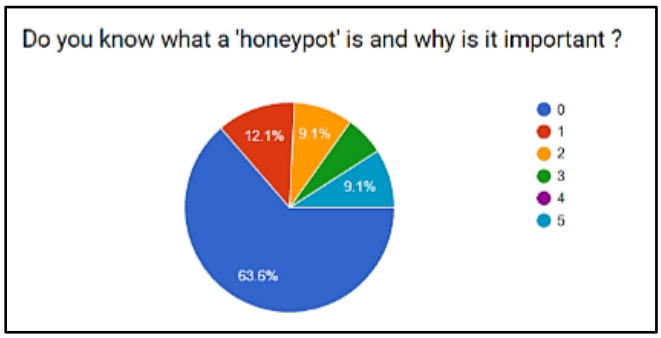

Fig 11:Graphical representation of survey on awareness of people about security and related jargons (6)

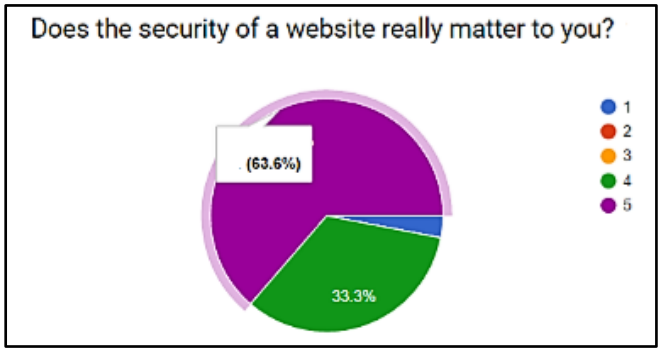

Fig 12:Graphical representation of survey on awareness of people about security and related jargons (7)

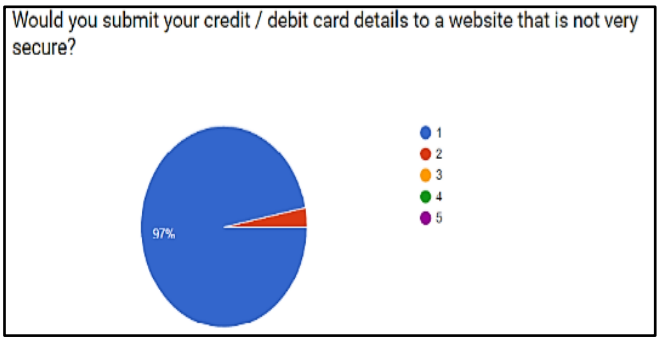

Fig 13:Graphical representation of survey on awareness ofpeople about security and related jargons (8)

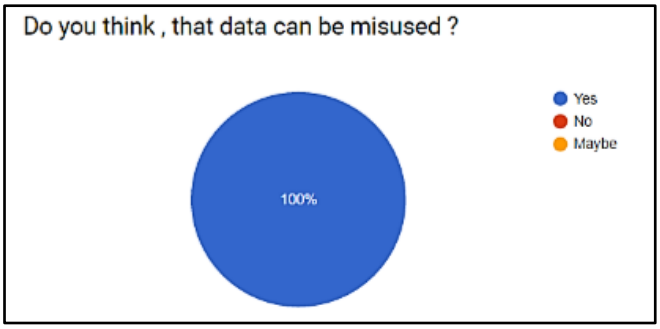

Fig 14:Graphical representation of survey on awareness of people about security and related jargons (9)

VII. CONCLUSION

E-spy can beused as low cost Intrusion Detection System that can benefit the society as a whole as it will encourage and ensure security of data among all the small organizations.

This system can be useful for monitoring and maintaining secure access to data and web applications.The project keeps track of the patterns used by different users while accessing data, and stores the attack scenario patterns dynamically.

The system is well efficient and equipped to combat all SQLIA attacks and denial of service and also solves the problem of slow detection. The process is rapid in notifying the administrator whenever required.Also, once the threshold value is reached of a user to send connection requests (flooding), the system automatically sends the warning message to the administrator.

E-spy can be a strong competitor to modern day Intrusion Detection Systems.

VIII. ACKNOWLEDGMENT

We owe a special thanks to Prof Prathamesh Churi for supporting, reviewing and guiding us throughout to enhance the quality of the paper.

REFERENCES

[1] Rahul Johari and Pankaj Sharma, “A Survey On Web Application Vulnerabilities (SQLIA, XSS) Exploitation and Security Engine for SQL Injection” in International Conference on Communication Systems and Network Technologies, 2012.

[2] Michelle Ruse, TanmoySarkar and SamikBasu, “Analysis & Detection of SQL Injection Vulnerabilities via Automatic Test Case Generation of Programs” in 10th Annual International Symposium on Applications and the Internet, 2010.

[3] YohanesWahyu Trio Pramono and Suhardi, “Design of Anomaly-based Intrusion Detection and Prevention System for Smart City Web Application using Rule-Growth Sequential Pattern Mining” in IEEE, 2014.

[4] Mohammad Qbea’h, Mohammad Alshraideh and KhairEddinSabri, “ Detecting and Preventing SQL Injection Attacks: A formal Approach” in Cybersecurity and Cyberforensics Conference, 2016.

[5] Abdelhamid MAKIOU, Youcef BEGRICHE and Ahmed SERHROUCHNI, “Improving Web Application Firewalls to Detect Advanced SQL Injection Attacks”, in IEEE, 2014.

[6] Anitha.V, SuphaLakshmi.A, Revathi.M and Selvi.K, “Detecting Various SQL Injection Vulnerabilities using String Matching and LCS Method”, in Sixth International

Conference on Advanced Computing, 2014.

[7] Hussein Alnabulsi, MdRafiqul Islam and QuaziMamun, "Detecting SQL Injection Attacks Using SNORT IDS", in IEEE, 2013.

[8] Chunmei YIN, Mingchu LI, Jianh MA and Jizhou SUN, "Honeypot and Scan Detection in Intrusion Detection System", in IEEE, 2004

[9] UmeshHodeghattaRao and BishwaPrakashPati, "Study of Internet Security Threats Among Home Users", in IEEE, 2012.

[10] Loras R. Even , 'Honey Pot Systems Explained' , July 12, 2000, URL : https://www.sans.org/security-resources/idfaq/what-is-a-honeypot/1/9