INTERNATIONAL JOURNAL OF RESEARCH IN COMPUTER APPLICATIONS AND ROBOTICS
ISSN 2320-7345

# A SOLUTION TO PHP CODE INJECTION ATTACKS AND WEB VULNERABILITIES

**Venkatesh Yerram[1], Dr G.Venkat Rami Reddy[2]**
*[1]Computer Networks and Information Security, venkatesh.yerrams@gmail.com*
*[2]Computer Science Engineering, 2nd gvr_reddi@yahoo.co.in*
*JNTU Hyderabad, India*

## Abstract
Over the decade web applications are grown rapidly. This leads to cyber crimes. Attacker injects various scripts to malfunction the web application. Attacker injects these scripts to text box of vulnerable web application from various compounds such as search bar, feedback form, login form etc and later which is executed by the server. Sometimes attacker modifies the URL to execute a successful attack. This execution of system calls and API on web server by attacker can damage the file system and or leaks information of web server. PHP is a server side scripting language, dynamic features and functionalities are controlled through the PHP language. Hence, the use of PHP language results in high possibility of successful execution of code injection attacks. The aim of this paper is first to understand the code web application vulnerability related to PHP code injection attack, the scenario has been developed. Secondly defeat the attack and fast incident determination from the developed domain dictionary. This proposed system is helpful for cyber forensics expert to gather and analyze the evidence effectively

**Keywords**: Code Injection, vulnerabilities, Attack, cyber forensics

## 1. INTRODUCTION

The web environment is growing rapidly day by day, the cyber crimes also increasing rapidly. The web applications are growing in areas like banking business, health care, educational, E-Commerce and other critical infrastructure. But due to the adhoc nature of web application development and due to design complexity it is difficult to attain foolproof of web application security. In order to provide security awareness for web application security Open Web Application Security Project (OWASP) [1] was formed in 2001. OWASP is a nonprofit organization and it is an open community, the main function is to finding and fighting the insecure software. For every three years OWASP rewrites the most common web application security vulnerabilities.

Web applications are frequently deployed with the critical software bugs that may be maliciously exploited. These applications are widely exposed that existing vulnerabilities are probably uncovered and hacker use this to exploit the application. To avoid these vulnerabilities, developers need to apply best coding practices, perform general security reviews of the code, use penetration test tools, use of code vulnerability analyzer, database analyzer etc. but many times developers more focus on the implementation of functionalities and more concern about user requirements and disregards the security features. Numerous developers are not specialized in the security aspects and common time to market constraint limit an in depth test for security issues.

The code injection attacks are common type of web application vulnerabilities, which occurs due to improper handling of users input. The common type of web application vulnerabilities are SQL-Injection, Cross Site Scripting (XSS).

To conduct SQL-Injection attacker injects malicious string, mostly it is a database query, into an available web forms and that is executed by database. The query inserted by attacker may damage the database,

eventually sensitive information, may lost, may alter the sensitive information, the users information will be lost. OWASP has rated injection attacks are first in web application vulnerabilities in 2010 and 2013.

To conduct Cross Site Scripting (XSS) attack an attacker may insert logical script code into available web forms which then reads and display the current cookie values or redirect the user to another web page. Cross Site Scripting (XSS) attack is ranked first on the OWASP Top 10 Web application vulnerabilities in 2007 and third in 2010 and 2013.

## 2. RELATED WORK

Many researchers are working on these attacks and suggest some methods to defeat these attacks. The Open Web Application Security Project (OWASP) [1] is working in this security area and suggesting many solution to the security vulnerabilities

OWASP suggested different techniques to defeat the SQL-inject. One solution to avoid SQL-injection is use of prepared statement. This prepared statement queries are forced to define all the code and pass parameters to query later, this style of coding allows database to differentiate the code and data, and regardless of what user input is given [3].

The second solution to defeat the SQL-injection is stored procedure, which have the same effect like prepared statement. This type of technique the developer needs to define SQL code first and then after the parameters will be passed. The difference between prepared statement and stored procedure in this SQL code for stored procedure defined and stored in database itself, and which is called from application [3].

The following are the rules proposed by the OWASP to prevent the cross site scripting attack. This rules do not allows the attacker a absolute freedom of putting the untrusted data into the forms. These rules cover the vast majority of common type of situations. Here no need to allow the all rules for application some applications needs only first and second rules in their application.

- Never allow the Untrusted Data in your application Except in Allowed Locations.

- Escape the HTML characters before allowing Untrusted Data into HTML Element Contents.
- Escape the Attribute Before allowing Untrusted Data into HTML Common Attributes
- Escape the JavaScript Before allowing Untrusted Data into JavaScript Data Values
- Escape the CSS and Validate Strictly Before allowing Untrusted Data into HTML Style Property Values.
- Escape the URL Before allowing Untrusted Data into HTML URL Parameter Values.
- Design a library to Sanitize HTML Markup characters.
- Avoid the DOM-based XSS

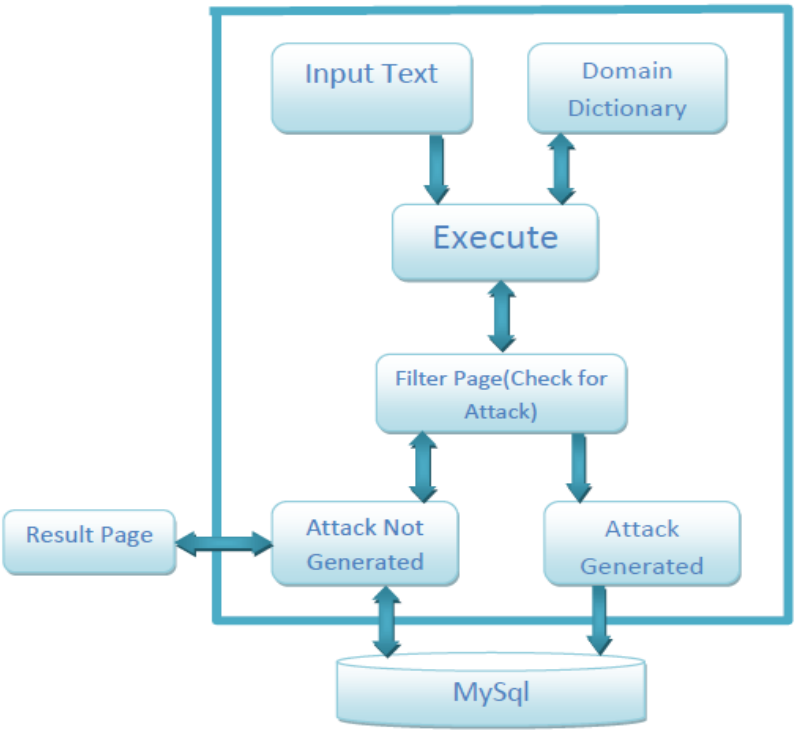
## 3. DESIGN OF NEW PROPOSED MODEL

Many methods are proposed to overcome these vulnerabilities in spite of this still many web application are facing the loss sensitive information, server configuration exposal, file vulnerabilities, to overcome such vulnerabilities in this proposing a design which avoid the execution of attacker script and which shows the solution for the above problems.

The proposed model has the input text, which is collected from the HTML pages for example text box, text area etc which is used in collecting user information like username, passwords, feedback, form, guest book etc, and then followed by execution. In this stage the data which is collected is transferred to the filter page, this page then check for the any attack string is present in the inputs inserted. If any such strings present in the input text such strings will be foreword to the attack generated stage, this will further verify the string and try to collect the attacker information who try to attack on the system and further these information is stored in database to analyze further in future. And this string which contains the malicious strings will be filtered and such strings not inserted in database, to avoid the damage to database. If the input string doesn't have any malicious string or vulnerable

string then such request is foreword to the no attack generated page, this page further verifies the input and then serves the user request. It communicates with the database to fulfill the user requirements. For next request onwards it communicates with the execution stage and filter page to validate the user inputs.

In this paper the solution to SQL Injection attack, Cross Site Scripting, Shell Injection Attack, Local File Read, Local vulnerability and Local File Write Vulnerability is provided.




## 4. IMPLEMENTATION

### 4.1 Shell Injection Attack:

Shell injection, also known as command injection is one of the most critical web application vulnerability, which allows the attacker to execute unwanted system commands and gain unauthorized access to user's information. An OS command injection attack occurs when an attacker attempts to execute system level commands through a vulnerable web application.

To understand how the most of shell injection attack works consider a PHP script which takes the input from the user and checks for the reach ability of the host in the internet using the internet protocol (IP).

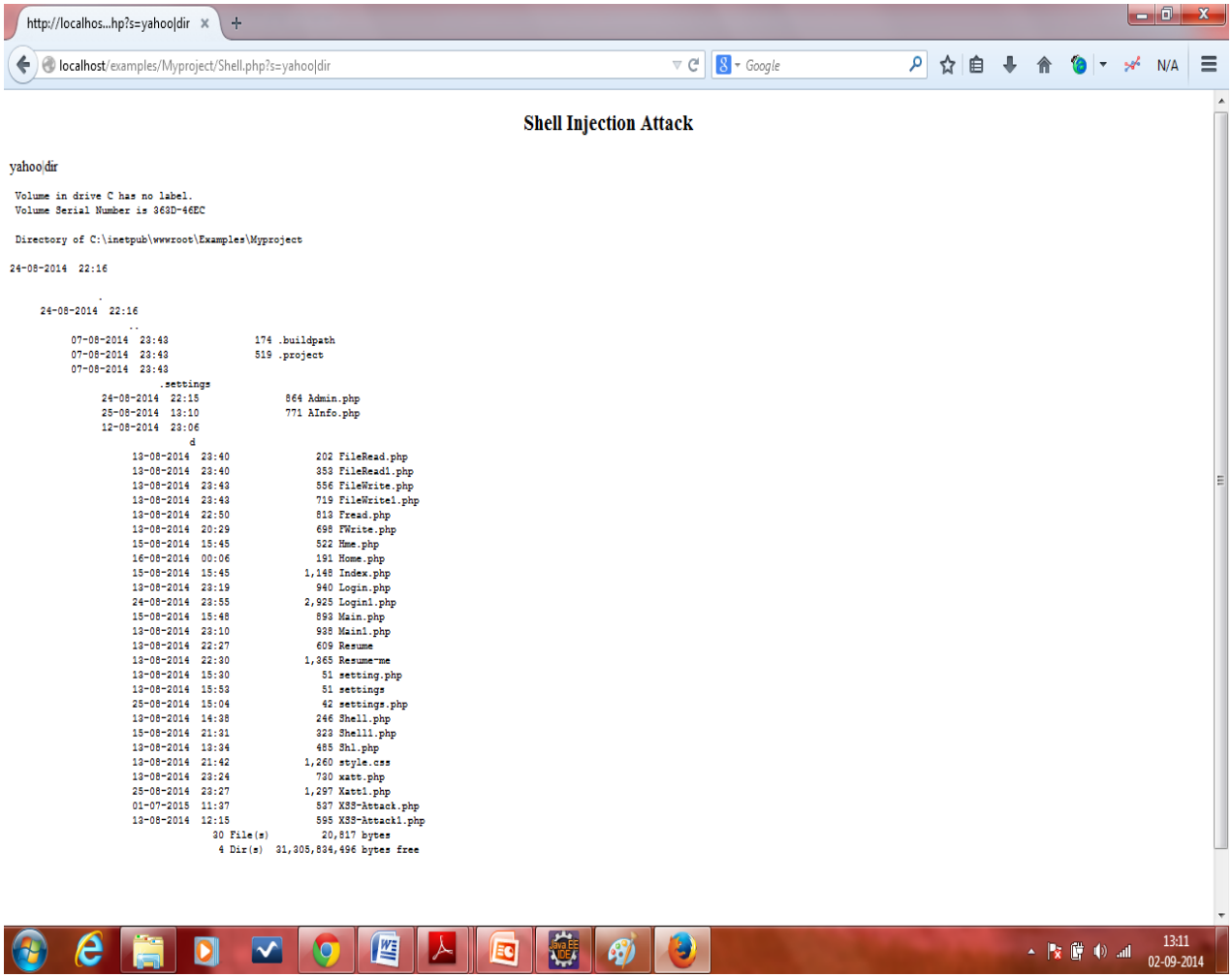
```
<?php
$q = "ping " . $_GET ['domainName'];
$out = shell_exec($query);
echo "<pre>$out<pre>";
?>
```

Now suppose an attacker enters the following GET request:

http://localhost/test.php?domain= www.yahoo.com|dir---(1)

In the above URL "|" or "&" is used to append the command onto execute this command at the time of execution.

The above example was executed in windows7 operating system, in the URL "www.yahoo.com|dir" means here




yahoo is domain name and dir is windows command which outputs the entire directory structure of a directory. For the above example with the URL (1) give the following output. This clearly shows the shell injection attack the list of files and directories are clearly prompted to attacker.

Figure 1: Shell Injection Attack

To avoid shell injection attack here in this paper the proposed system will do the operations and that leads to avoid the shell injection attack. In this proposed system the system will take the input from the text box which has the name 'domainName', these input is forwarded to the execution page, this page checks for the malicious code and script here in this shell injection attack the filter page checks for the malicious strings like ampersand ('&'), slash or vertical bar ('|') and backslash (''), if any of this malicious strings are present then this input is forwarded to attack generated page, this page further verifies the vulnerable string and then this information regarding the attack will be stored in the database. This data is

Available for the future use and if there are no malicious strings their then that will be foreword to no attack generated page, these page will serve the user request and which further verifies the input with the filter page serves to the user.
Finally, complete content and organizational editing before formatting. Please take note of the following items when proofreading spelling and grammar.

### 4.2 File Read Vulnerability

In order to read a file from the local file system PHP has a
build in function called 'file_get_contents'. This function is used to get the file contents and these contents will be displayed by echo method. Though this methods have the advantages but this method suffers from the remote

file inclusion, because of these the web application is used to load the remote file if this remote file contains any malicious code, such code will be executed by the server. If the remote file is executed in the server attacker know that the web application is suffering from remote file inclusion. Then attacker includes the file which contains the malicious script and steals the server information, configuration etc and attacker can also this vulnerability to make the server busy in serving the attacker request then here it may be used like denial of service for the other legitimate users.

To understand this vulnerability clearly assumes a web application which opens and reads file in a local directory. This following code opens the file using the fopen() and displays the file content using file_get_contents() and after the file will be closed by fclose() function.

```
<?php
$f_handle = fopen("$_get[‘filename’]", "w");
while(!feof($f_handle))
{
echo file_get_contents ($_get[‘filename’]);
}
fclose($f_handle);
?>
```

Above code works well if local file is included and it shows the error message if the file not present in directory. If attacker makes the following request

http://localhost/f_read.php?filename=http://www.yahoo.com

then file_get_contents function will open the http://www,yahoo.com the output of above URL is

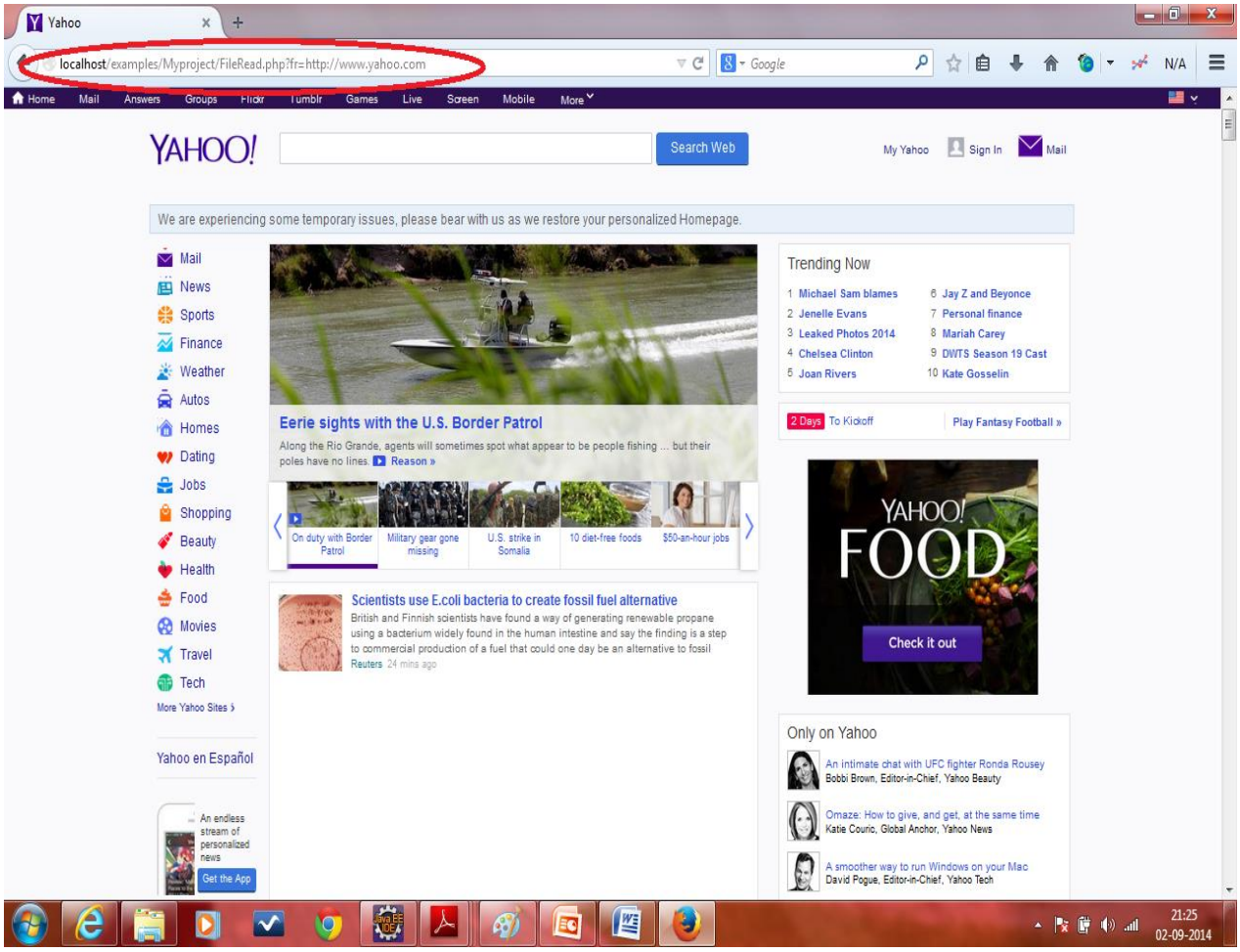



Figure 2: File Read Vulnerability

In order to avoid this vulnerability above proposed model will checks for any malicious script from the input text, from this text if any vulnerable string is present then such input is blocked and doesn't allow executing, if input text is doesn't consist malicious script which allows executing. In this file read vulnerability the filter page checks for malicious script are the input string contains the strings like http, https URL and file path traveler etc.

### 4.3 File Write Vulnerability

PHP has a built function fwrite(), this function is used to write the file contents into the file. But this function won't checks for the malicious script which allows the PHP, HTML, javasxript to insert into a file. After inserting this while opening this content the PHP, HTML and JavaScript will be executed, this leads to vulnerability.

```
<?php
$filename = 'settings.php';
$user = $_GET['fw'];
if (!$handle = fopen($filename, 'w')){
  print "Cannot open file ($filename)";
  exit;  }
 $s=fwrite($handle, $user);
 echo "Number of Characters Written are ".$s."<br>";
if($s>0){
  echo "Write Successful<br>";
   }
   else{
            echo "Write failed";
   }
 ?>
 <?php
```

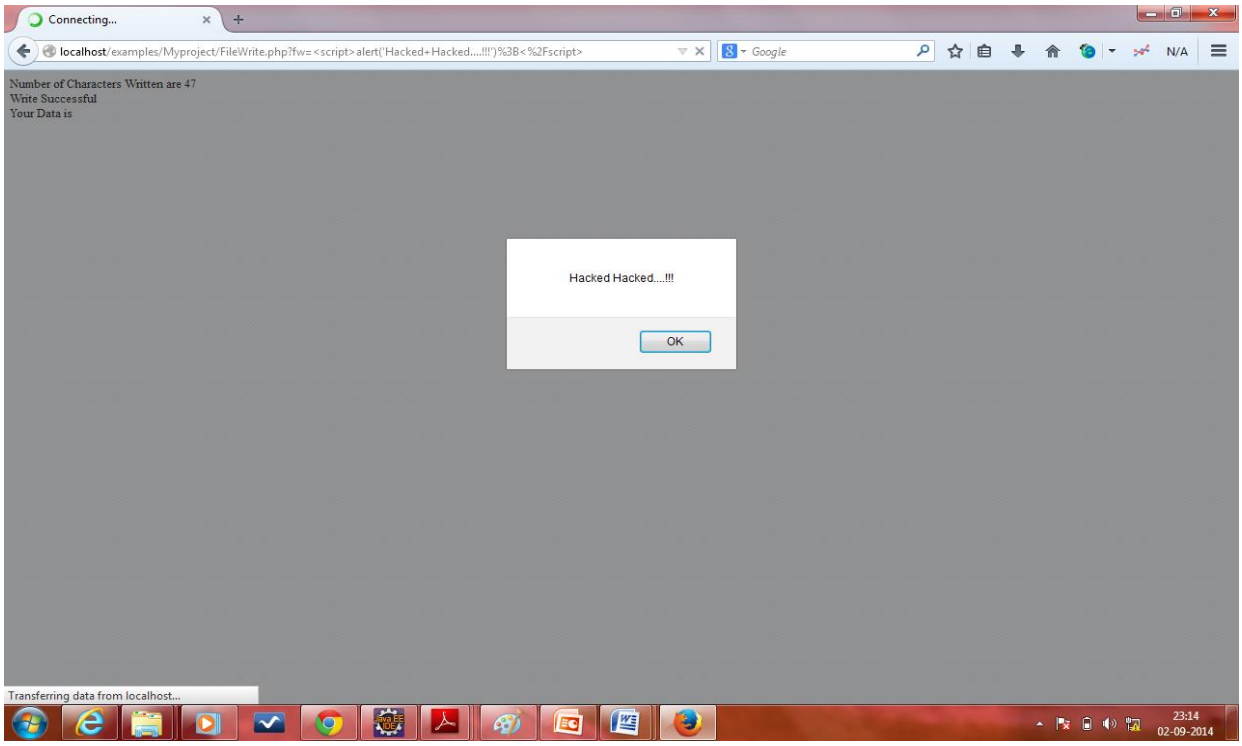



Figure 3: File Write Vulnerability

The above program takes the input from the user and stores in the settings.php, it will works normally if the user enter normal trusted data, if the user enters any malicious script like PHP or JavaScript then while opening the content definitely it will not open normally, it executes the code and that leads to vulnerability by observing

this attacker enters different string and he gains the access to web application. For example if the attacker enters following string.

http://localhost/f_Write.php?fw =”<script>alert (‘Hacked Hacked....!!!’); </script>”

The output of the above program with the above URL is shown in Figure 3.

In order to avoid this vulnerability the above proposed design will first checks for the input text from the user input and if this text contains any malicious scripts like HTML, JavaScript or PHP code then that code will be removed and plain text will added to the file. And the attacker information who wants to inject this code will be stored into the database. And if the user supplied input text doesn’t have any malicious string then such data directly written to the file.

## 5. WEB FORENSICS ANALYSIS AND PHP CODE INJECTION ATTACK

Web forensics is a use of science and technology to investigate and establish a fact from that and a decisive action against it. The objective of the forensics is to discover and prove the digital evidences in order to prove that there is a criminal incident activity in cyber world. In web forensics logs are important but this are initially created for troubleshoot purpose only and which are not purposefully designed for web digital forensics. The present web servers will not tell that is an attack successful or unsuccessful and which doesn’t tell how the system damaged extend. The available logs are also incapable of record the code injection by the user in to the input text fields. This is an important source to collect the evidences in the code injection attacks. Therefore a tool is proposed, designed and developed for capturing the code which is fired by the attacker/user. Web forensics analysis of the PHP code injection attacks and web vulnerabilities takes place in two phases. Evidence gathering phase this collects evidence from developed logging system. Another phase is analysis of gathered evidences.

### *Evidence Gathering for code injection Attack’s and Web Vulnerabilities*

In web environment malicious code will be injected in three entry points. Normal web application forms are used for entering particular information, by a string query which is passed through a URL and by modifying the HTTP user agent.

### *Evidence Analysis code injection Attack’s and Web Vulnerabilities*

The end user activity log has been developed HTTP log has been classified into two types. Legitimate activities are which do not belong to the attacker. Attacker activates which are suspicious. Manual tagging of this information is not possible because enormous of data will be generated if the user submit a request to a server and it is consuming. Hence there is a need of automated system which effectively captures the vulnerable strings and code. In the normal logging system type and nature of actively will not get clearly by looking into the captured log therefore an evidence tagging system through a domain dictionary developed. This will be helpful in identifying and separating the code injection attack from the rest.

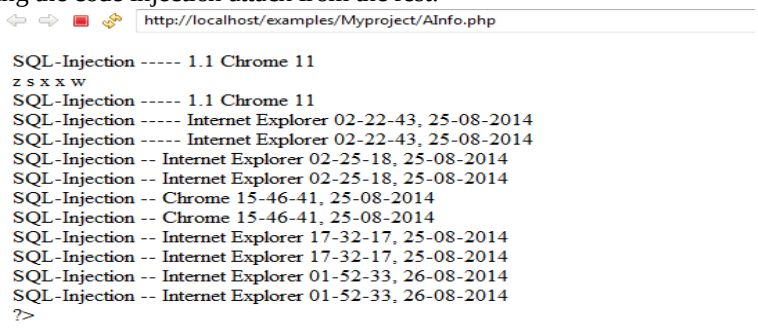



Figure 4: Evidences

## 6. RESULTS

This paper has attempted to demonstrate the some of the potential dangers which are associated with the PHP code injection by developing the three attack scenario. The effects of these three scenarios are shown in figure 1, figure 2 and figure 3. Due to space restriction the intermediate results aren't showing. The developed system is successfully capturing the PHP code injection activities of the attacker as shown in figure4.

## References

[1] OWASP
https://www.owasp.org/
[2] SQL-Injection
https://www.owasp.org/index.php/SQL_Injection
[3] SQL_Injection_Prevention_Cheat_Sheet
https://www.owasp.org/index.php/SQL_Injection_Prevention_Cheat_Sheet
[4] Cross-site Scripting (XSS)
https://www.owasp.org/index.php/Cross-site_Scripting_(XSS)
[5] XSS (Cross Site Scripting) Prevention Cheat Sheet
https://www.owasp.org/index.php/XSS_(Cross_Site_Scripting)_Prevention_Cheat_Sheet
[6] Command Injection
https://www.owasp.org/index.php/Command_Injection
[7] Mirza Mohammed Akram Baig Spring 2012
Security Vulnerabilities in Php Applicatons

## Biography

**Venkatesh Yerram** is pursuing his post graduation from School of Information Technology, JNTU Hyderabad. He has completed B.Tech from CM Engineering College. His research area interests include Information Security, Computer Networks, Operating Systems, Data Mining.

**Dr G.Venkat Rami Reddy** is presently Associate Professor in Computer Science and Engineering at school of Information Technology. He is more than 11 years of experience in Teaching, and Software Development.
His areas of interests are: image Processing, Computer Networks, and Analysis of Algorithms, Data mining, Operating Systems and Web technologies ords.