An Intellectual Approach for Providing Secure Environment in Real World Web Application

C.I. Arthi
Vivekanand Education Society's Institute of Technology, Mumbai, India

Priyadharshini R.
Shanmuganathan Engineering College, Anna University, Tamilnadu, India

ABSTRACT

In recent years, widespread adoption of the internet has resulted in rapid advancement in information technologies. Internet is used by the general population for the purposes such as financial transactions, educational activities and countless other activities. This development of the Internet use has unfortunately been accompanied by a growth of malicious activity in the web application. Every organization uses web application for accessing their data from the database, these applications use user inputs to create a query for storing, retrieving data from the database. Although all the organizations are concerned about their data security, the attackers can still corrupt the data by using the techniques like SQL injection, Hijack future session attack, Privilege Escalation attack, which are commonly referred to as intrusion, it also includes another Application layer attack called DDoS attack. In this paper we present a novel approach for detecting all the intrusions and Application layer attack by analyzing user behavior and analyzing the user input queries using low interaction Honeypot technique.

General Terms

Web Application Security, Mapping Model Algorithm.

Keywords

Denial of Service attack, Hijack Future Session attack, Honeypot, Intrusion Detection System, SQL Injection, User behavior.

1. INTRODUCTION

Internet has become a vital part in day to day life for wide-ranging population and used for many purposes like business transactions, instructive purposes etc. Since many organizations rely on online business transactions there is a need for their database to be secured from intruders.

The use of the internet for accomplishing important tasks such as transferring a balance from bank account always comes with a security risk. Today's websites strive to keep their user's data confidential and after years of doing secure business online, these companies have become experts in information security. The database systems behind these secure websites store non-critical data along with sensitive information, in a way that allows the information owners to quickly access the information while blocking break-in attempts from unauthorized users. A common break-in strategy is to access sensitive information from a database by first generating a query that will cause the database parser to malfunction, followed by applying this query to the desired database. Such an approach that gains access to private information is called SQL injection.

Since databases are everywhere and are accessible from the internet, dealing with SQL injection has become more important than ever. Although current database systems have little vulnerability, the Computer Security Institute discovered that every year about 50% of databases experience at least one security breach. The loss of revenue associated with such breaches has been estimated to be over four million dollars.

An intrusion can be defined as any set of actions that attempt to compromises the integrity, privacy, confidentiality or accessibility of resources of the system. An Intrusion detection system aims to identify an intruder breaking or misusing system resources.

A. Intrusion Detection Approaches

1) Misuse Detection: It is based on the knowledge of vulnerabilities and known attack signatures. Misuse detection is concerned with detecting intruders who are attempting to break into a system by using some known vulnerabilities. Signature based IDS stores patterns of Known attacks. It uses stored behavior patterns to identify and detect attacks. It can detect only known attacks. The main drawback of Signature based IDS is that it cannot detect new attacks or previously unseen attacks

Anomaly Detection: Anomaly detection assumes that intrusions will always reflect some variation from normal pattern. This type of IDS stores normal behavior of system (using previously seen behavior). It is used to classify any behavior that violates it as attacks. Anomaly based IDS detects new attacks but it produces false alarm for legitimate but previously unseen system behavior which is termed as false positives.

Distributed Denial of Service (DDoS) attack is a malicious attempt to create a server or the resource of network unapproachable to its users, generally by interrupting provisionally and by preventing a computer to access internet resource. Application layer attack includes Slowloris, DDoS attacks that target Apache, Windows security systems and more. Including the legal and unknowing requests, these attacks targets web server crashing and that's magnitude could be measured in requests per second.

Honeypot systems are system's setup to gather information about an attacker or intruder into the system. A Honeypot is designed to catch would be attackers before they invade the real servers and services. The main idea of Honeypot is to setup an attractive system that appears to have some vulnerability for easy access to resources. Honeypots are setup not to capture the attacker but to monitor and learn from their actions, then find how people probe and exploit the system and how those exploitations can be prevented.

In this paper, we propose a framework, which accurately detect attacks in multi-tier web application. We present a novel Intrusion Detection System based on analyzing user behavior and by analyzing the user input queries using Honeypot technique to reduce false positives in Dynamic web application**.** Honey pots are designed to check the activity of intruders, save log files and record events. By collecting such data, the Honeypots work to improve security. By monitoring both web requests and subsequent database queries, we are able to detect attacks that an independent Intrusion Detection System would not be able to identify.

The routine models of peerless user sessions with both the front-end HTTP requests and back-end database queries are done in this approach. In order to implement this, a fragile virtualization methodology is implemented to allot web session to a container, and unique virtual environment for each user. Unique ID is used for every container to collaborate the web request with the ultimate DB queries. IDS uses predefined knowledge to identify an attack, it doesn't have the capability to identify any new attacks, but a

Honeypot gives a real-time approach on how the attack had happened, through which it will be possible to strengthen the security.

2. RELATED WORK

Polygraph [11] is a signature generation system that produces signatures that match polymorphic worms. To protect multi-tiered web application, Intrusion detection systems have been commonly used to detect known attacks by matching misused signatures or traffic patterns. [9] Proposes Signature Generation Algorithm which defines polymorphic signature generation problem. It proposes different classes of signature suitable for matching polymorphic worm payloads and creates signatures in these classes. Signature classes for polymorphic worm includes

□ Token Subsequence Signatures

□ Bayes Signatures

Run Signature Generation algorithm on workloads consisting of samples of the polymorphic worms to evaluate the quality of the signature produced by the algorithm and evaluate the computational cost of signature generating algorithm. The Evaluation of signature generation algorithm on a range of polymorphic worms exhibits low false negatives. The main drawback of Signature based Intrusion Detection System is that it uses a database of previous attacks. It cannot detect new attacks or previously unseen attacks.

Anomaly Detection of Web based Attacks [3] presents an Intrusion detection system that uses a number of anomaly detection techniques to detect attacks against web servers. Statistical analysis on historical data has been performed and pattern models on that are produced. This paper introduces a novel approach to perform anomaly detection using HTTP queries containing parameters. The parameter characteristics such as length, structure are earned from input data. The system correlates the server side programs with client queries with the parameters contained in the queries. The main drawback of Anomaly based detection is that they detect deviations from the learned patterns of the user behavior. It can detect new attacks but it produces false alarm for legitimate but previously unseen system behavior.

Parse Tree Validation is used to detect and prevent SQL Injection attack [6]. This technique is based on comparing the parse tree of the query before including user's input with the result after including input at run time. By incorporating a simple SQL parser, it evaluate all user input without requiring a call to the database, thus reducing runtime costs. This aims to satisfy the following 3 criteria:

- Eliminate the possibility of attack
- Reduces the effort essential for the programmer
- Minimize the run time overhead.

A Parse tree is the Data Structure built by the developer for the parsed representation of a statement. In this method, by parsing two statements and comparing their parse trees, we can check if the two queries are equal. When attacker successfully injects SQL attack into the Database, the parse tree of the intended SQL query and the SQL query which was generated by an attacker differs [6]. This method reduces the effort required by the programmer, because it captures both the planned query and the actual query, comparing them and throwing an exception when appropriate[15] In previous work a common-purpose framework that employs the power of a fragile virtualization to track applications interactions in a feasible manner has been discussed. The main goal is to use this framework for

- Application auditing
- Intrusion detection
- System recovery from attacks.

Each Virtualized Environment is constructed in a innovative way that reduces the scope of application events that need to be followed to protect integrity and interactions among Virtualization and system resources like a network and a system memory. The framework termed Journaling Computing System (JCS) makes the following contributions

- Model system events as transactions. All interactions among virtualized application's executions and interactions with remote hosts/servers need to be monitored.
- Isolating and Monitoring Virtual Environments.

In JCS, Summarizations preserve history for very long periods of time. So it occupies high disk space. The client-side XSS filter should be placed between the HTML parser and the JavaScript engine, instead of mediating between the network stack and the HTML parser [4]. So this design achieves high performance and high fidelity. This post-parser design examines the semantics of HTTP response, as interpreted by the browser, without performing a time consuming and error-prone simulation. The usefulness of the filter depends on what percentage of vulnerabilities the filter covers and the rate of false positives and false negatives. This new filter design block suspected attacks by preventing the injected script from being passed to the JavaScript engine rather than performing risky transformations on the HTML.

A server roaming scheme has been proposed in [12], where a model of the scheme is evaluated, and has been studied, and also simulated. The honeypot's location is not constant and is roaming within a pool of server hence its made difficult for attackers to track their traffic from the honeypots and detection of honeypots is avoided. The scheme allows only k out of N series to be concurrently active where as the remaining N-K act as honeypot. The location at the current active server and the honeypot are changed according to a pseudo-random schedule shared among the servers and legitimate client. Therefore legitimate clients always send their service request to active servers whereas attack request may reach the honeypot. The source address of any request that hits a honeypot is black listed so that all future requests from this source are subsequently dropped. The source address is not blacklisted unless a full service handshake is recorded to ensure that it is not spoofed.

In this paper, we propose a framework, which accurately detects attacks in multi-tier web application. We present a novel Intrusion Detection System based on analyzing user behavior and the user input queries using mapping model algorithm to reduce false positives in Dynamic web application**.** Honey pots are designed to check the activity of intruders, save log files and record events. By collecting such data, the Honeypots work to improve security. By monitoring both web requests and subsequent database queries, we are able to detect attacks that an independent Intrusion Detection System would not be able to identify.

Our approach can create normality models of isolated user sessions that include both the front-end HTTP requests and back-end Database Queries. To achieve this, we employ a lightweight virtualization technique to assign each user's web session to a committed container, an isolated virtual environment. Here, we use ID of the container to accurately associate the web request with the subsequent DB queries. IDS uses predefined knowledge to identify an attack, it doesn't have the capability to identify any new attacks, but a

Honeypot gives a real-time approach on how the attack had happened, through which it will be possible to strengthen the security.

3. STATE OF ART

Honeypot is a unique system that is connected to the organization network in order to attract the attackers, to connect with them and learn their behavior, through which it is possible to identify any kind of new attacks. Furthermore, it can be used to monitor behavior of an individual who has gained access to the Honeypot. Honeypots are a unique tool to learn about the policy of hackers to compromise the system security.

Intrusion Detection System can be used as an extension of a Honeypot for improving storage capabilities. The concept involved in Honeypot is that any packet or any traffic route to the Honeypot system is assumed as a suspect for an attack. A system administrator cannot find any fault or attack sensation in his/her organization, then he/she may be satisfied with the security they have for their organization, but by using Honeypot we can obtain the recorded information about an attack, which the firewall failed to detect.IDS uses predefined knowledge to identify an attack, it doesn't have the capability to identify any new attacks by the blackhats, but a Honeypot gives a real-time approach on how the attack had occurred, through which it will be possible to strengthen the security.

4. PROPOSED METHOD

4.1 Building Honeypot

Set up a server and then fill it with attractive files. Build this setup hard but not impossible to crack into. After that sit and wait for the attackers to show up. Monitor them as they gambol around in the server. Record their interactions and study them like watching insects under a magnifying glass. The Honey pot system should appear as common system.

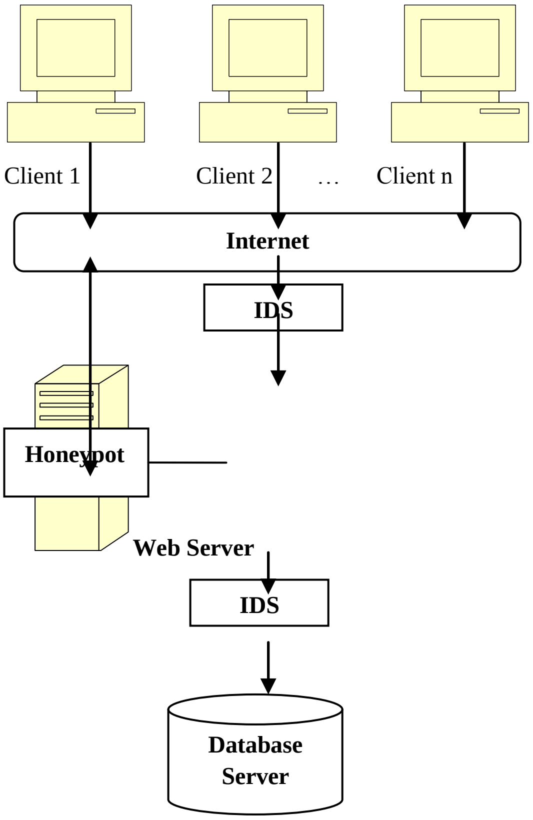


Fig 1: Building Honeypot

4.2 Traffic Collection

The Client sends request to the Web Server through Web browser. Then corresponding Queries are generated which are then transferred from web server to Database server. Client gets response from Database Server through Web Server. All the traffic to a Honeypot should be considered suspicious. Honeypots are designed to review the activity of intruder, save log files and record events. By gathering activity of intruder, the Honey pots work well to improve security.

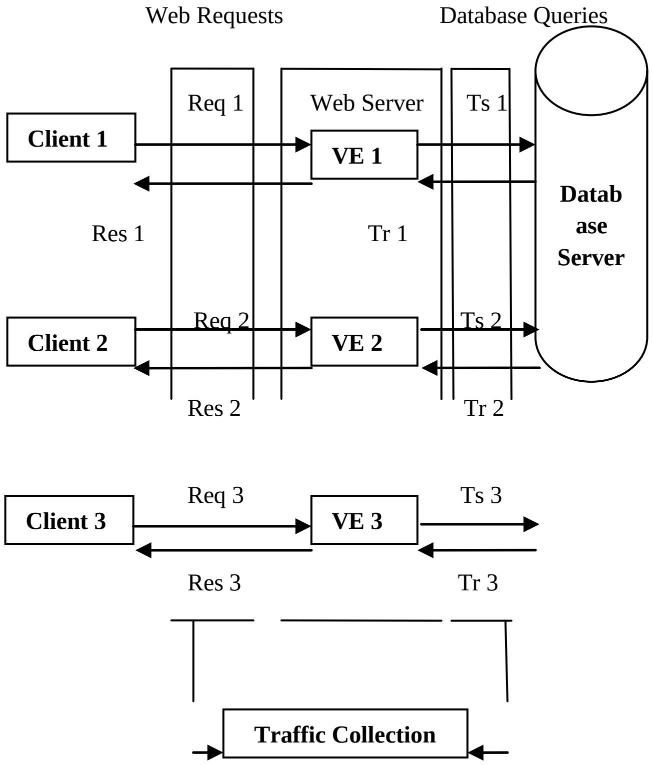


Fig 2: Traffic Collection

4.3 Attack Scenarios

4.3.1 SQL Injection attack: SQL injection is one of the most common types of attack in web connected Databases. Attacker inserts an unauthorized SQL statement through SQL data channel. This attack is caused by non validated input parameters. SQL injection attack is one of the most prominent threats today. SQL injection is a security vulnerability that occurs in the database layer of an application.

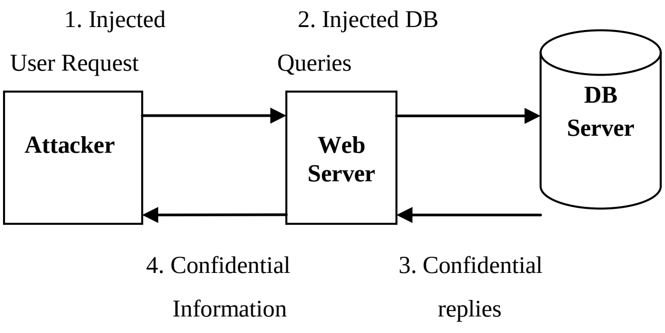


Fig 3: SQL Injection attack

4.3.2 Hijack future session attack: This attack is mainly aimed at the Web Server. An attacker takes over the web server and hijacks all the subsequent legitimate user sessions to launch attacks.

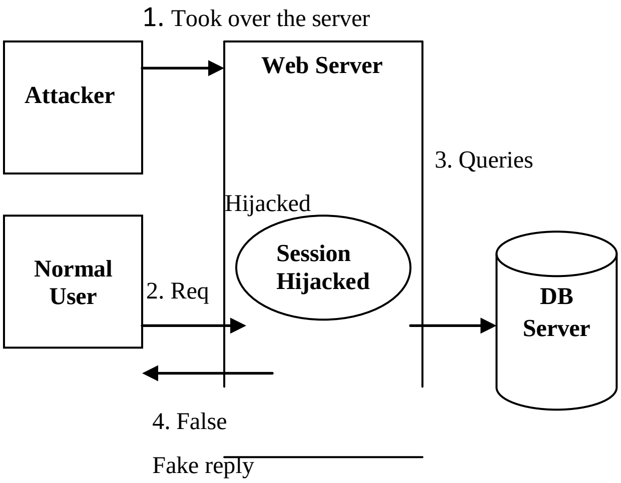


Fig 4: Hijack future session attack

4.3.3 Privilege Escalation attack: Privilege means what a user is allowed to do. Common privileges include viewing files, editing files, deleting files. Privilege escalation means a user takes privileges they are not allowed to.

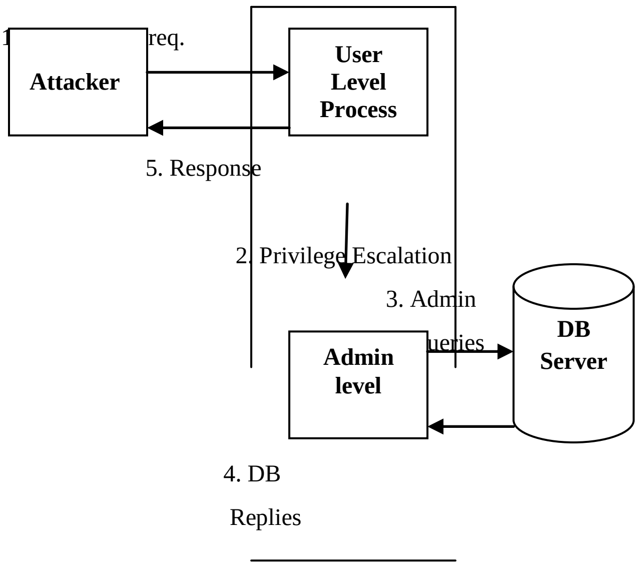
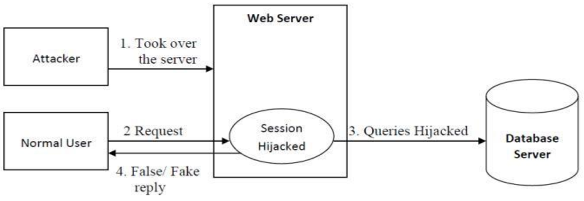


Fig 5: Privilege Escalation attack

4.3.4 Denial of Service attack: DOS attack is a malicious attempt by a single person or a group of people to cause the compromised nodes to deny service to its victim.In Computer world, Honeypot is the technology used to trap the attackers by learning their flow in attacking a system, though there are lot of attacks that can be detected, the distributed denial of service attack cannot be handled efficiently. A distributed denial of service attack uses multiple machines to prevent the legitimate use of a service. For example Stream of packets consuming a key resource, malformed packets confusing an application or protocol, overloading the internet infrastructure.

Thus by identifying these above mentioned abnormalities in the incoming packets, the honeypot will be able to learn the distributed denial of service attack and handle it so efficiently.

In DDoS attack in order to re-assemble the data consequence numbers received by the end system. If one consequence number is missing the end system will not receive the fore coming packets. Hence to serve the web requests, the end-



system waits until it receives the packet with the consequence number for it is waiting. The attacker tries to fool the system by sending two packets with the same sequence number. Among those numbers one containing false data to be accepted by the IDS and not forwarded to the end-system, and the other one containing the attackers' desired data to be accepted by the end system, and skipped by the Intrusion detection system. Thus, whenever it detects two various packets with the same consequence number for one terminal, it considers it as an attack and prevents the load balancer from sending that packet to the end-system. Also it marks the source IP of this packet, as an attacker IP, causing the load balancer to forward all the packets originated from this IP to the Honeypot.

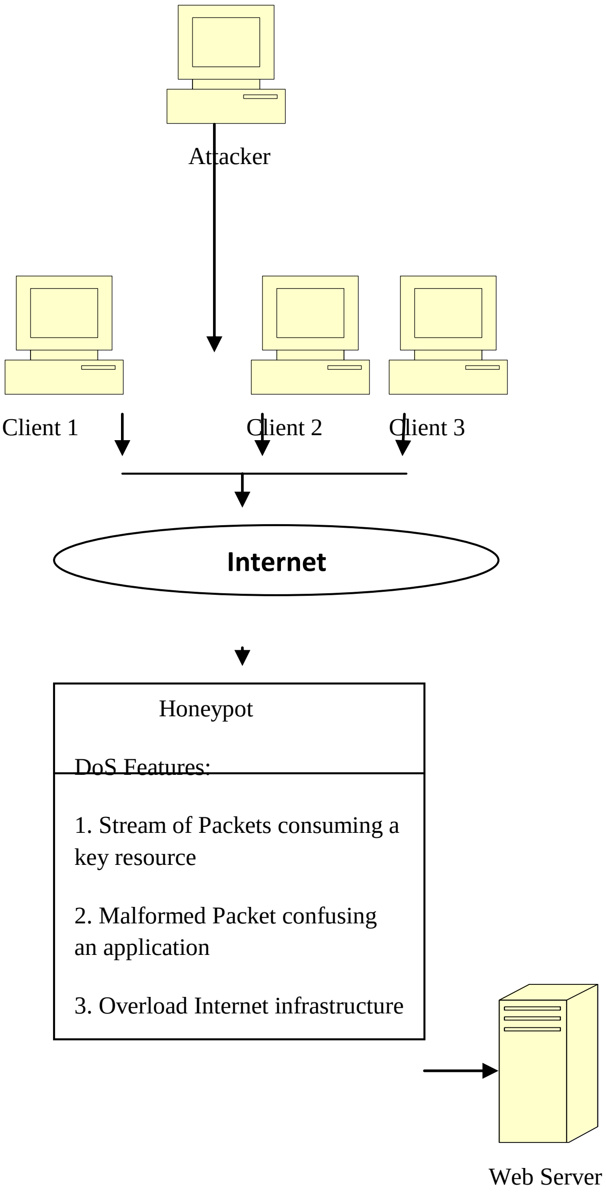

Attacker
Client 1
Client 2
Client 3
Internet
Honeypot
DoS Features:
1. Stream of Packets consuming a key resource
2. Malformed Packet confusing an application
3. Overload Internet infrastructure
Web Server

Fig 6: Handling Denial of Service attack

4.4 Intrusion Detection System

4.4.1 Virtualization Container based Web Server: In our approach, we are utilizing lightweight process containers, disposable servers for client sessions in the Web server. It is possible to initialize thousands of containers on a single physical machine and these virtualized containers can be discarded or quickly reinitialized to serve new sessions. A single physical web server runs many containers. Each one of the container is an exact copy of the unique web server. This approach dynamically generates new containers and recycles used containers. As a result, a single physical server can run continuously. Here Virtualization methodology is used to isolate objects and enhance security.

4.4.2 Session Separated Web Server: Assign each session to a dedicated Web server container. Single user always deals with the same container of the Web server. As each user's web requests are isolated into a separate container, where an attacker can never break into other user's sessions. He cannot hijack other user sessions. Therefore, the legitimate sessions will not be compromised directly by an attacker.

5. EXPERIMENTAL ANALYSIS

This framework was implemented using Apache Web server as front end web server and MySQL as back end database server. Attack tools such as Sqlmap, metasploit are used to launch attacks manually. Honeypot technique works by monitoring the intruder's activities during their use of Honeypot. Honey pots are generally designed to check the activity of intruder, save log files and record events. Using Honeypot, we can obtain the information about an attack like how the attack had happened, through which it will be possible to strengthen the security.

Algorithm Summarization

```
Step 1: Get client request from a browser
Step 2: Associate the client with the session ID Step 3: The
    request is passed to the web server
Step 4: The Honeypot at the web server records the session ID
    and other captured information
4.1 For legitimate user, it allows the request to get
    converted into corresponding queries.
4.2 For suspected traffic (two packets carrying
    same sequence number) using heuristic technique, it
    blocks the intruder.
4.2.1 It checks the IP address
    and MAC address of the packet.
4.2.2 If two packets carry same
    MAC address but different IP address, it blocks.
4.3 Legitimate user identified as an intruder will
    be recognized as an authenticated user after using some
    heuristic technique
4.3.1 By referring cache table,
    the honeypot identifies the legitimate users
Step 5: Web server resolves the web request to corresponding
    SQL query
Step 6: The Database server checks the validity of the query
    With the help of the mapping model available in IDS
Step 7: By this, the system reduces the false positive rate
```

Table 1: Sample request and Corresponding Queries

Request	Date	IP	Query String
Android	2013/Feb 23 10:52:35	192.215.39.110	http://en.wikipedia. org/wiki/Android
Mouse	2013/Mar 05 05:04:12	192.215.39.122	http://en.wikipedia. org/wiki/Mouse
CPU	2013/April 20 08:05:32	192.215.38.140	http://en.wikipedia. org/wiki/CPU

Table 2: Sample request and mismatched response

Request	Date	IP	Query String
Apple	2013/Feb 23 10:53:40	192.215.39.110	http://en.wikipedia. org/wiki/mango
Rose	2013/Mar 05 05:06:12	192.215.39.122	http://en.wikipedia. org/wiki/lotus

6. CONCLUSION AND FUTURE WORK

In this paper, communications are categorized as sessions which identify the mapping between web server request and subsequent DB queries. Using this approach, at database side, we are able to tell which DB transaction corresponds to which client request. This helps us to identify the mapping between web server request and corresponding Database queries. By using this mapping model, we detect abnormal behavior on a session or client level. Because of the isolation property of our container based web server design, an attacker can stay only within a web server container. So that attacker cannot hijack other user sessions. In Computer world, Honeypot is the technology used to trap the attackers by learning their flow in attacking a system, though there are lot of attacks that can be detected, the distributed denial of service attack cannot be handled efficiently. A distributed denial of service attack uses multiple machines to prevent the legitimate use of a service. For example Stream of packets consuming a key resource, malformed packets confusing an application or protocol, overload the internet infrastructure. Thus by identifying these above mentioned abnormalities in the incoming packets, the honeypot can be able to learn the distributed denial of service attack and handle it so efficiently.

Intrusion detection system using low interaction Honeypot technique can be able to detect intrusions more accurately by analyzing the user behavior and by analyzing input queries in order to reduce the false positive rate in Dynamic web application. Honeypot gives a real-time approach on how the attack had happened, through which it will be possible to strengthen the security.

7. REFERENCES

[1] Bazara.I.A.Barry and H.Antony Chan, ”Syntax and Semantics Based Signature Databases for Hybrid Intrusion Detection System,” Security & Communication Networks, vol. 2, Issue 6, Dec. 2009.

[2] Christian Doring, “Improving Network Security with Honeypots,” German Honeynet Project, 2005.

[3] Christopher Kruegel and Giovanni Vigna, “Anomaly Detection of Web Based Attacks,” Association for Computing Machinery Conference. Computer and Comm. Security (CCS ’03), Oct. 2003.

[4] Daniel Bates, Adam Barth and Collin Jackson, “Regular Expressions Considered Harmful in Client Side XSS Filter,” Proceedings of the 19th International Conf. World Wide Web, 2010.

[5] Giovanni Vigna, Willam Robertson, Vishal Kher, “A Stateful Intrusion Detection System for World Wide Web Servers,” 19th conference of the Computer Security Applications on the year 2003.

[6] Gregory T. Buehrer, Paolo A. G. Sivilotti and Bruce W. Weide, “Prevent SQL Injection Attack by Validating Parse Tree,” Association for Computing Machinery 2005.

[7] A. Harrison, the DoS attack Aftermath, http://www.cnn.com/2000/TECH/computing/02/14/ dos.aftermath.idg, 2000

[8] Hellman M.E Diffie W., An Introduction to Cryptography, Volume 67, Pages 397-427, Proc. IEEE, 1999.

[9] Herv Debar, Marc Dacier and Andreas Wespi, “Towards a Taxonomy of Intrusion Detection System,” Computer Networks, volume 31 in the year 1999.

[10] HoneyNet Project, http://project.honeynet.org/

[11] James Newsome, Brad Karp and Dawn Song, “Polygraph: Automatically Generating Signatures for Polymorphic Worms,” Proceedings of the Security and Privacy, IEEE Symposium on May 2005.

[12] S. M. Khattab, C. Sangpachatanaruk, D. Moss´e, R. Melhem, and T. Znati, “Roaming Honeypots for Mitigating Service-level Denial-of-Service Attacks,” In ICDCS, 2004

[13] H.A.Kim and Brad Karp, “Autograph: Toward Automated Distributed Worm Signature Detection,” Proceedings of the 13th Conference, USENIX Security Symposium on the year 2004.

[14] Vicktoria Felmetsger, L.Cavedon, Christopher Kruegel and Giovanni Vigna, “Towards Automated Detection of Logic Vulnerabilities in Web Applications,” Proceedings of the USENIX Security Symposium Conference on the year 2010.

[15] Yih Huang, Angelos Stavrou, Aup K.Ghosh and Sushil Jajodia, “Efficiently Tracking Application Interactions Using Lightweight Virtualization,” Proceedings of the First Workshop on Association for Computing Machinery, Oct. 2008