Proactive Security Framework for Online Business Web Portals with Implementation Details

Hemraj Saini*‡, Sitanshu Mishra**, Prakhar Prateek**

* Department of Computer Science & ICT, Jaypee University of Information Technology, Wakanaghat-173234 INDIA

**Department of Information Technology, Orissa Engineering College, Bhubaneswar-752050 INDIA

‡ Corresponding Author;

e-mail: hemraj1977@yahoo.co.in; hemraj.saini@juit.ac.in

Abstract- Most of the critical information is stored or travelled throughout the Internet and prone to cyber threats all the time. The current manuscript provides a process to develop and implement an automated proactive security framework to alert/avoid such cyber threats for the critical online information. In addition, it also describes a feasibility study towards the adoption of the proposed process by the current user community with favourable results.

The proposed work is able to help for the development of security add-ons to almost all the embedded software applications for the better secured services to the users**.**

Keywords- Cyber defense, SQL injection, Automated Security Framework, Design Methodology, Feasibility Study.

1. Introduction

Related literature explains many of the strategies to secure the information at different levels e.g. layered approach of information security [1, 2, 3, 4], protocol level security [5, 6], database level security [7, 8] etc. but very few concentrates over the explanation and implementation issues of the framework to secure the online information at the time of attack detection through user's involvement [9]. Therefore, the current manuscript focuses over the point to alert or avoid the cyber threats automatically and proactively through the web security portal and user's involvement. Throughout the text a special case of cyber threat i.e. SQL injection [10, 11] has been considered for the detailed explanation about the secured, automated and proactive framework.

The proposed system is about automatic and proactive cyber defense, it means that the cyber system will be defended automatically. This is the technique which can be applied to any of the existing system as an add-on or can be applied in a completely new developed systems based on cyber or web technologies. Proposed system will not only defend the web application, but it will also intimate or communicate the administrator or owner of the application system that the application was trying to be attacked by the attacker.

The system is about making the web applications inaccessible to the attacker. This can be done by tracing the IPs of the attackers and blocking them. Basically, blocking of IPs is nothing but adding the attackers IP address in the database system, therefore, whenever the attacker tries to access the web application through that IP address, the system does not give access to the attacker of web of the web application. Not even a single content out of the main contents other than the blocked message is displayed to the attacker.

The communication or intimation process goes in the manner that the attacker when tries an attack on the system, a communication by automatic

email through the e-mail server and a SMS on mobile is initiated through the SMS gateway through server system. The attacker is trying to attack the web application along with its IP address, therefore, it is sent as the main content of the intimated e-mail and SMS to the administrator.

After few number of defined attempts (default 3 attempts), the system will automatically add the attacker's IP address in the Block IP address list in the database and also a final communication will be intimated to the administrator so that the web application has automatically be secured from the attacker. Now, whenever intruder next tries to navigate the web application even if he/she will not be able to gain access to any of the main contents of the web application and instead a message is displayed to him/her that the IP address has been blocked. Rest of the paper is organized in different other sections such as section-2 to section-9 illustrate- Design methodology, Dealing with unauthorized login process, SQL injection security (DFD Level-0), SQL injection security (DFD Level-1), SQL injection security (DFD Level-2), security against other attacks, main working (for automatic defense), testing the proposed system respectively and conclusion respectively.

2. Design Methodology

The explanation with implementation of the whole designing process, represented by Data Flow Diagrams (DFDs) at different levels, can be divided into different phases as below-

- Dealing with Unauthenticated Login Process
- SQL Injection Security (DFD LEVEL-0)
- SQL Injection Security (DFD LEVEL-1)
- SQL Injection Security (DFD LEVEL-2)
- Security against other attacks
- IP Blocking (Process from a particular IP)

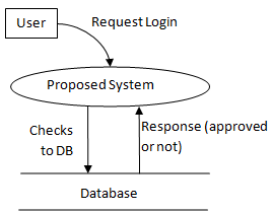

Fig. 1. Authenticated Login.

- Main Working (for Automatic Defense)

3. Dealing With Unauthorized Login Process

Unauthorized login can be possible by many of the ways such as SQL Injection, Phishing request, DDoS, etc. but SQL injection will be of our more concern due to the extreme usage of query systems for online processing. An authorized login process must be having Pre-Conditions like- Username and Password posted to the proposed system and Post-Conditions like- Username and Password checked for SQL injection attack. Both of pre-conditions and post-conditions must be satisfied for the process of handling the unauthorized login attempts. Fig. 1 depicts the authenticated login process and Appendix-1 [line no.35 to line no. 44]

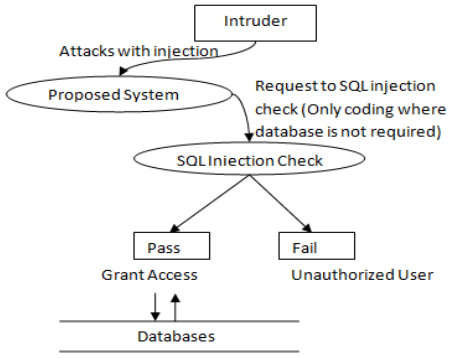

Fig. 2. Security process against SQL Injection Attack.

represents the corresponding segment of code.

4. SQL Injection Security (DFD Level-0)

Some of the user inputs might be used in framing SQL statements that are then executed by the application on the database. It is possible for an application not to handle the inputs given by the user properly. If this is the case, a malicious user could provide unexpected inputs to the application that are then used to frame and execute SQL statements on the database. This is called SQL injection.

Fig. 2 depicts the security process against SQL Injection Attack. The process has two states, first, state-1: the intruder attached the injection for attack and second, state-2: the injection is to be checked. The proposed system has the Pre-Conditions for state-1 like- Username and password remain unchanged where Injection attached by attacker and Post-Conditions for state-1 such as Username and password remain unchanged where Injection attached by attacker.

Pre-Conditions for state-2 are Username string and Password string received from the user, these two strings are to be gone through the process of checking injection individually and if injection string tempered with Stripslashes it has to be detected by the check function. But as the Post conditions for state-2: the tempered string if injection else clean string for further processing.

5. SQL Injection Security (DFD Level-1)

In process to achieve the security against SQL injection, it is required to identify the source IP of the attack. It can be done by IP tracing. IP tracing is done by predefined PHP global variable '$_SERVER" which contains both clients and server's address. In the IP tracing, pre conditions are like- updated records in the database are in consistent state from the respective IP and post conditions are like- updated records in the database must be in consistent state from the respective IP and initiate e-mail or SMS raising process if number unauthorized attempt to access database is increased beyond the threshold as shows in fig. 3.

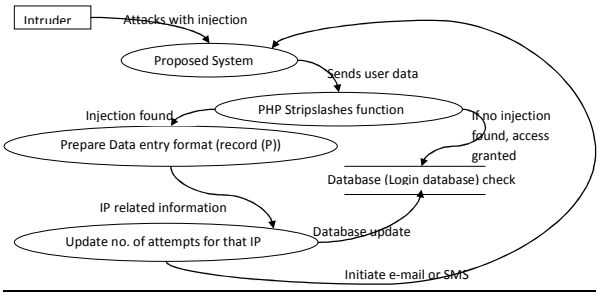

Fig. 3. SQL injection DFD (Level-1)

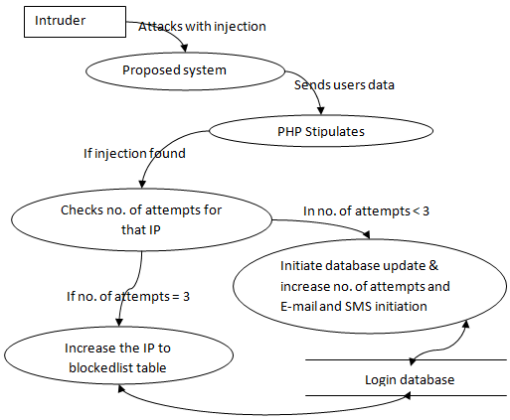

Fig. 4. SQL injection DFD (Level-2)

This can be handled by PHP Stripslashes. Stripslashes are predefined library function in PHP. It checks for unusual characters like- "'" etc. and adds "\" slashes with them and make them useless. Whole process is depicted in the Fig. 4.

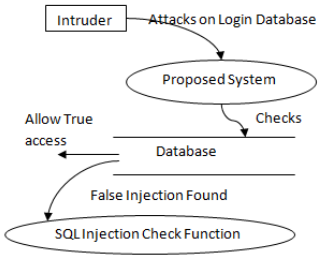

Fig. 5. Other different attacks

Appendix-1 [line no. 47 to line no. 61] and Appendix-1 [line no. 62 to line no. 70] represent the required segment of code for the process.

6. SQL Injection Security (DFD LEVEL-2)

After getting the IP containing SQL injection,

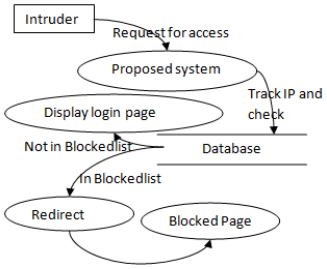

Fig. 6. IP Blocking

the IP has to be blacklisted at the next level depicted in Fig. 5 where number of attempts stands for total number of attempts made for attack by that IP.

7. Security against Other Attacks

Before displaying home or login page to the user, IP is tracked and is checked if present in the blocked list; the system has already secured and will not allow access to home or login page to the user. It is depicted in Fig. 6 and Appendix-1 [line no. 01 to line no. 24] represents the corresponding segment of code. Further Appendix-1 [line no. 79 to line no. 107] and Appendix-1 [line no. 04 to line no. 08] shows the e-mail alert and SMS alert, if attempts to database access are beyond the threshold.

8. Main Working (for Automatic Defense)

Abstract working of the proposed framework is depicted by the Fig. 7. Intruder attacks on a system having the proposed system, through the proposed framework, number of database access attempts is to be checked in addition to blocked IP list. If attempts are beyond the threshold (here, 03) an alert in the form of (email or SMS) has been forwarded and the IP is blocked for the automatic defense.

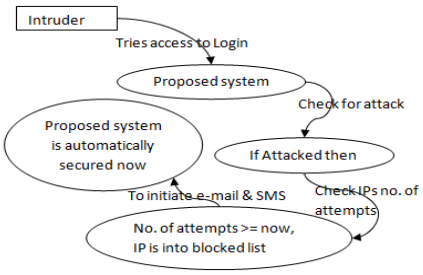

Fig. 7. Automatic Defense (Main Working)

9. Testing the Proposed System

A user Acceptance Testing (UAT) [12, 13, 14] has been carried out to evaluate whether the proposed system/framework is accepted widely among the users or not. UAT is the software testing process where system tested for acceptability & validates the end to end business flow [15]. UAT is executed by client in separate environment and confirms whether system/framework meets the requirement as per the SRS [16] or not. UAT is performed after all the enhancements in the system have been done after a rigorous testing process. It is to be carried out at the final phase of the SDLC [17] but prior to the system being delivered to a live environment.

involves running a suite of tests on the completed system. Fig. 8 depicts the sample test cases used to test the proposed framework under UAT and Fig. 9 shows the position of UAT in a testing suit.

The completion of User Acceptance Testing

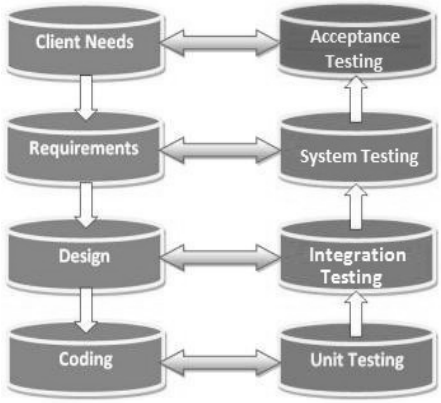

Fig. 9. Position of System Testing in a testing suit

(UAT) is the significant milestone for traditional testing method. The following key deliverable of User Acceptance Testing phase:

Test Plan: Outlines of the Testing Strategy used to test the proposed system.

Browser compatibility:- WorldWideWeb || Mosaic || Netscape Navigator || Netscape Communicator || Internet Explorer || Opera || Mozilla Navigator || Safari || Mozilla Firefox || Google Chrome
Test data set:- set 1 || set 2 ||……|| set n
Screen texts:- readability || spelling || grammar errors || legal and style
Check mandatory fields:- marked with an asterisk || error message when left blank
Check field sizes:- size validation is applied properly || error message when beyond
Check formatting:- ABN/ACN number || phone number with spaces || warn users to omit spaces
Check date fields:- use date picker || delivers the same result as manual entry || error f|| no invalid dates
Check data storage:- truncate || data type || type || key
Check time-outs:- adverse effects are addressed
Security checks:- Configuration management || authentication || authorization || user and session management || data validation || error handling || logging || reporting || coordination and explanation
Security checks (visible inputs) by entering:- positive data || negative data
Security checks (hidden fields) by entering:- positive data || negative data

Fig. 8: Sample test cases used to test the proposed framework under UAT

The User Acceptance Testing is a "black box" tests, means UAT users doesn't aware of internal structure of the code, they just specify the input to the system & check whether systems respond with correct result. UAT users or end users are concentrating on end to end scenarios & typically

- **UAT Test cases:** The Test cases help the team to effectively test the application in UAT environment.

- **Test Results and Error Reports:** This is a log of all the test cases executed and the actual results.
- **User Acceptance Sign-off:** This is the system, documentation, and training materials have passed all tests within acceptable margins.
- **Installation Instructions:** This is document which helps to install the system in production environment.

10. Conclusion

A significant level of proactive defense has been achieved at the user level through the usage of web portal. A complete framework has been discussed with implementation to secure the web resources. Specifically the security against SQL injection attacks is discussed. However, the usage of the proposed framework is not restricted up to SQL injection attack; it can be used to secure the web resources against other attacks. The purpose of the proposed framework is of two fold, first, it provides proactive defense and second, it informs to the user about the unwanted situation.

In addition to the implementation of the proposed framework a User Acceptance Testing (UAT) is also carried out by considering various test cases to ensure the acceptability of the proposed framework among the users. The testing results are favourable and due to the simplicity of implementation the framework must become a successful tool for the proactive defense of the web resources.

References

[1] H. Saini and T. C. Panda, "Extended Cyber Defense Architecture for a University –A Case study", *The IUP Journal of Science & Technology*, 6(2):33-47, 2010.

[2] H. Saini and D. Saini, "Proactive cyber Defense and Reconfigurable Framework of Cyber Security", *International journal named International Review on Computer and Software (IRECOS)*, 2(2):89-97, 2007.

[3] S. Bayat, R.H.Y. Louie, Z. Han, B. Vucetic and Y. Li, "Physical-Layer Security in Distributed Wireless Networks Using Matching Theory", *Information Forensics and Security, IEEE Transactions on*, Volume: 8, Issue: 5, pp.- 717-732, 2013.

[4] E. Harrin, "Taking a Layered Approach to IT Security", Retrieved on 30th April, 2013, Available at: *http://www.esecurityplanet.com/network-security/taking-a-layered-approach-to-it-security.html*

[5] I. Lien, Y. Lin, J. Shieh and J. Wu, "A Novel Privacy Preserving Location-Based Service Protocol with Secret Circular Shift for k-NN Search", *Information Forensics and Security, IEEE Transactions on* , Volume: PP , Issue: 99, 2013, DOI: 10.1109/TIFS.2013.2252011.

[6] H. Saini, K. D. Sharma, P. Dadheech and T. C. Panda, "Enhanced 4-way Handshake Process in IEEE802.11i with Cookies", *International Journal of Information & Network Security (IJINS)*, 2(3), pp. 229~238, 2013.

[7] L. Sankar, S. Rajagopalan and H. Poor, "Utility-Privacy Tradeoff in Databases: An Information-theoretic Approach", *Information Forensics and Security, IEEE Transactions on*,Volume: PP , Issue: 99, pp.-1-15, 2013.

[8] J. Han, W. Susilo, and Y. Mu, "Identity-Based Secure Distributed Data Storage Schemes", *Computers, IEEE Transactions on*, Volume: PP , Issue: 99, 2013, DOI: 10.1109/TC.2013.26.

[9] H. Saini, B. K. Mishra and T. C. Panda, "Computing the Spreading Power of a Business Portal to Propagate the Malicious Information in the Network", *International Journal of Web Protals*, 3(2), 14-22, 2011.

[10] B. Simic and J. Walden, "Eliminating SQL injection and cross site scripting using aspect oriented programming", *Proceedings of the 5th international conference on Engineering Secure Software and Systems (ESSoS'13)*, Jan Jürjens, Benjamin Livshits, and Riccardo Scandariato (Eds.). Springer-Verlag, Berlin, Heidelberg, 213-228, 2013.

[11] V. Shanmughaneethi, R. Yagna Pravin, C. Emilin Shyni and S. Swamynathan, "SQLIVD - AOP: Preventing SQL Injection Vulnerabilities Using Aspect Oriented Programming through Web Services", *A Mantri et al. (Eds): HPAGC 2011, CCIS 169*, pp.-327-337, 2012, Springer-Verlag, Berlin, Heidelberg.

[12] Z. M. Jiang, A. Avritzer, E. Shihab, A.E. Hassan and P. Flora, "An Industrial Case Study on Speeding Up User Acceptance Testing by Mining Execution Logs", *Secure Software Integration and Reliability Improvement (SSIRI)*, 2010 Fourth International Conference on , OI: 10.1109/SSIRI.2010.15 , Page(s): 131 – 140.

[13] L. Yu, W. Di X. Zhao, C. Kong, W. Zhao, Q. Wang and J. Zhu, "Towards Call for Testing: An Application to User Acceptance Testing of Web Applications", *Computer Software and Applications Conference, COMPSAC '09. 33rd Annual IEEE International*, Volume: 1, DOI: 10.1109/COMPSAC.2009.31, Page(s): 166 – 171, 2009.

[14] K. R. P. Leung and W. L. Yeung, "Generating User Acceptance Test Plans from Test Cases", *Computer Software and Applications Conference*, 2007. COMPSAC 2007. 31st Annual International, Volume: 2, DOI: 10.1109/COMPSAC.2007.125, Page(s): 737 - 742.

[15] All about UAT, Retrieved on 30th April, 2013, Available at: *http://www.guru99.com/user-acceptance-testing.html*

[16] C. J. Date, "An Introduction to Database Systems", *Addison-Wesley Professional*, 2003, Ed. 8th, ISBN: 0321197844

[17] Jeffrey D. Ullman and Jennifer D. Widom, "Database Systems: The Complete Book", *Prentice Hall*, 2008, Ed. 2nd, ISBN: 0130319953.

Appendix-I: index.php

```
<?php
if(isset($_GET['info']) && $_GET['info']=="logout")
        {
        session_start();
        session_destroy();
        }
if(isset($_POST['submit']) &&
$_POST['submit']=='Login')
        {
        session_start();
        require('connect.php');
        $ip=$_SERVER['REMOTE_ADDR'];
        $ip2=quote_smart($ip, $sqlinject);

        $query="SELECT * from blockedip WHERE
    ip=$ip2";
        //echo $query;
        $result=mysqli_query($con, $query);
        if($result)
                {
                $nnr=mysqli_num_rows($result);
                if($nnr>0)
                        {
                        header('Location: blockedip.php');
                        }
                }
        $login=0;
        $email=$_POST['email'];
        $passen=$_POST['pass'];
        $email2= quote_smart($email, $sqlinject);
        $pass2= quote_smart($passen, $sqlinject);
        $quer="select * from userlogin where
    email=".$email2." and pass=".$pass2;
        //echo $quer;
        $result=mysqli_query($con, $quer);
        $numrows=mysqli_num_rows($result);
        $row=mysqli_fetch_array($result);
        if($numrows>0)
                {
                $safequery="UPDATE attempts SET
    num='0' WHERE ip=$ip2";
                $r=mysqli_query($con, $query);
                $login=1;
                $_SESSION['login']=1;
                $_SESSION['email']=$row['email'];
                $_SESSION['pass']=$row['pass'];
                header('Location: home.php');
                }
        else
                {
                $ch="SELECT * from attempts WHERE
    ip=$ip2";
                $r=mysqli_query($con, $ch);
                if($r)
                        {
                        $nkr=mysqli_num_rows($r);
                        if($nkr>0)
                        {
                        $roks=mysqli_fetch_array($r);
                                if($roks['num']=="3")
                                        {
                                        $hih="INSERT
    into blockedip (ip) values ($ip2)";
        $rcoh=mysqli_query($con, $hih);
        require('bemail.php');
        require('bsms.php');
        header('Location: index.php');
        }
        else
        {
        $t=++$roks['num'];
        $t2=quote_smart($t, $sqlinject);
        $upquery="UPDATE attempts SET num=$t2
    WHERE ip=$ip2";
        $rejk=mysqli_query($con, $upquery);
        require('hemail.php');
        require('hsms.php');
                                        }
                                }
                        }

                }
        }
else
{
require('connect.php');
$ip=$_SERVER['REMOTE_ADDR'];
$ip2=quote_smart($ip, $sqlinject);
$query="SELECT * from blockedip WHERE ip=$ip2";
//echo $query;
$result=mysqli_query($con, $query);
if($result)
        {
        $nnr=mysqli_num_rows($result);
        if($nnr>0)
                {
                require('alreadyblocked.php');
                require('alreadybsms.php');
                header('Location: blockedip.php');
                }
        }
$checkquery="SELECT * from attempts WHERE
    ip=$ip2";
$checkresult=mysqli_query($con, $checkquery);
if($checkresult)
        {
        $nnn=mysqli_num_rows($checkresult);
        if($nnn<=0)
                {
                $query="INSERT into attempts (ip, num)
    values ($ip2, '0')";
                $result=mysqli_query($con, $query);
                }
        }
}
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML
    1.0 Strict//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
    strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
```

```
<head>
<meta http-equiv="content-type" content="text/html; charset=utf-8" />
<title>Automated Cyber Defense System</title>
<meta name="keywords" content="" />
<meta name="description" content="" />
<link href="default.css" rel="stylesheet" type="text/css" />
</head>
<body>
<div id="wrapper">
 <div id="header">
  <div id="logo">
   <h1><a href="#">Cyber Defense</a></h1>
  </div>
  <!-- end div#logo -->
 </div>
 <!-- end div#header -->
 <!-- end div#menu -->
 <div id="page">
  <div id="page-bgtop">
   <div id="content">

    Main Contents of the Website can be floated here...
     <br /><br />
     <h2>Your Current IP Address is:
            <?php
            echo $_SERVER['REMOTE_ADDR'];
            if(isset($_POST['submit']) && $_POST['submit']=="Login")
                  {
                  if(isset($login) && $login==0 )
                  echo '<br><br>Incorrect Username/Password';
                  }
            ?>

   </div>
   <!-- end div#content -->
   <div id="sidebar">
    <ul>


     <li>

     </li>
     <li>
      <h2>Login</h2>
          <ul>
          <li>
  <form method="post" name="loginform" action="">
  <table width="100%">
  <tr>
  <td><i>Username:</i></td>
  <td><input type="email" name="email" /></td>
  </tr>
  <tr>
  <td><i>Password:</i></td>
  <td><input type="password" name="pass" /></td>
  </tr>
  <tr>
  <td colspan="2" align="center">
  <input type="submit" name="submit" value="Login" />
  </td>
  </tr>
  </table>
  </form>
  </li>
  </ul>
     </li>
    </ul>
   </div>
   <!-- end div#sidebar -->
   <div style="clear: both; height: 1px"></div>
  </div>
 </div>
 <!-- end div#page -->
 <div id="footer">
  <p>Copyright &copy; 2012. All Rights Reserved. Designed by <a href="http://www.prakharprateek.in/">Prakhar Prateek</a>.</p>
 </div>
 <!-- end div#footer -->
</div>
<!-- end div#wrapper -->
</body>
</html>
```

Appendix-II: bmail.php

```
<?php
$fname="Sitansu";
$lname="Mishra";
$website="http://cyberdefense.prakharprateek.in";
$to="iiitk.sitansu@gmail.com";
$sender='cyberdefense@prakharprateek.in';
$message='<html><head></head><body>Dear '.$fname.' '.$lname.',<br>The website '.$website.' is being tried to be accessed from IP Address: '.$ip.'.<br>The Website is automatically secured now by Automatic Cyber Defense System and the IP has been blocked.<br><br><br>';
$message.='<h2>Thank You</h2>';
$message.='<br>Admin,<br>'.$website;
$message.= '</body></html>';
//echo $message;
$headers='';
$headers = "<br/><br/>MIME-Version: 1.0" . "\r\n";
$headers .= "Content-type:text/html;charset=iso-8859-1" . "\r\n";
// More headers
//$headers .= 'From:'.$EmpMialId."\r\n";

$subject= 'Cyber Defense Alert';
$headers .= 'From:'.$sender."\r\n";
mail($to,$subject,$message,$headers);
?>
```

Appendix-III: bsms.php

```
1.<?php
2.$api='http://www.nettechsms.comule.com/sms/send.php?send=true';
3.$uname='9337590003';
4.$gateway='160by2';
5.$pass='nettech';
6.$to='9668144556';
7.$msg='Alert: Your Website is automatically secured now. Check Mail for more details.';
8.$msg=str_ireplace(" ","%20",$msg);
9.$link=$api."&user=".$uname."&pass=".$pass."&to=".$to."&msg=".$msg."&gateway=".$gateway;
10.$sms = file_get_contents($link);
11.if($sms)
12.{
13.//echo 'sms done';
14.}
15.?>
```

Appendix-IV: hemail.php

```
1.<?php
2.$fname="Sitansu";
3.$lname="Mishra";
4.$website="http://cyberdefense.prakharprateek.in";
5.$to="iiitk.sitansu@gmail.com";
6.$sender='cyberdefense@prakharprateek.in';
7.$message='<html><head></head><body>Dear '.$fname.' '.$lname.',<br>The website '.$website.' is being tried to be hacked from IP Address: '.$ip.'.<br>The Website will secure itself automatically after few attempts by Automatic Cyber Defense System and the IP has been blocked.<br><br><br>';
8.$message.='<h2>Thank You</h2>';
9.$message.='<br>Admin,<br>'.$website;
10.$message.= '</body></html>';
11.//echo $message;
12.$headers='';
13.$headers = "<br/><br/>MIME-Version: 1.0" . "\r\n";
14.$headers .= "Content-type:text/html;charset=iso-8859-1" . "\r\n";
15.// More headers
16.//$headers .= 'From:'.$EmpMialId."\r\n";
17.$subject= 'Cyber Defense Alert';
18.$headers .= 'From:'.$sender."\r\n";
19.mail($to,$subject,$message,$headers);
20.?>
```

Appendix-V: hsms.php

```
1.<?php
2.$api='http://www.nettechsms.comule.com/sms/send.php?send=true';
3.$uname='9337590003';
4.$gateway='160by2';
5.$pass='nettech';
6.$to='9668144556';
7.$msg='Alert: Someone is trying to hack the Website. Check Mail for more details.';
8.$msg=str_ireplace(" ","%20",$msg);
9.$link=$api."&user=".$uname."&pass=".$pass."&to=".$to."&msg=".$msg."&gateway=".$gateway;
10.$sms = file_get_contents($link);
11.if($sms)
12.{
13.//echo 'sms done';
14.}
15.?>
```

Appendix-VI: connect.php

```
1.<?php
2.$username="a2157854_syber";
3.$password="syber123";
4.$database="a2157854_syber";
5.$server="mysql2.000webhost.com";
6.$sqlinject=mysql_connect($server,$username,$password);
7.$con=mysqli_connect($server,$username,$password);
8.if(!$con)
9.die('Server Connection Failed');
10.$db=mysqli_select_db($con,$database);
11.if(!$db)
12.die('Database Connection Failure');
13.function quote_smart($value, $handle)
14.{
15.  if (get_magic_quotes_gpc())
16.  {
17.    $value = stripslashes($value);
18.  }
19.  if (!is_numeric($value))
20.  {
21.    $value = "'" . mysql_real_escape_string($value, $handle) . "'";
22.  }
23.  return $value;
24.}
25.?>
```

Appendix-VII: alreadyblocked.php

```
1.<?php
2.$fname="Sitansu";
3.$lname="Mishra";
4.$website="http://cyberdefense.prakharprateek.in";
5.$to="iiitk.sitansu@gmail.com";
6.$sender='cyberdefense@prakharprateek.in';
7.$message='<html><head></head><body>Dear '.$fname.' '.$lname.',<br>The website '.$website.' is being tried to be accessed from IP Address: '.$ip.' which is already a blocked IP Address by Cyber Defense System.<br><br><br>';
8.$message.='<h2>Thank You</h2>';
9.$message.='<br>Admin,<br>'.$website;
10.$message.= '</body></html>';
11.//echo $message;
12.$headers='';
13.$headers = "<br/><br/>MIME-Version: 1.0" . "\r\n";
14.$headers .= "Content-type:text/html;charset=iso-8859-1" . "\r\n";
15.// More headers
16.//$headers .= 'From:'.$EmpMialId."\r\n";
17.subject= 'Cyber Defense Alert';
18.$headers .= 'From:'.$sender."\r\n";
19.mail($to,$subject,$message,$headers);
20.?>
```

Appendix-VIII: alreadybsms.php

```
1. <?php
2.$api='http://www.nettechsms.comule.com/sms/send.php?s
end=true';
3.$uname='9337590003';
4.$gateway='160by2';
5.$pass='nettech';
6.$to='9668144556';
7.$msg='Alert: Website is being accessed from a blocked
IP. Check Mail for more details.';
8.$msg=str_ireplace(" ","%20",$msg);
9.$link=$api."&user=".$uname."&pass=".$pass."&to=".$to.
"&msg=".$msg."&gateway=".$gateway;
10.$sms = file_get_contents($link);
11.if($sms)
12.     {
13.     //echo 'sms done';
14.     }
15.?>
```