   **PREVENTION OF PHISHING ATTACK**

##### A PROJECT REPORT

##### *Submitted by*

**ASMATH.M (1012106702)**

**HEMALATHA.M (1012106704)**

**MANOJ KUMAR.G (1012106706)**

***in partial fulfillment for the award of the degree***

***of***

**BACHELOR OF TECHNOLOGY**

IN

INFORMATION TECHNOLOGY

SONA COLLEGE OF TECHNOLOGY, SALEM

ANNA UNIVERSITY: CHENNAI 600 025

**APRIL 2016**

ANNA UNIVERSITY: CHENNAI 600 025

**BONAFIDE CERTIFICATE**

Certified that this project report “**PREVENTION OF PHISHING ATTACK”** is the bonafide work of “**ASMATH.M (1012106702), HEMALATHA.M (1012106704), MANOJ KUMAR.G (1012106706)”** who carried out the project work under my supervision.

**SIGNATURE SIGNATURE**

|  |  |
| --- | --- |
| Dr.J.AKILANDESWARI | Dr.J.JEBA EMILYN |
| **HEAD OF THE DEPARTMENT** | **SUPERVISOR** |
| Professor | Associate Professor |
| Department of Information Technology | Department of Information Technology |
| Sona College of Technology, | Sona College of Technology, |
| Salem-636 005. | Salem-636 005. |

Submitted for U10IT802R Project Phase-II viva voice examination held on ....................................

**INTERNAL EXAMINER EXTERNAL EXAMINER**

**ABSTRACT**

Online services have become important part of our lives as they allow anytime, anywhere access to information. Clearly, such services are not only useful for Internet users, but they have also become indispensable for financial organizations because they help reduce operational costs. Phishing is a form of identity theft in which a combination of social engineering and web site spoofing techniques are used to trick a user into revealing confidential information with economic value. In this project, we have developed two prime modules for hacking the information and detection of phishing. While hacking the information, the hacker will hack the whole details of the user and admin without the knowledge of them. By this, the hacker can transfer the funds of the user, collect vital information about them and misuse the credentials for fraud and breaching of authenticity. In the detection of phishing, we have used the anti-phishing technique called PhishNet which helps the user to detect the defected phish page from the phish tank which prevents them from visiting the phishing sites page through URL verification

**LIST OF FIGURES**

**FIGURE NO FIGURE NAME PAGE NO**

4.1.1 Bank login 13

4.1.2 User login 14

4.1.3 Hacker login 14

4.1.4 Phishing search engine 15

6.2.1 Bank Administration login 21

6.2.2 Customer login 23

6.2.3 Hacker Terminal 25

6.2.4 Phishing Search Engine 26

**TABLE OF CONTENTS**

**CHAPTER NO TITLE PAGE NO**

**ABSTRACT iii**

**LIST OF FIGURES iv**

**1. INTRODUCTION**

1.1General Information 6

1.2 Objective 6

**2. SYSTEM SPECIFICATION** 7

**3. LITERATURE SURVEY** 8

**4. DESIGN ARCHITECTURE** 4

**5. SOFTWARE DESCRIPTION** 13

**6. PROJECT DESCRIPTION** 16

**7. PSEUDO CODE** 20

**8. CONCLUSION** 53

**9. FUTURE ENHANCEMENT** 54

**10. REFERENCES**  55

**CHAPTER 1**

**INTRODUCTION**

* 1. **GENERAL INFORMATION**

Phishing is a form of identity theft in which a combination of social engineering and web site spoofing techniques are used to trick a user into revealing confidential information with economic value. In this project, we have developed two major concepts such as hacking the information and detection of phishing. In the hacking concepts, we had developed all about hacking the bank details where hacker terminal track the activity of user for performing the operations such as transferring the funds without the knowledge of the user .The next is about the detection of phishing. By using a phishing technique called PhishNet, the user can detect the phish page and secure the information from the hacker through URL verification from the phish tank.

**1.2 OBJECTIVE**

The main objective of the phishing attack is to steal the information’s of the user. The project describes how hacking can be done in the banking website and helps the user to find the phish page using the URL verification by which the user can secure the confidential information from the hacker

**CHAPTER 2**

**SYSTEM SPECIFICATIONS**

The minimum software requirement document is the official statement of what is required to the developer. It includes both the user requirement for the system and a detailed specification of the system requirement. It specifies only system behavior. It specifies constrains on the implementation. It is easy to change. It serves as reference tool. It is acceptable to undesired changes.

**2.1 HARDWARE REQUIREMENTS**

Processor : Intel Core I3 Processor

Hard disk : 120 GB

Ram : 1 GB

CPU Clock : 2.3 GHZ

**2.2 SOFTWARE REQUIREMENTS**

Operating System : Windows XP

Database : Mysql 5.0

Tools : Xampp (1.8)

User Interface : Html, CSS and PHP.

**CHAPTER 3**

**LITERATURE REVIEW**

**3.1 SYSTEM ANALYSIS**

Analysis is a study of various operations performed by a system and their relationships within and outside the system. Analysis involved a detailed study of the current system, leading to specifications of a new system. During analysis, data are collected on the available files, decision points and transactions handled by the present system. Interviews, on-site observations and questionnaire are the tools used for system analysis. Using the following steps it becomes easy to draw the exact boundary of the new system under consideration.

All procedures, requirements must be analyzed and documented in form of detailed data flow diagrams, data diagrams, logical data structures and miniature specifications for new systems.

**3.2 EXISTING SYSTEM – A REVIEW**

In this, we have studied that the phishing will made hack the personalized and secure information of the user .Major technologies and algorithms are have to be used for providing secure information’s

In the existing system, they had referred the few papers which helps us to made as the references. Specifically, they proposed CANTINA+, the most comprehensive feature-based approach in the literature including eight novel features, which exploits the HTML Document Object Model (DOM), search engines and third party services with machine learning techniques to detect phish.

Moreover, they designed two filters to help reduce FP and achieve runtime speedup. The first is a near-duplicate phish detector that uses hashing to catch highly similar phish. The second is a login form filter, which directly classifies webpage with no identified login form as legitimate. They extensively evaluated CANTINA+ with two methods on a diverse spectrum of corpora with 8118 phish and 4883 legitimate webpage. In the randomized evaluation, CANTINA+ achieved over 92% TP on unique testing phish and over 99% TP on near-duplicate testing phish, and about 0:4% FP with 10% training phish. In the time-based evaluation, CANTINA+ also achieved over 92% TP on unique testing phish, over 99% TP on near-duplicate testing phish, and about 1:4% FP under 20% training phish with a two-week sliding window. Capable of achieving 0:4% FP and over 92% TP, our CANTINA+ has been demonstrated to be a competitive anti-phishing solution

In the phishdef, they take the following steps to identify phishing URLs. First, they carefully select lexical features of the URLs that are resistant to obfuscation techniques used by attackers. Second, they evaluate the classification accuracy when using only lexical features, both automatically and hand-selected ,vs. when using additional features. They show that lexical features are sufficient for all practical purposes. Third, they thoroughly compare several classification algorithms, and we propose to use an online method (AROW) that is able to overcome noisy training data.

Based on the insights gained from our analysis, they propose PhishDef , a phishing detection system that uses only URL names and combines the above three elements. Phishdef is a highly accurate method (when compared to state-of-the-art approaches over real datasets), lightweight (thus appropriate for online and client-side deployment), proactive (based on online classification rather than blacklists), and resilient to training data inaccuracies (thus enabling the use of large noisy training data)

There are two types features that can be used in URL classification: lexical features, i.e., features which are readily available from the URL names; and external features,i.e., features acquired from queries to remote servers. We refer to lexical and external features together as full features. Lexical features are based only on the URL names and are appropriate for implementation at the client. External features rely on the availability of remote servers, introduce additional latency due to the required queries, and consume more resources of the client, e.g. , battery life and bandwidth of mobile phones. Nonetheless, one would expect that relying on a more comprehensive set of features, rather than lexical features only, would lead to higher classification accuracy.

In another existing paper, they study the anatomy of phishing URLs that are created with the specific intent of impersonating a trusted third party to trick users into divulging personal data. Unlike previous work in this area, they only use a number of publicly available features on URL alone; in addition, we compare performance of different machine learning techniques and evaluate the efficacy of real -time application of our method. Applying it on real -world data sets, they demonstrate that the proposed approach is highly effective in detecting phishing URLs with an error rate of 0.3%, false positive rate of 0.2% and false negative rate of about 0.5%, thereby improving previous results on the important problem of phishing detection.

The performs the operations such as 1) A demonstration that a phishing URL can be detected by using the information on the URL alone without looking at the actual web page contents and regardless of the context or medium the URL is distributed. 2) An examination of the importance of publicly available information on a URL in the evaluation of whether that URL is phishing. 3) A comparison of a number of publicly available machine learning classifiers to determine the best for classification of phishing URLs. 4) A demonstration that the proposed methodology can be used for near real -time application in detecting phishing URLs. 5) A demonstration that the properties of phishing URLs change over time and how the data drift can affect classifiers‟ performances

. Using the whole URL including all path information would not yield any value-able results. After conducting some rest trials, we found that for the mostly long queries, small spelling mistakes do not return usable suggestions. Hence we developed algorithms to detect possible search terms that are worth checking. We derived those from common attacks in literature and from what we found during the analysis of existing phishing URLs. shows four example URLs with the extracted terms highlighted. The four cases are as follows:

1. BASENAME

The base name is the real domain name as registered at the registrar for the domain. The base name usually consists of the top-level domain (e.g. `com') and the domain name itself (e.g. `PayPal'). Phishes cannot use the original domain name as it is already registered by the original company. Instead, they register misspellings or similar looking domain names.

1. SUB DOMAINS

For each base domain, the owner can specify an arbitrary number of sub domains. This is often used to prep end the domain name of the websites that is attacked. Prep ending the sub domains us.battle.net to any other domain may fool users into thinking that they are on the real domain. Domain highlighting in the browser's location bar is used by browser vendors to counteract such attacks but users are still being tricked by them

1. PATH DOMAIN

In some cases, phishes neither have access to the base name or a sub domain (e.g. when hosting their attack on a free web hosting service). In this case, they place a second domain as a subfolder of the URL path usually right after the domain name. For the remainder of this paper, will refer to those terms as path domains

1. BRAND NAME

A last check they performed was for certain brand names. In some cases not a whole domain but only a brand name is inserted somewhere in the URL. For this special case we did not use the help of a search engine. Instead, they only counted the sole occurrence

**CHAPTER 4**

**DESIGN ARCHITECTURE**

**4.1. BLOCK DIAGRAM**

The below block diagram shows the way of creation of the application. The application provides user interface.

**4.1.1 HACKING INFORMATION**

**(i) APPLICATION LOGIN**

Bank Login

Login

Username, password

Verified

View customer details

Add new customer

**(ii) USER LOGIN**

Username

Login

Account summary

Transaction

Password

Verified

Transfer

**(iii) HACKER LOGIN**

Username

Login

Hack customer details

Transfer funds

Password

Verified

Transaction

**4.1.2 DETECTION OF PHISHING**

Login

PhishNet

Obtain result

Username

Password

Copy URL

Verify from phish tank

**CHAPTER 5**

**SOFTWARE DESCRIPTIONS**

**Apache Server**

XAMPP's name is an [acronym](http://en.wikipedia.org/wiki/Acronym_and_initialism) for:

* X (to be read as "cross", meaning [cross-platform](http://en.wikipedia.org/wiki/Cross-platform))
* Apache HTTP Server
* [MySQL](http://en.wikipedia.org/wiki/MySQL)
* [PHP](http://en.wikipedia.org/wiki/PHP)
* [Perl](http://en.wikipedia.org/wiki/Perl)

The program is released under the terms of the GNU General Public License and acts as a free [web server](http://en.wikipedia.org/wiki/Web_server) capable of serving dynamic pages. XAMPP is available for Microsoft Windows, [Linux](http://en.wikipedia.org/wiki/Linux), Solaris, and Mac OS X, and is mainly used for web development projects. This software is useful while you are creating dynamic WebPages using programming languages like PHP, JSP, and Servlets.

XAMPP's designers intended it for use only as a development tool, to allow website designers and programmers to test their work on their own computers without any access to the Internet. To make this as easy as possible, many important security features are disabled by default. In practice, however, XAMPP is sometimes used to actually serve web pages on the World Wide Web a special tool is provided to password-protect the most important parts of the package. XAMPP also provides support for creating and manipulating databases in MySQL and SQLite among others.

**PHP**

PHP (PHP: Hypertext Preprocessor) is an HTML-embedded programming language which executes scripts on a server and allows web designers to create dynamic content that interacts with databases. It supports the use of database servers such as MySQL, Informix and Solid.PHP outsourcing services include PHP and MySQL web development, online shopping carts.

**Advantages and Features of PHP:**

2)     It provides high performance, fast speed and great reliability

6)     PHP programming has multilingual support

7)     PHP is compatible with the vast majority of servers (Apache, IIS, etc.) and is capable of running on most platforms (Unix, Windows, etc.).

8)     PHP is easy to understand and learn, particularly for those who have a background in HTML or JavaScript.

**MYSQL**

MySQL is primarily an RDBMS and ships with no GUI tools to administer MySQL databases or manage data contained within the databases. Users may use the included command line tools, or download MySQL front-ends from various parties that have developed desktop software and web applications to manage MySQL databases, build database structures, and work with data records.

MySQL Workbench is available in two editions, the regular free and open source Community Edition which may be downloaded from the MySQL website, and the proprietary Standard Edition which extends and improves the feature set of the Community Edition.

MySQL allows for many data types for columns, and it is very easy to forget what constraints there are on these data types. This is especially important when it comes to validation - many developers do not check the length of data before inserting it into a table. This section lists the data types available and their constraints.

Many developers are completely unaware that MySQL actually has a huge number of functions built in, to do everything from regular expression-based string comparisons to complicated mathematical calculations. The outsides of the cheat sheet list MySQL functions (note: a few of the more unusual and least used functions have not been included due to constraints of space).

**HTML**

**Hyper Text Markup Language** (**HTML**) is the main markup language for web pages. HTML elements are the basic building-blocks of web pages. HTML is written in the form of HTML elements consisting of tags enclosed in angle rackets (like <html>), within the web page content. HTML tags most commonly come in pairs like <h1> and </h1>, although some tags, known as empty elements, are unpaired, for example <img>.

The first tag in a pair is the start tag, the second tag is the end tag (they are also called opening tags and closing tags). In between these tags web designers can add text, tags, comments and other types of text-based content.

The purpose of a web browser is to read HTML documents and compose them into visible or audible web pages. The browser does not display the HTML tags, but uses the tags to interpret the content of the page. HTML elements form the building blocks of all websites. HTML allows images and objects to be embedded and can be used to create interactive forms. It provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. It can embed scripts in languages such as JavaScript which affect the behavior of HTML web pages.

**CHAPTER 6**

**6.1 PROJECT DESCRIPTION**

Phishing is a form of identity theft in which a combination of social engineering and web site spoofing techniques are used to trick a user into revealing confidential information with economic value. In a typical phishing attack, a large number of spoofed e-mails are sent to random users (i.e., analogous to spam e-mail). These e-mails are disguised such that an unsuspecting victim is easily convinced that the e-mail is coming from a legitimate organization such as a bank. Typically, these e-mails request the victims to "update" their online banking information.

This Project is all about hacking the information where hacker terminal track the activity of user and detect the phishing pages and detect the phishing pages which prevents the user from phish URL. This project contain two different modules such as

* Hacking information
* Detection of phishing

In the hacking phase, we have three different login for customer, Bank admin and hacker login to account which does the following

* Hacker can view the admin details.
* Hacker can know the user details.
* Hacker can transfer the funds without the knowledge of user.

**I**

In the application login, the admin can login to the account by providing the information such as username and password, then view all add the new customer and view all the details of the customer

In the customer login page, the customer can login into the account using the information such as username and password. Then the customer perform certain operations such as summary about the account and able to transfer the funds from their account to another account

In the hacker terminal, the hacker login to the account using the same information for admin and customer login (i.e. username and password).Then hacker able to steal the information of the whole customer details without the knowledge of administrator and transfer the fund without the knowledge of the customer

To prevent these types of theft, the detection of phishing is developed to prevent the phishing attacks .By copying and pasting the URL in the search engine , the user can identify whether it is an defected page or not. The searching phishing page are checked in the phish tank

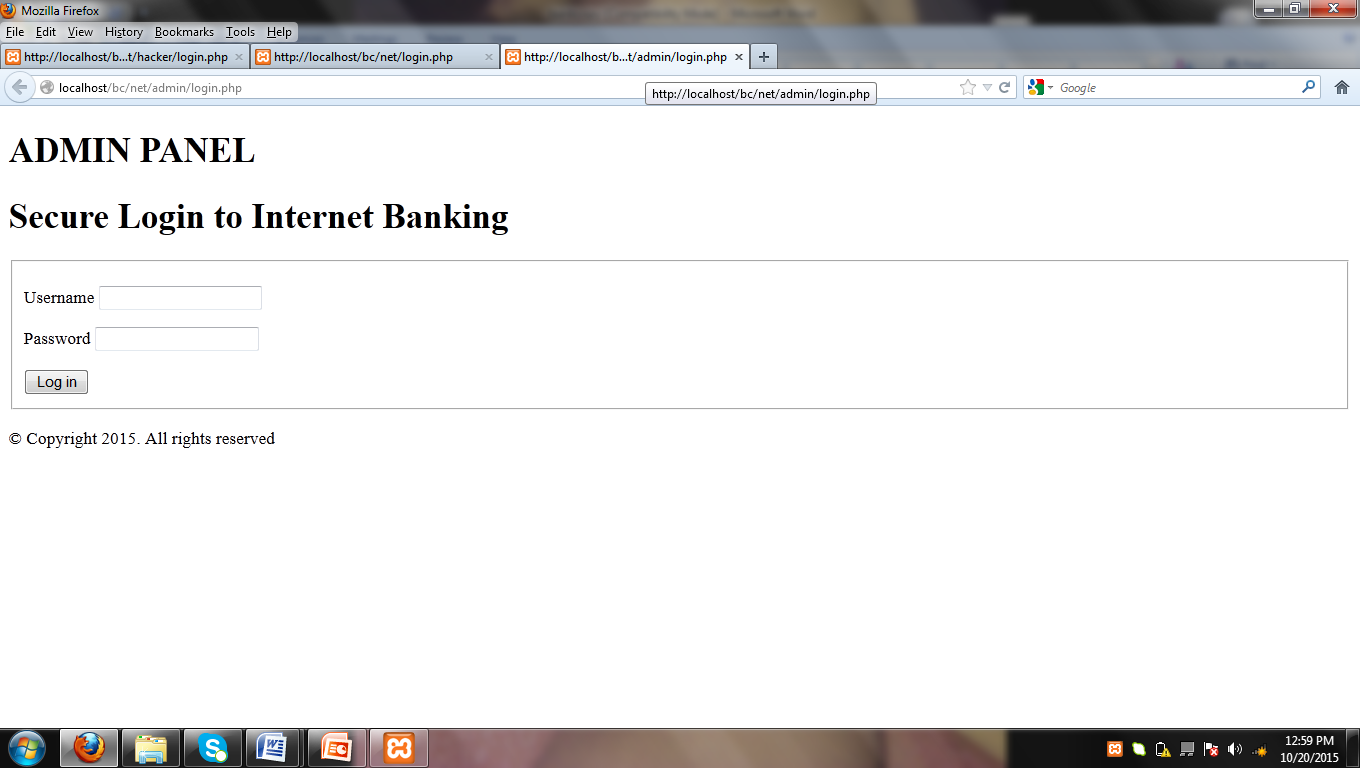
* 1. **MODULES DESCRIPTION**
     1. **HACKING INFORMATION**

In the hacking module, the information of the user are hacker by third party i.e. by hacker through providing the phished url .In this, we had used phishing technique such as link manipulation and phishing through search engines

Link manipulation is a technique in which the phisher sends a link to a website. When the user clicks on the deceptive link, it open up the phisher’s website mentioned in the link

Phishing through is a technique which some phishing scams involve search engine where the user is directed to products sites which may offer low cost products or services. When the user tries to buy the product by entering the credit card details, it’s collected by the phishing sites. There are many fake bank websites offering credit cards or loans to users at a low rate but they are actually phishing sites

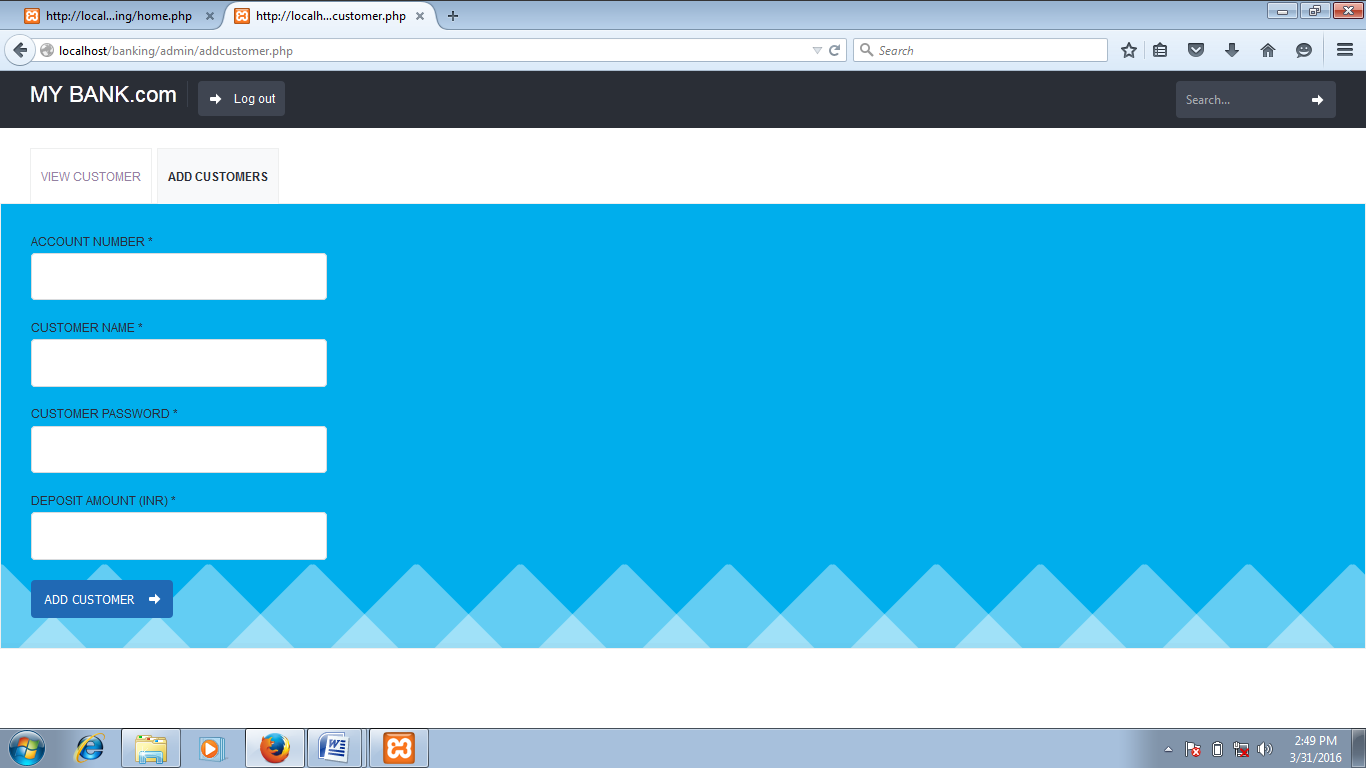
**(i) APPLICATION LOGIN**



Bank login an administrator can add money to customer account they can add new customer and all administrator related activities .This module made the bank report generation easier with very few clicks. It performs two operation such as

* Add customer
* View customer

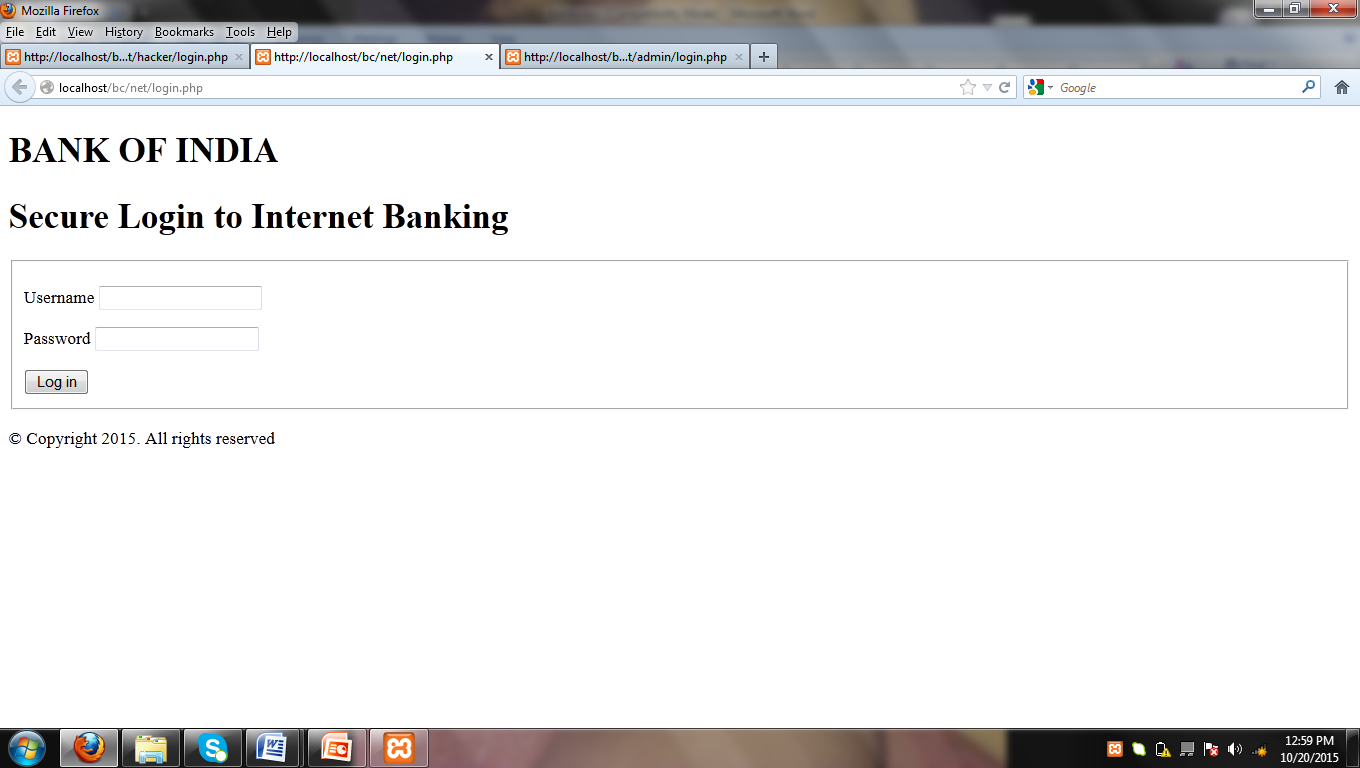
The admin can a new customer by information such as account number, customer name, customer password and deposit amount



The admin can view the customer details such as account number, account name and amount



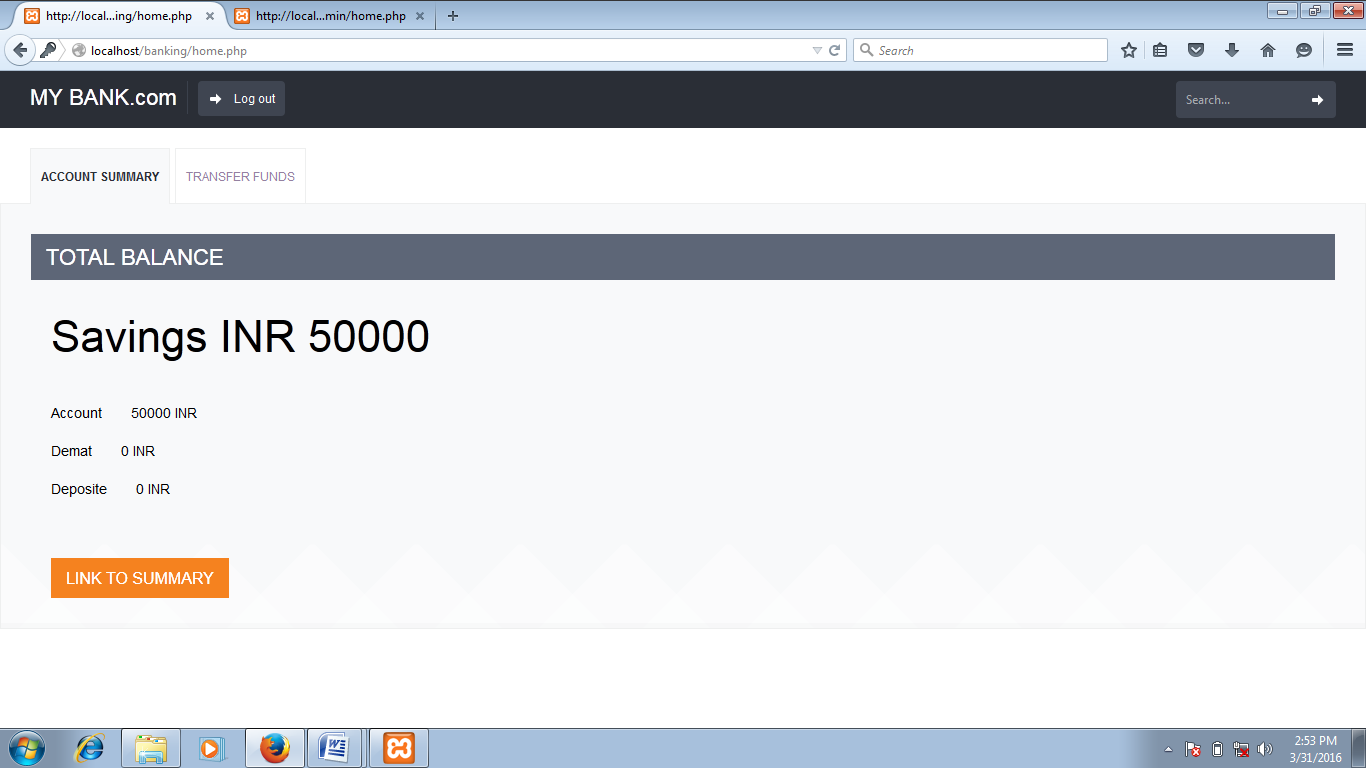
**(ii) CUSTOMER OR USER LOGIN**



In this module where user needs to login credentials (i.e., Username or email and password.) to login. If the login is successful it redirects to the corresponding page through which the user can transfer the funds and view the banking details of the user. It performs the operation such as

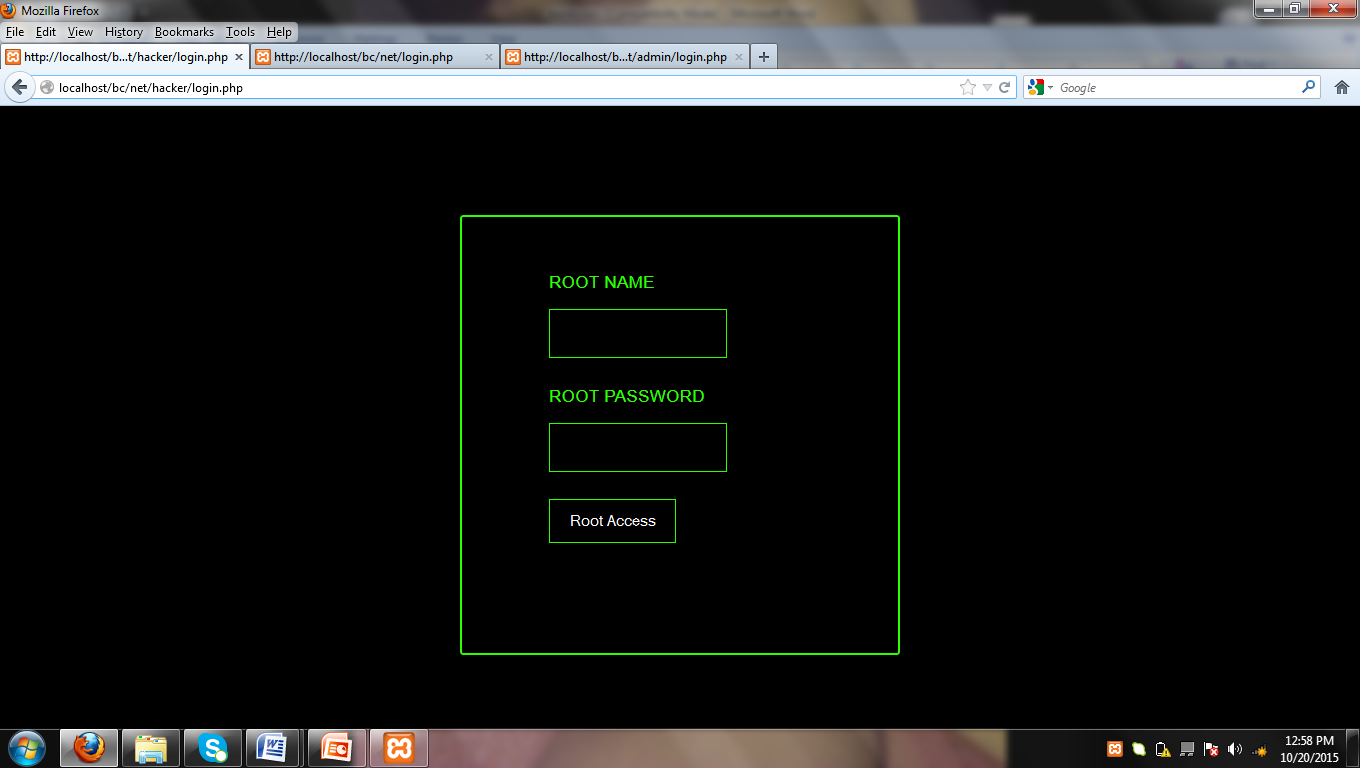
* + - * Amount summary
      * Transaction

In the amount summary, it contain the information about the total balance in the account,demat and deposit

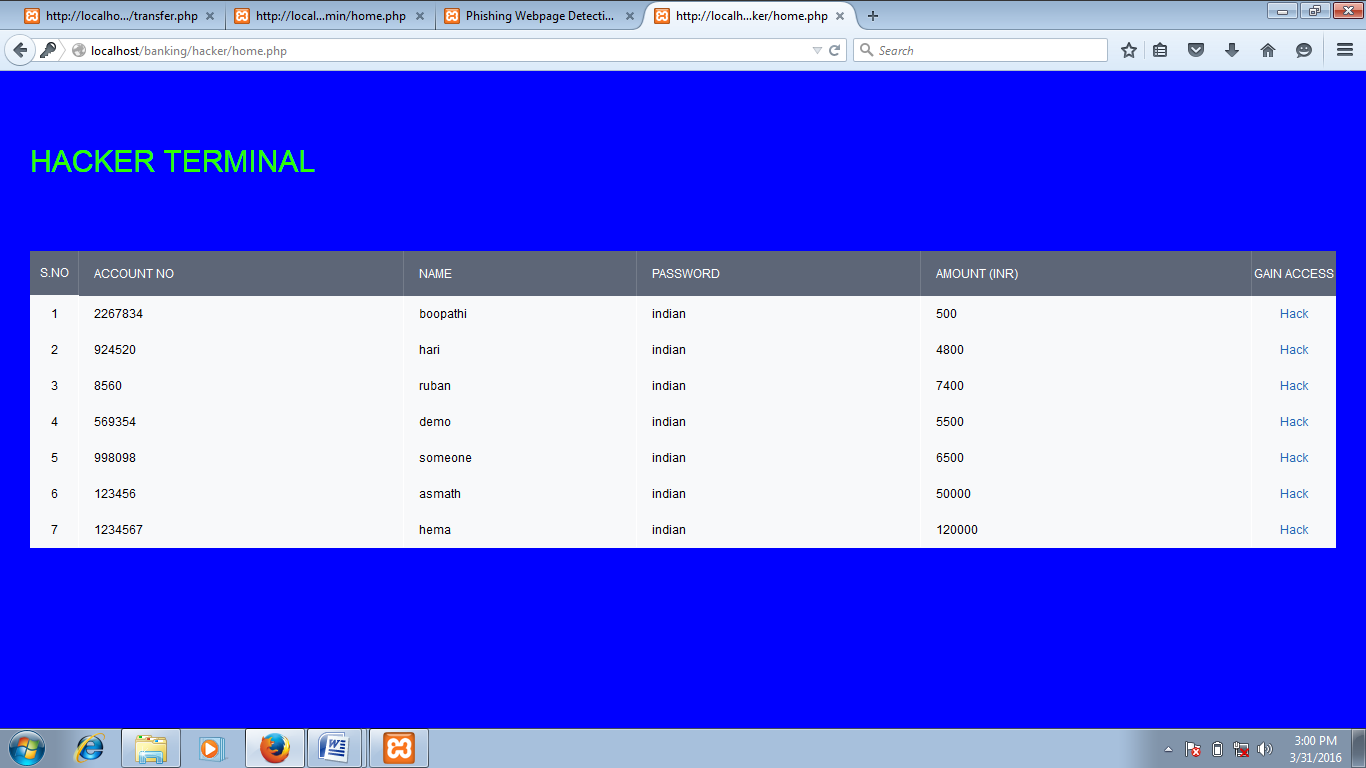


In the transfer fund, it helps the customer to transfer their account balance to another account by providing the information such as payee account name, payee account name and amount

**(iii) HACKER TERMINAL**



This terminal is actually not developed by Bank or Administrator. It was designed by hackers where they can track the activity of user and admin.Then hacker will transfer funds without the knowledge of user and hack the confidential information from the admin login



The hacker will comes to know about the details of every customer such as account number, customer name and total balance in their account

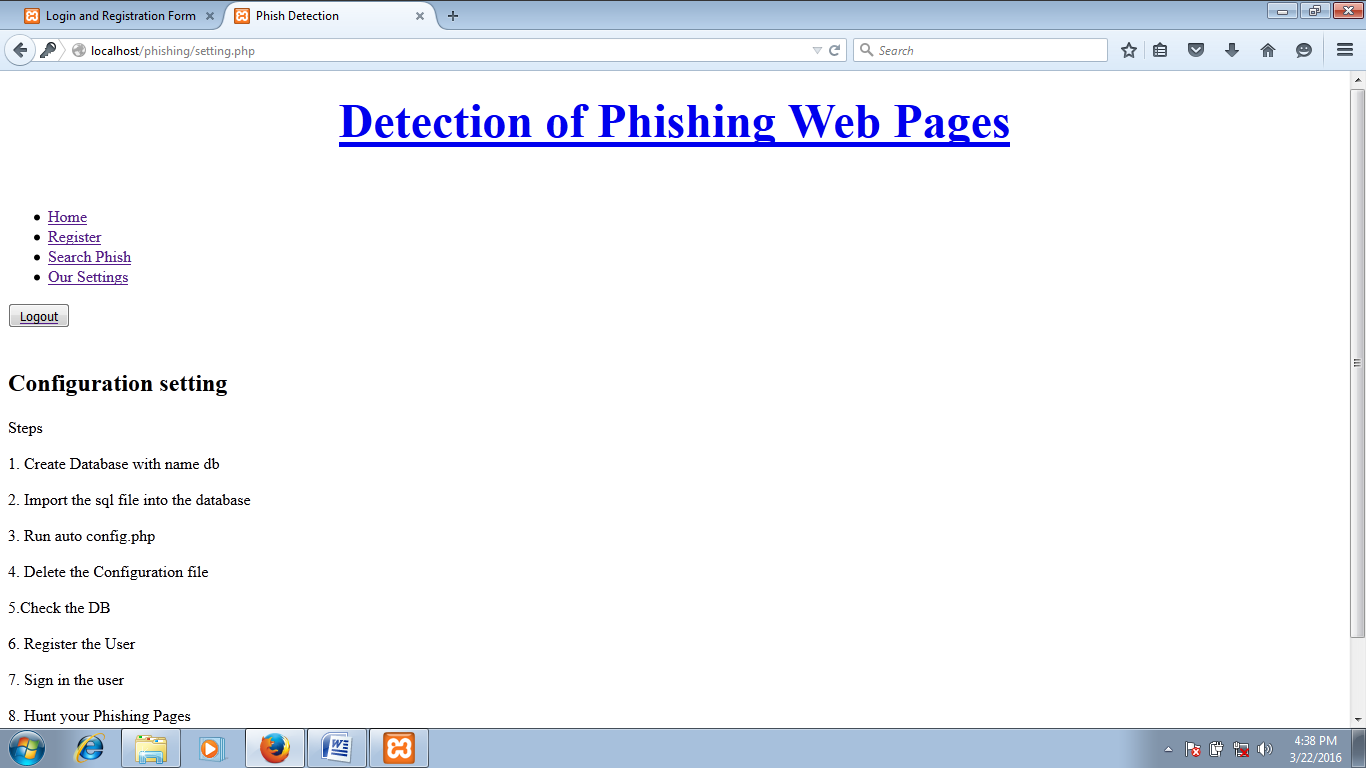
* + 1. **DETECTION OF PHISHING**

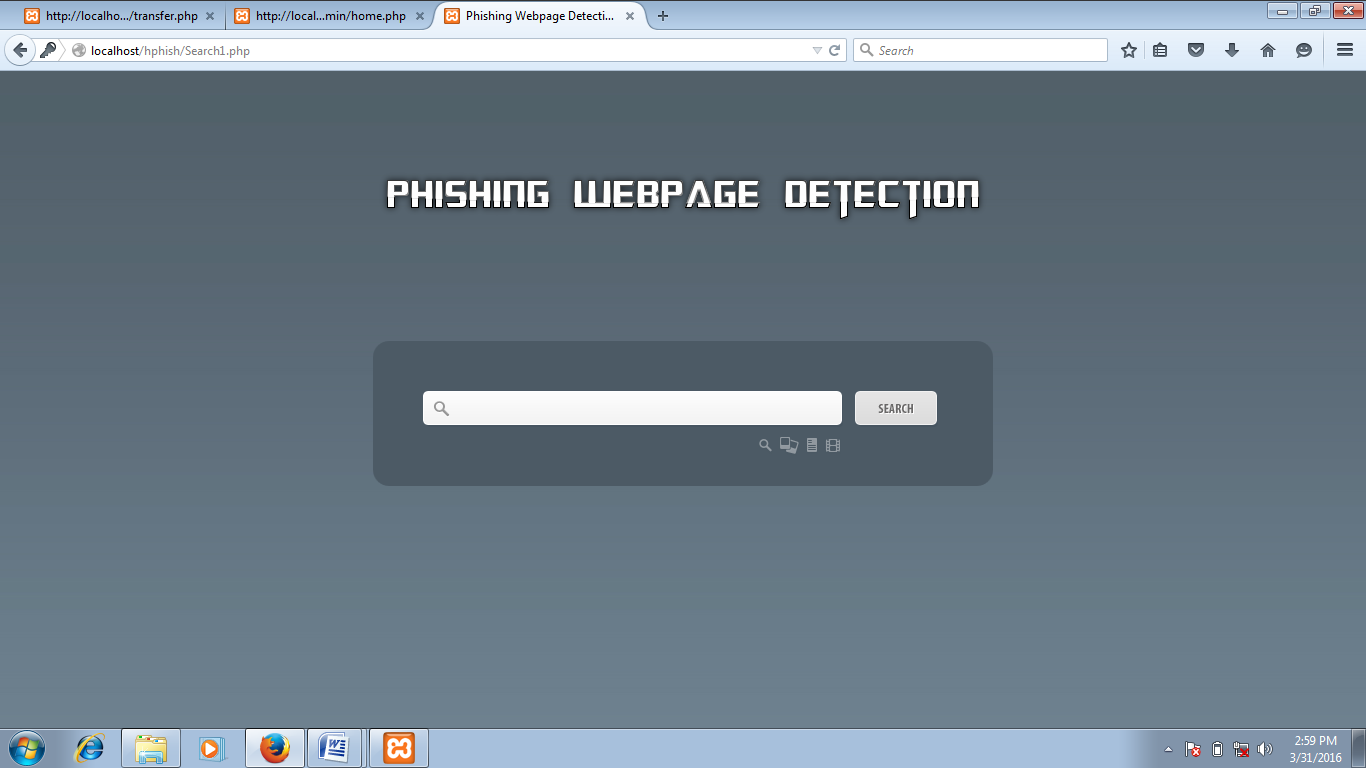
In this project, we have proposed the anti-phishing programs known as phish net as like phish tank sitechecker which detect the phishing attacks through URL from phish tank

PhishNet is technique which predicts malicious URLs from the user request. In this module, using the search engine, the user can detect the defected phish page which matches the URL from the phish tank database.

The phish tank is the database for phishing web pages. Its gives some regular service to the API developer.

We made a link between the phish tank servers to the search engine. The phish tank web databases are storing the URL of the phish page.





**CHAPTER 7**

**PSEUDOCODE**

**HOME.PHP**

<?php include '../core/connection.php'; ?>

<?php include '../core/function.php'; ?>

<?php

ob\_start();

session\_start();

if(!isset($\_SESSION['bank']))

{

header('location: login.php');

}

?>

<!DOCTYPE html>

<html>

<head>

<title></title>

<link href='http://fonts.googleapis.com/css?family=Droid+Sans:400,700' rel='stylesheet' type='text/css'>

<link rel="stylesheet" href="css/style.css" />

</head>

<body>

<div id="top-bar">

<div class="page-full-width clearfix">

<ul id="nav" class="fl">

<li class="v-sep"><span class="logo">MY BANK.com</span></li>

<li ><a href="#" >Logged in as <strong>Bank Admin</strong></a>

<ul>

<li><a href="#">User profile</a></li>

<li><a href="#">password</a></li>

</ul>

</li>

<li><a href="#" >Settings</a></li>

<li><a href="logout.php" >Log out</a></li>

</ul>

</div><!-- page-full-width -->

</div><!-- top-bar -->

<div id="header-with-tabs">

<div class="page-full-width clearfix">

<ul id="tabs">

<li><a href="home.php" class="active-tab">VIEW CUSTOMER</a></li>

<li><a href="addcustomer.php" >ADD CUSTOMERS</a></li>

</ul>

<a href="#" class="fr" id="company-logo-small"><!--<img src="img/company-logo.png" alt="Techforge" >--></a>

</div><!-- page-full-width -->

</div><!-- header -->

<div id="content">

<div class="page-full-width clearfix">

<table>

<thead>

<tr>

<th>S.no</th>

<th>Account No</th>

<th>Name</th>

<th>Amount (INR)</th>

</tr>

</thead>

<tbody>

<?php

$received = mysql\_query("SELECT \* FROM customer");

while($rec = mysql\_fetch\_assoc($received))

{

$customer\_id = $rec['customer\_id'];

$customer\_account = $rec['customer\_account'];

$customer\_name = $rec['customer\_name'];

$customer\_deposit = $rec['customer\_deposit'];

?>

<tr>

<td>

<?php echo $customer\_id; ?>

</td>

<td>

<?php echo $customer\_account; ?>

</td>

<td><?php echo $customer\_name; ?></td>

<td><?php echo $customer\_deposit; ?></td>

</tr>

</tbody>

</table>

</div><!-- page-full-width -->

</div><!-- content -->

<div id="footer">

<p>&copy; Copyright 2015. All rights reserved</p>

</div><!-- footer -->

</body>

</html>

**HACK.PHP**

<?php include '../core/connection.php'; ?>

<?php include '../core/function.php'; ?>

<?php

ob\_start();

session\_start();

if(!isset($\_SESSION['hacker']))

{

header('location: login.php');

}

if(isset($\_GET['acc']))

{

$acc = mysql\_real\_escape\_string($\_GET['acc']);

$details = user\_assets\_account($acc);

}

else

{

header('location: home.php');

}

?>

<!DOCTYPE html>

<html>

<head>

<title></title>

<link href='http://fonts.googleapis.com/css?family=Droid+Sans:400,700' rel='stylesheet' type='text/css'>

<link rel="stylesheet" href="css/style.css" />

<script type="text/javascript" src="js/jquery.js"></script>

<style type="text/css">

html {

background-color: black;

}

</style>

</head>

<body>

<?php

# SEND FUND

if(isset($\_POST['payee\_name']) && isset($\_POST['payee\_account']) && isset($\_POST['amount']))

{

$error = Array();

$payee\_name = mysql\_real\_escape\_string($\_POST['payee\_name']);

$payee\_account = mysql\_real\_escape\_string($\_POST['payee\_account']);

$amount = mysql\_real\_escape\_string($\_POST['amount']);

if(customer\_account\_exists($payee\_account) != 1)

{

$error[] = 'Account Number not exists!';

}

if($payee\_name == '')

{

$error[] = 'Enter Payee Account Name';

}

if($payee\_account == '')

{

$error[] = 'Enter Payee Account Number';

}

if($amount == '')

{

$error[] = 'Enter Amout to send funds!';

}

if($details['customer\_deposit'] == 0)

{

$error[] = 'Insufficient Balance!';

}

if($details['customer\_deposit'] <= $amount)

{

$error[] = 'Available Balance is Low!';

}

if(empty($error))

{

$debit = $details['customer\_deposit'] - $amount;

mysql\_query("UPDATE customer SET customer\_deposit = '$debit' WHERE customer\_name = '$details[customer\_name]'");

$payee = user\_assets\_account($payee\_account);

$new\_deposit = $payee['customer\_deposit'] + $amount;

mysql\_query("UPDATE customer SET customer\_deposit = '$new\_deposit' WHERE customer\_name = '$payee[customer\_name]'");

$date = date('d/M/Y');

$statement = mysql\_query("INSERT INTO statement (customer\_account, payee\_account, amount, dated) VALUES('$details[customer\_account]', '$payee[customer\_account]', '$amount', '$date')");

if($statement)

{

echo '<script>alert("Fund Transfered Successfully!");</script>';

//header('location: transfer.php');

}

else

{

foreach ($error as $key => $value) {

?>

<script type="text/javascript">

$(document).ready(function(){

alert(<?php echo '"'.$value.'"'; ?>);

});

</script>

<?php

}

}

}

?>

<div class="page-full-width clearfix">

<div class="navi clearfix">

<div class='hack-logo'>HACKER TERMINAL</div>

<div class='root-logo'>ROOT ACCESSED BANK.com</div>

</div>

<div class="naviga">

<a href="home.php">Gained users</a>

</div>

<div class="clearfix">

<div class="hack-left">

<form action="" method="post" class="hack">

<fieldset>

<p>

<label>Target Account \*</label>

<input type="text" class="default-width-input round" name="customer\_account" value="<?php echo $details['customer\_account']; ?>" disabled />

</p>

<p>

<label>Total Available Balance</label>

<input type="text" class="default-width-input round" name="customer\_deposit" value="<?php echo $details['customer\_deposit']; ?>" disabled />

</p>

<p>

<label>Payee Account Name \*</label>

<input type="text" class="default-width-input round" name="payee\_name"/>

</p>

<p>

<label>Payee Account Number \*</label>

<input type="text" class="default-width-input round" name="payee\_account"/>

</p>

<p>

<label>Amount (INR) \*</label>

<input type="text" class="default-width-input round" name="amount"/>

</p>

<input type="submit" value="SEND FUND" name="edit" class="button image-right text-upper round blue" id="add-category">

</fieldset>

</form>

</div><!-- hack-left -->

<div class="hack-right">

<h1>Screenshot of <?php echo $details['customer\_name']; ?> - Terminal</h1>

<img src="img/hacker.jpg">

</div>

</div>

</div><!-- page-full-width -->

</body>

</html>

**SEARCH.PHP**

<!DOCTYPE html>

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

<title>Phishing Webpage Detection</title>

</head>

<body>

<br>

<?php

mysql\_connect("localhost", "root", "") or die("Error connecting to database: ".mysql\_error());

/\*

localhost - it's location of the mysql server, usually localhost

root - your username

third is your password

\*/

mysql\_select\_db("db") or die(mysql\_error());

$query = $\_GET['query'];

$min\_length = 3;

if(strlen($query) >= $min\_length){ // if query length is more or equal minimum length then

$query = htmlspecialchars($query);

$query = mysql\_real\_escape\_string($query);

$raw\_results = mysql\_query("SELECT \* FROM phish2

WHERE (`url` LIKE '%".$query."%') OR (`urldetail` LIKE '%".$query."%')") or die(mysql\_error());

if(mysql\_num\_rows($raw\_results) > 0){ // if one or more rows are returned do following

while($results = mysql\_fetch\_array($raw\_results)){

echo "<center>";

echo "<font size=10 color=blue> This is Defected Phish Page</font> <br>";

echo "The URL Verified following Times";

echo "<p><b>Url Description: </b>".$results['url']." <b> </p>

<p>Verify By :</b> ".$results['urldetail']."</p>";

echo "</center>";

break;

}

}

else{ // if there is no matching rows do following

echo "<center> The Url is Not Contain any worms </center>";

}

}

else{ // if query length is less than minimum

echo "Minimum length is ".$min\_length;

}

?>

</body>

</html>

**LOGIN.PHP**

<?php include '../core/connection.php'; ?>

<?php

ob\_start();

session\_start();

if(isset($\_SESSION['hacker']))

{

header('location: home.php');

}

?>

<!DOCTYPE html>

<html>

<head>

<title></title>

<link href='http://fonts.googleapis.com/css?family=Open+Sans:300italic,400italic,600italic,700italic,800italic,700,300,600,800,400' rel='stylesheet' type='text/css'>

<link rel="stylesheet" href="css/style.css" />

<style type="text/css">

html {

background: url(img/hack3.png) no-repeat center center fixed;

-webkit-background-size: cover;

-moz-background-size: cover;

-o-background-size: cover;

background-size: cover;

}

</style>

</head>

<body>

<div class="hacker-login">

<?php

if(isset($\_POST['username']) && isset($\_POST['password']))

{

$username = mysql\_real\_escape\_string($\_POST['username']);

$password = mysql\_real\_escape\_string(md5($\_POST['password']));

$sql = mysql\_query("SELECT \* FROM hacker WHERE username = '$username' AND password = '$password' ");

$row = mysql\_num\_rows($sql);

if($row == 1)

{

$\_SESSION['hacker'] = $username;

header('location: home.php');

}

?>

<form action="" method="post">

<label>Root name</label>

<input type="text" name="username">

<label>Root password</label>

<input type="password" name="password">

<input type="submit" value="Root Access">

</form>

</div>

</body>

</html>

**TRANSFER.PHP**

<?php include 'core/connection.php'; ?>

<?php include 'core/function.php'; ?>

<?php

ob\_start();

session\_start();

if(!isset($\_SESSION['customer\_name']))

{

header('location: login.php');

}

$details = user\_assets($\_SESSION['customer\_name']);

?>

<!DOCTYPE html>

<html>

<head>

<title></title>

<link href='http://fonts.googleapis.com/css?family=Droid+Sans:400,700' rel='stylesheet' type='text/css'>

<link rel="stylesheet" href="css/style.css" />

<script type="text/javascript" src="js/jquery.js"></script>

</head>

<body>

<?php

# SEND FUND

if(isset($\_POST['payee\_name']) && isset($\_POST['payee\_account']) && isset($\_POST['amount']))

{

$error = Array();

$payee\_name = mysql\_real\_escape\_string($\_POST['payee\_name']);

$payee\_account = mysql\_real\_escape\_string($\_POST['payee\_account']);

$amount = mysql\_real\_escape\_string($\_POST['amount']);

if(customer\_account\_exists($payee\_account) != 1)

{

$error[] = 'Account Number not exists!';

}

if($payee\_name == '')

{

$error[] = 'Enter Payee Account Name';

}

if($payee\_account == '')

{

$error[] = 'Enter Payee Account Number';

}

if($amount == '')

{

$error[] = 'Enter Amout to send funds!';

}

if($details['customer\_deposit'] == 0)

{

$error[] = 'Insufficient Balance!';

}

if($details['customer\_deposit'] <= $amount)

{

$error[] = 'Available Balance is Low!';

}

if(empty($error))

{

$debit = $details['customer\_deposit'] - $amount;

mysql\_query("UPDATE customer SET customer\_deposit = '$debit' WHERE customer\_name = '$\_SESSION[customer\_name]'");

$payee = user\_assets\_account($payee\_account);

$new\_deposit = $payee['customer\_deposit'] + $amount;

mysql\_query("UPDATE customer SET customer\_deposit = '$new\_deposit' WHERE customer\_name = '$payee[customer\_name]'");

$date = date('d/M/Y');

$statement = mysql\_query("INSERT INTO statement (customer\_account, payee\_account, amount, dated) VALUES('$details[customer\_account]', '$payee[customer\_account]', '$amount', '$date')");

if($statement)

{

echo '<script>alert("Fund Transfered Successfully!");</script>';

//header('location: transfer.php');

}

}

else

{

foreach ($error as $key => $value) {

?>

<script type="text/javascript">

$(document).ready(function(){

alert(<?php echo '"'.$value.'"'; ?>);

});

</script>

<?php

}

}

}

?>

<?php include 'temp/navigation.php'; ?>

<div id="header-with-tabs">

<div class="page-full-width clearfix">

<ul id="tabs">

<li><a href="home.php">ACCOUNT SUMMARY</a></li>

<li><a href="transfer.php" class="active-tab">TRANSFER FUNDS</a></li>

</ul>

<a href="#" class="fr" id="company-logo-small"><!--<img src="img/company-logo.png" alt="Techforge" >--></a

</div><!-- page-full-width -->

</div><!-- header -->

<div id="content">

<div class="page-full-width clearfix">

<form action="" method="post">

<fieldset>

<p>

<label>Your Account \*</label>

<input type="text" class="default-width-input round" name="customer\_account" value="<?php echo $details['customer\_account']; ?>" disabled />

</p>

<p>

<label>Total Available Balance</label>

<input type="text" class="default-width-input round" name="customer\_deposit" value="<?php echo $details['customer\_deposit']; ?>" disabled /> </p>

<p>

<label>Payee Account Name \*</label>

<input type="text" class="default-width-input round" name="payee\_name"/>

</p>

<p>

<label>Payee Account Number \*</label>

<input type="text" class="default-width-input round" name="payee\_account"/>

</p>

<p>

<label>Amount (INR) \*</label>

<input type="text" class="default-width-input round" name="amount"/>

</p>

<input type="submit" value="SEND FUND" name="edit" class="button image-right text-upper round blue ic-right-arrow" id="add-category">

</fieldset>

</form>

</div><!-- page-full-width -->

</div><!-- content -->

<script type="text/javascript">

</script>

</body>

</html>

**CHAPTER 8**

**CONCLUSION**

Most phishing sites prompt user to update sensitive information by logging in with a password or verifying personal information such as credit card number or social security number. Currently most anti-phishing techniques function in a similar manner in detecting viruses. Manual verification of a few phishing sites shows that the anti-phishing program is effective at detecting the new phishing site. If the user attempts to visit a URL in the phish tank then the user is prevented from visiting the site or warned that the site is a known phishing site

**CHAPTER 9**

**FUTURE ENHANCEMENT**

Unfortunately, the usefulness of online services has been overshadowed by large-scale phishing attacks launched against Internet users .Online services have become important part of our lives as they allow anytime, anywhere access to information. Phishing is a never-ending battle like most security related problems. New detection methods will be introduced and then the Phishers will find ways around the detection methods.

In future enhancements, the phish attacks on the online services can be prevented such as net banking, online shopping, and social networks. Major technologies and encrypted algorithms have to be used for providing more secure information’s.

**CHAPTER 10**

**REFERENCES**

* Phishdef: URL Names Say It Alld published by Anh Le, Athina Markopoulou from University of California, Irvine in 12 Sep 2010
* CANTINA+:A Feature-rich Machine Learning Framework for Detecting Phishing Web Sites published by GUANG XIANG, JASON HONG, CAROLYN P. ROSE, LORRIE CRANOR, School of Computer Science from Carnegie Mellon University
* Learning To Detect Phishing URLs published by Ram B. Basnet, Andrew H. Sung, Quingzhong Liu Colorado Mesa University, USA New Mexico Tech, Leroy Pl. Socorro , USA Sam Houston State University, Huntsville, USA in 06 Jun 2014
* Sophisticated Phishers Make More Spelling Mistakes: Using URL Similarity against Phishing published by Max-Emanuel Maurer and Lukas Hofer from University of Munich Media Informatics Group, Germany
* Techniques for detecting zero day phishing websites published by Michael Blasi from Iowa state university in 2014
* <http://www.w3org.com>
* <http://www.w3school.com>
* <http://hacklearning.com>
* <http://hackeracademy.com>