



Jaypee University of Information Technology
Solan (H.P.)
LEARNING RESOURCE CENTER

Acc. Num. *SP07021* Call Num:

General Guidelines:

- ◆ Library books should be used with great care.
- ◆ Tearing, folding, cutting of library books or making any marks on them is not permitted and shall lead to disciplinary action.
- ◆ Any defect noticed at the time of borrowing books must be brought to the library staff immediately. Otherwise the borrower may be required to replace the book by a new copy.
- ◆ The loss of LRC book(s) must be immediately brought to the notice of the Librarian in writing.

Learning Resource Centre-JUIT



SP07021

BREAKING CAPTCHAs
&
IMPLEMENTING NEWER METHODS TO MAKE THEM MORE SECURE

071213 PALLAVI JHA

071244 ANISHA PABBI

Under the supervision of
BRIG. (RETD.) S.P. GHRERA
DR. NITIN CHANDERWAL



MAY 2011

Submitted in partial fulfillment the Degree of
Bachelor of Technology

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING &
INFORMATION TECHNOLOGY

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY

WAKNAGHAT



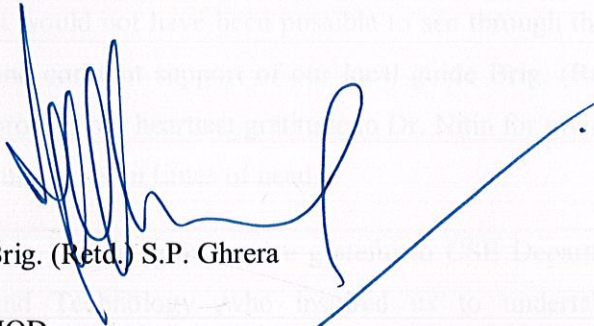
TABLE OF CONTENTS

Chapter No.	Topic	Page No
	Certificate from the Supervisor	iii
	Acknowledgement	iv
	Summary	v
	List of Figures	vi
	List of Symbols and Acronyms	vii
I	INTRODUCTION	1-5
	1. CAPTCHAs	1
	2. Applications	1
	3. Objective and Scope	3
	4. Implementation	4
	5. Resources and Limitations	4
II	LITERATURE SURVEY	6-11
	1. PHP	6
	2. Types of CAPTCHAs	7
	2.1 Text CAPTCHA	7
	2.2 Image CAPTCHA	8
	2.3 Audio CAPTCHA	9
	3. CAPTCHA Authentication Process	9
	4. Loopholes	10
III	BREAKING TICKETMASTER CAPTCHAs	12-15
	1. Background Removal	12
	1.1 Removing By Color Delta	12
	1.2 Quantizing the result	13
	2. Line Removal	14
	2.1 Line Pattern Discovery	14
	2.2 Sonar Sweep Process	14

IV	IMPLEMENTATION	16-75
	1. Text CAPTCHA	16
	2. Image CAPTCHA	19
	3. MAPTCHA	21
	4. Animated MAPTCHA	26
	5. Audio CAPTCHA	48
	6. CAPCHA based email hacking prevention	70
	6.1 Image grid algorithm	70
	6.2 Image mining algorithm	71
	Process Description	
	Level 0 DFD	73
	Level 1 DFD	74
	Activity Diagram	75
V	TESTING	76-79
	1. Testing	76
	2. Types of Testing	
	2.1 White-Box Testing	76
	2.2 Black Box Testing	77
	3. Testing Levels	77
	4. Regression Testing	78
VI	CONCLUSION	80-81
	Future of CAPTCHAs	80
	References	82
	Publications	82

CERTIFICATE

This is to certify that the work titled **Breaking CAPTCHAs and Implementing Newer Methods to Make Them More Secure** submitted by **Pallavi Jha – 071213, Anisha Pabbi - 071244** in partial fulfillment for the award of degree of Bachelor of Technology in Computer Science Engineering of Jaypee University of Information Technology, Waknaghat has been carried out under my supervision. This work has not been submitted partially or wholly to any other University or Institute for the award of this or any other degree or diploma.



Brig. (Retd.) S.P. Ghera

HOD
Department of Computer Science Engineering and Information Technology
Jaypee University of Information Technology
Waknaghat



Dr. Nitin

Professor
Department of Computer Science Engineering and Information Technology
Jaypee University of Information Technology
Waknaghat

ACKNOWLEDGEMENT

“Life doesn't require that we be the best, only that we try our best.”

Apart from the efforts by us, the success of this project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our sincere indebtedness and sense of gratitude to the people who have been instrumental in the successful completion of this project.

It would not have been possible to see through the undertaken project without the guidance and constant support of our local guide Brig. (Retd.) S. P. Ghrera .We would also like to provide our heartiest gratitude to Dr. Nitin for giving us the idea of the project and being very supportive in times of need.

As a final note, we are grateful to CSE Department of Jaypee University of Information and Technology ,who inspired us to undertake difficult tasks by their strength of understanding our calibre and our requirements and taught us to work with patience and provided constant encouragement to successfully complete the project.

Pallavi Jha

Pallavi Jha

(071213)

A Pabbi

Anisha Pabbi

(071244)

SUMMARY

Security mechanisms are of the utmost importance in today's world where the usage of the World Wide Web has increased by ten folds. These mechanisms are implemented through CAPTCHA (Completely Automated Public Turing Test to tell Computers and Humans Apart) which is a type of challenge-response test used in computing as an attempt to ensure that the response is not generated by a computer.

CAPTCHAs are used in attempts to prevent automated software from performing actions which degrade the quality of service of a given system, whether due to abuse or resource expenditure. CAPTCHAs can be deployed to protect systems vulnerable to e-mail spam, such as the webmail services of Gmail, Hotmail, and Yahoo! Mail. They are also used to minimize automated posting to blogs, forums and wikis, whether as a result of commercial promotion, or harassment and vandalism. CAPTCHAs also serve an important function in rate limiting. Automated usage of a service might be desirable until such usage is done to excess and to the detriment of human users.

Like any security system, design flaws in a system implementation can prevent the theoretical security from being realized. Many CAPTCHA implementations, especially those which have not been designed and reviewed by experts in the fields of security, are prone to common attacks. Therefore, it becomes incumbent to come up with newer methods - preventive measures against hackers. Our project aims to make CAPTCHA's more secure and infallible by breaking existing CAPTCHA's and devising newer ones.

LIST OF FIGURES

FIGURE	PAGE NO.
Figure 1: Simple Text CAPTCHA by Alta Vista	1
Figure 2: Commonly Used CAPTCHAs	4
Figure 3: PHP: How it works	7
Figure 4: Some text CAPTCHAs	7
Figure 5: Audio CAPTCHA	9
Figure 6: Typical Ticketmaster CAPTCHAs	12
Figure 7: Ticketmaster CAPTCHAs after applying the algorithms	15
Figure 8: Text CAPTCHA Snapshot	16
Figure 9: Image CAPTCHA Snapshot	21
Figure 10: MAPTCHA Snapshot	23
Figure 11: Animated MAPTCHA Snapshot 1	28
Figure 12: Animated MAPTCHA Snapshot 2	29
Figure 13: Animated MAPTCHA Snapshot 3	29
Figure 14: Audio CAPTCHA Snapshot	52
Figure 15: Image Mining Algorithm	71
Figure 16: Email Hacking Prevention Using CAPTCHAs	72
Figure 17: Level 0 DFD	73
Figure 18: Level 1 DFD	74
Figure 19: Activity Diagram	75

LIST OF SYMBOLS AND ACRONYMS

API	Application Programming Interface
ASR	Automatic Speech Recognition
CAPTCHA	Completely Automated Public Turing Machines To Tell Computers and Humans Apart
CPU	Central Processing Unit
CSS	Cascading Style Sheets
DFD	Data Flow Diagram
E Commerce	Electronic Commerce
E Mail	Electronic Mail
ESP	Email Service Provider
HTML	Hypertext Markup Language
ISP	Internet Service Provider
JSP	Java Server Pages
MAPTCHA	Mathematical CAPTCHA
MD5	Message Digest Hash Algorithm 5
OCR	Optical Character Recognition
PHP	Hypertext Preprocessing
TTF	True Type Font
URL	Uniform Resource Locator

WWW

World Wide Web

Chapter – I

INTRODUCTION

1. CAPTCHAs

A CAPTCHA is a program that can generate and grade tests that humans can pass but current computer programs cannot. For example, humans can read distorted text, but current computer programs can't. The process usually involves one computer (a server) asking a user to complete a simple test which the computer is able to generate and grade. Because other computers are supposedly unable to solve the CAPTCHA, any user entering a correct solution is presumed to be human. Thus, it is sometimes described as a reverse Turing test, because it is administered by a machine and targeted to a human, in contrast to the standard Turing test that is typically administered by a human and targeted to a machine. A common type of CAPTCHA requires the user to type letters or digits from a distorted image that appears on the screen.

Submission Code:



Enter Submission Code:

Figure 1: An example of a simple Text CAPTCHA by Alta Vista

2. Applications of CAPTCHAs

CAPTCHAs have several applications for practical security, including (but not limited to):

- **Preventing Comment Spam in Blogs.** Most bloggers are familiar with programs that submit bogus comments, usually for the purpose of raising search engine ranks of some website (e.g., "buy penny stocks here"). This is called comment spam. By using a CAPTCHA, only humans can enter comments on a blog. There is no need to make users sign up before they enter a comment, and no legitimate comments are ever lost!
- **Protecting Website Registration.** Several companies (Yahoo!, Microsoft, etc.) offer free email services. Up until a few years ago, most of these services suffered from a specific type of attack: "bots" that would sign up for thousands of email accounts every minute. The solution to this problem was to use CAPTCHAs to ensure that only humans obtain free accounts. In general, free services should be protected with a CAPTCHA in order to prevent abuse by automated scripts.
- **Protecting Email Addresses From Scrapers.** Spammers crawl the Web in search of email addresses posted in clear text. CAPTCHAs provide an effective mechanism to hide your email address from Web scrapers. The idea is to require users to solve a CAPTCHA before showing your email address.
- **Online Polls.** In November 1999, <http://www slashdot.org> released an online poll asking which was the best graduate school in computer science (a dangerous question to ask over the web!). As is the case with most online polls, IP addresses of voters were recorded in order to prevent single users from voting more than once. However, students at Carnegie Mellon found a way to stuff the ballots using programs that voted for CMU thousands of times. CMU's score started growing rapidly. The next day, students at MIT wrote their own program and the poll became a contest between voting "bots." MIT finished with 21,156 votes, Carnegie Mellon with 21,032 and every other school with less than 1,000. Can the result of any online poll be trusted? Not unless the poll ensures that only humans can vote.
- **Preventing Dictionary Attacks.** CAPTCHAs can also be used to prevent dictionary attacks in password systems. The idea is simple: prevent a computer from being able to iterate through the entire space of passwords by requiring it to solve a CAPTCHA after a certain number of unsuccessful logins. This is better than the classic approach of locking an account after a sequence of unsuccessful logins, since doing so allows an attacker to lock accounts at will.

- **Search Engine Bots.** It is sometimes desirable to keep webpages unindexed to prevent others from finding them easily. There is an html tag to prevent search engine bots from reading web pages. The tag, however, doesn't guarantee that bots won't read a web page; it only serves to say "no bots, please." Search engine bots, since they usually belong to large companies, respect web pages that don't want to allow them in. However, in order to truly guarantee that bots won't enter a web site, CAPTCHAs are needed.
- **Worms and Spam.** CAPTCHAs also offer a plausible solution against email worms and spam: "I will only accept an email if I know there is a human behind the other computer." A few companies are already marketing this idea.

3. Objective and Scope of the Project

Considering the recent news that the hotmail and yahoo image CAPTCHA would be gamed, it's important to offer a wide range of CAPTCHA types. If only one CAPTCHA type is offered, spammers would only require information on the particular CAPTCHA to spam the website. If a wide range of CAPTCHA types is offered, it is much harder (and less profitable) for spammers to target websites. As long as humans are more flexible than spam bots, this should work. Secondly, the image and audio CAPTCHAs are probably hard for spammers to break, but the CAPTCHA's happen to be more CPU intensive, compared to a simple text based CAPTCHA.

Our project is not limited to developing existing CAPTCHAs. We also came up with our own implementation of some more secure CAPTCHAs.

Gmail	Yahoo!	Hotmail
clati	sl88FLwy	6HJH6CTN
2umso	77evrMF	EXXTENHK
omictieu	7pALS7	XYHNXCIR

Figure 2: Commonly used CAPTCHAs

4. Implementation

- Text CAPTCHA (basic prototype)
- Image CAPTCHA
- Simple Mathematical CAPTCHA (MAPTCHA)
- Animated MAPTCHA
- Audio CAPTCHA

5. Resources and Limitations

For our project on CAPTCHA's, the requirements are a web server (XAMPP) and supporting PHP. The client side coding is done using PHP, HTML, JavaScript. Hardware resources required were a server and a computer. Data set was derived from the study of various research papers and material available on the internet. Limitations came in the form of hosting the website as all major hosting services require a paid account. To find a free

hosting server (for testing) was a difficult task. Free hosting servers have a space constraint and all of them do not support PHP.

Chapter – II

LITERATURE SURVEY

We have used HTML, PHP and JavaScript for developing our CAPTCHAs in which we have stored the text in session, CAPTCHAs protect against spam bots, which are automated scripts that crawl on the net searching for URLs containing some kind of application forms such as forums or popular blogs and then automatically posting whatever the spammer wants everybody to know,

To implement CAPTCHA, our algorithm challenges users to rewrite a certain text presented or solve a mathematical equation. Another algorithm is to present the use with a grid of images and challenge them to select certain images based on any given criteria. Also, for visually impaired people we have audio CAPTCHAs which present a sound and challenges them to write whatever they have heard.

1. PHP

PHP (Hypertext Preprocessor) is a general-purpose scripting language originally designed for web development to produce dynamic web pages. For this purpose, PHP code is embedded into the HTML source document and interpreted by a web server with a PHP processor module, which generates the web page document. It also has evolved to include a command-line interface capability and can be used in standalone graphical applications. PHP can be deployed on most web servers and as a standalone interpreter, on almost every operating system and platform free of charge.

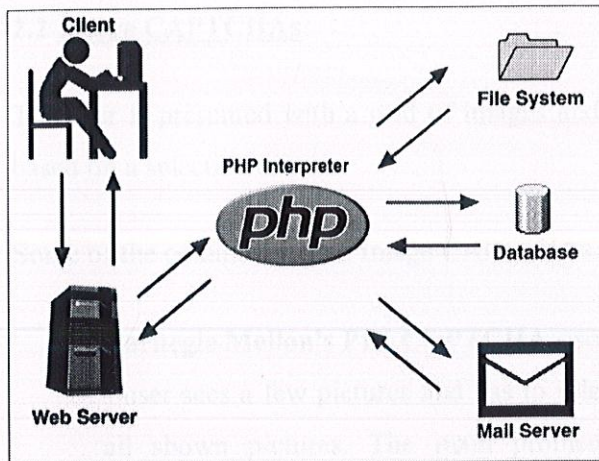


Figure 3: How it works

2. Types of CAPTCHAs

2.1 Text CAPTCHAs

Text CAPTCHA is a distorted image containing short text. It is displayed in such a format so that only human eyes can recognize the alphabets clearly. At the time of registration, such image is displayed on the form and the user is asked to write the same text in given text field. The robots fail to recognize the short text. Thus, website owners can prevent robots from registration and can ensure that all the members using free services are humans.

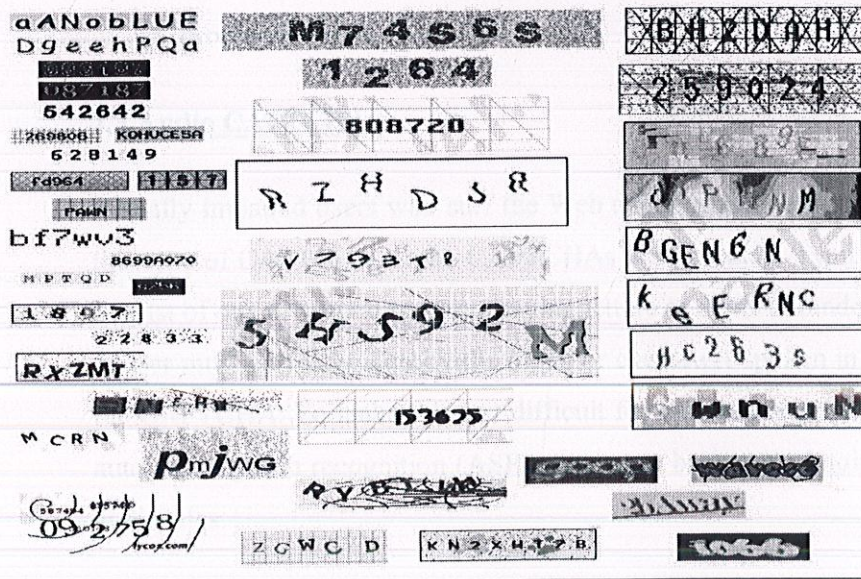


Figure 4: Some Text CAPTCHAs

2.2 Image CAPTCHAs

The user is presented with a grid of images and is required to select certain images based on a selection criteria.

Some of the commonly used Image CAPTCHAs are:

- **Carnegie Mellon's PIX CAPTCHA** - so called "naming images CAPTCHA" - user sees a few pictures and has to select a word that is appropriated to the all shown pictures. The main problem of this type of CAPTCHAs is misspelling while writing the answer and synonyms for the answer-word (for example: dog, hound, pooch). In the described case this solved by means of transferring all variants of the answer to the client side.
- **Oli Warner's KittenAuth** - in order to prove his humanity visitor has to select all animals of specified species among the proposed pictures. But the limited number of the pictures allow to recreate the picture base manually.
- **IMAGINATION - CAPTCHA** that requires two steps to be passed. At first step visitor clicks elsewhere on the picture that composed of a few images and selects in this way a single image. At second step the selected image is loaded. It is enlarged but very distorted. Also variants of the answer are loaded on the client side. The visitor should select a correct answer from the set of the proposed words.

2.3 Audio CAPTCHAs

Visually impaired users who surf the Web using screen-reading programs cannot see this type of CAPTCHA, audio CAPTCHAs were created. Typical audio CAPTCHAs consist of one or several speakers saying letters or digits at randomly spaced intervals. A user must correctly identify the digits or characters spoken in the audio file to pass the CAPTCHA. To make this test difficult for current computer systems, specifically automatic speech recognition (ASR) programs, background noise is injected into the audio files.

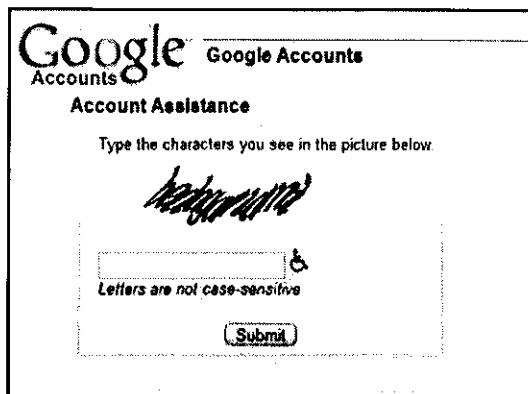
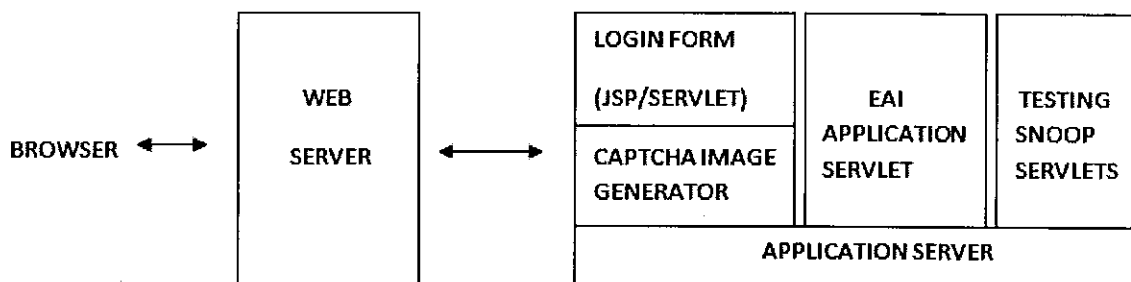


Figure 5: Audio CAPTCHA

3. CAPTCHA Authentication Process



In the above diagram:

- The browser connects to a back end application server, in order to access the desired URL the user needs to pass the authentication and authorization of the server.
- Then the HTTP request arrives at server. The server finds that the URL needs authentication and authorization before accessing it. Then, it asks the browser to redirect to the login form URL.
- The browser gets the redirect request from and then connects to the login form URL.
- The login form calls the CAPTCHA image generator to generate a random number CAPTCHA image and then send the login form (with ID and password input field

and CAPTCHA random number image) back to the browser. At the same time, the login form JSP/Servlet will store the random number in the HTTP session.

- The user inputs the user ID, password, and the random number of the CAPTCHA image. After the user clicks submit or login button, the browser sends the ID, password, and the random number to the Web server, this information is passed to the EAI application.
- The EAI application checks to ensure the user ID, password, and the random number are correct. (The EAI application can retrieve the random number from the HTTP session and compare it with the number input by the user.)
- If the details are valid, the EAI application inserts the user ID in the response HTTP header and sends the response to Web server.
- The web server gets the HTTP response from the EAI application, extracts the user ID from the HTTP header, performs the authorization for the user, and then creates a session for the user.
- The server sends a redirect request to the browser and asks the browser to redirect to the protected URL that the user inputs in Step 1.
- The browser gets the redirect request from the Web server and then connects to the protected URL. The URL request arrives at Web server again. Because the user has already passed the authentication, the server then authorizes the request and routes the URL request to the back-end application (testing snoop servlet).

4. Loopholes

- Image recognition CAPTCHAs face many potential problems which have not been fully studied. It is difficult for a small site to acquire a large dictionary of images which an attacker does not have access to and without a means of automatically acquiring new labelled images, an image based challenge does not usually meet the definition of a CAPTCHA. KittenAuth, by default, only had 42 images in its database.

- In reCAPTCHA which is used in facebook it is possible to omit anything except the letters and still you will pass the test.
- CAPTCHAs in gmail are optional on certain browsers like Firefox. This actually means that if you enter the password correct in say the second or the third attempt then you are not required to solve the CAPTCHA.
- People have written bots that are capable of doing OCR (Optical Character Recognition) in order to foil these tests. Hence the need for more complex CAPTCHAs increases.

Chapter – III

Breaking Ticketmaster CAPTCHAs

Ticketmaster is a site which sells tickets for concerts online. Some people in order to purchase bulk tickets so that they can sell them at exorbitant prices later generate bots. To prevent this, these bots need to solve the CAPTCHAs for every transaction that they make. So such CAPTCHAs need to be more secure.

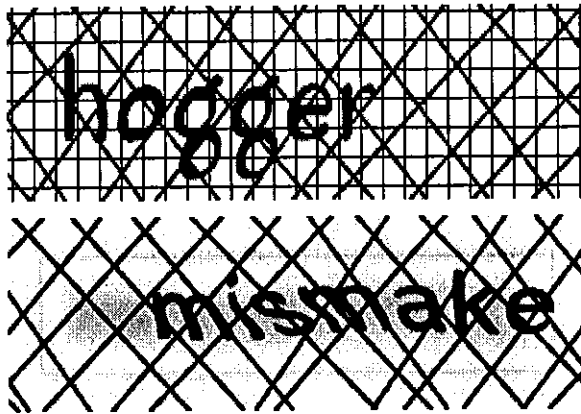


Figure 6: Typical Ticketmaster CAPTCHAs

These CAPTCHAs have certain characteristics like they have very bright backgrounds and have a lot of lines which cross over the text making it more complex.

Steps to break the CAPTCHA are as follows:

1. Background Removal

It is a two-step process.

1.1 Removing by color delta

// Let $F(x,y)$ be a binary image of size $N \times M$, s.t. $0 \leq x < N$, $0 \leq y < M$.

// F(x,y).red is the red component, etc. Assume each component ranges from 0 to 100% saturated.

```
for x = 0 to N-1
  for y = 0 to M-1
    if ( abs( F(x,y).red - F(x,y).blue ) ≥ Δ or
        abs( F(x,y).red - F(x,y).green ) ≥ Δ or
        abs( F(x,y).blue - F(x,y).green ) ≥ Δ )
      F(x,y) = white
```

1.2 Quantizing the result

- The images are then quantized to monochromatic tone in the second pass of the process, whereby the average of the R, G, B components, F(x,y).average, is taken. If F(x,y).average ≥ δ, then F(x,y).red, etc. are set to 100%, resulting in white.
- A value for δ of 40% was experimentally determined to work well.

2. Line Removal

- **Distributed probabilistic detection scheme:**

A preliminary step creates four copies of the original image, called C[1..4]. C[1] is the unchanged original image. C[2] is the original image rotated 90° clockwise. C[3] is the image rotated 180° and C[4] is the image rotated 270°.

- **Line pattern discovery:**

```
best = 0
for 5° ≤ θ ≤ 175°
  BC = 0 // black count
  TC = 0 // total count
```

```

D = 0 // Distance
while x+cos(D) < N and sin(D) < M
  if F(x+cos(D),sin(D)).average < δ
    BC = BC + 1
    TC = TC + 1
  ratio = BC / TC
  If ratio > best then
    best = ratio, best_θ = θ

```

SONAR Sweep:

- This is used to determine at what points the line cross over the text because at those points the width will be greater than the average width.
- This is necessary because different CAPTCHA lines sometimes have varying widths.
- SONAR line is a line which runs perpendicular to the value of θ for a distance of $\pm S$ units. For example if $\theta = 90^\circ$, this SONAR line would examine starting points along the line $F(x, 0 \leq y < N)$ looking in turn at pixels $F(x, y-S \leq z \leq y+S)$ along the perpendicular SONAR line.
- Each SONAR sweep maintains an array of structures. The structure elements contain the pixel coordinate (to save time later) and a Boolean indication of whether that pixel is $F(x,0).average < \delta$; whether it is black or white. Thus each SONAR sweep along the line contains a Boolean array $y-S \leq z \leq y+S$ of values.

SONAR Sweeping Process:

This process checks if the line is crossing over the text. If it does, it is not removed and if it doesn't then the line is eliminated.

```

for 0 ≤ s < S
  If sonar[s].solid and (sonar[s].R - sonar[s].L)+1 ≤ AW
    // Eliminate line at this sonar sweep from the image
    For sonar[s].L ≤ d ≤ sonar[s].R

```



```

F(sonar[s].data[d].x, sonar[s].data[d].y) = 0
// Eliminate pixels to the left (rectilinear in F)
if F(sonar[s].data[d].x-1, sonar[s].data[d].y).average  $\geq \delta$ 
    F(sonar[s].data[d].x-1, sonar[s].data[d].y) = 0
// Eliminate pixels to the right (rectilinear in F)
If F(sonar[s].data[d].x+1, sonar[s].data[d].y). average  $\geq \delta$ 
    F(sonar[s].data[d].x+1, sonar[s].data[d].y) = 0

```

- Since there are four copies of the image, in all four orientations. As a last step these are rotated back to the original orientation. At that point we combine the images back into the original:

```

for x = 0 to N-1
  for y = 0 to M-1
    F(x,y) = C[1](x,y) & C[2](x,y) & C[3](x,y) & C[4](x,y)

```

The result after applying the techniques discussed above are:

hogger
mismake

Figure 7: Result after applying the techniques

Now typical OCR techniques which include steps like preprocessing, segmentation and recognition can make out these characters and the CAPTCHA is broken

Chapter – IV

Implementation

1. Basic Text CAPTCHA

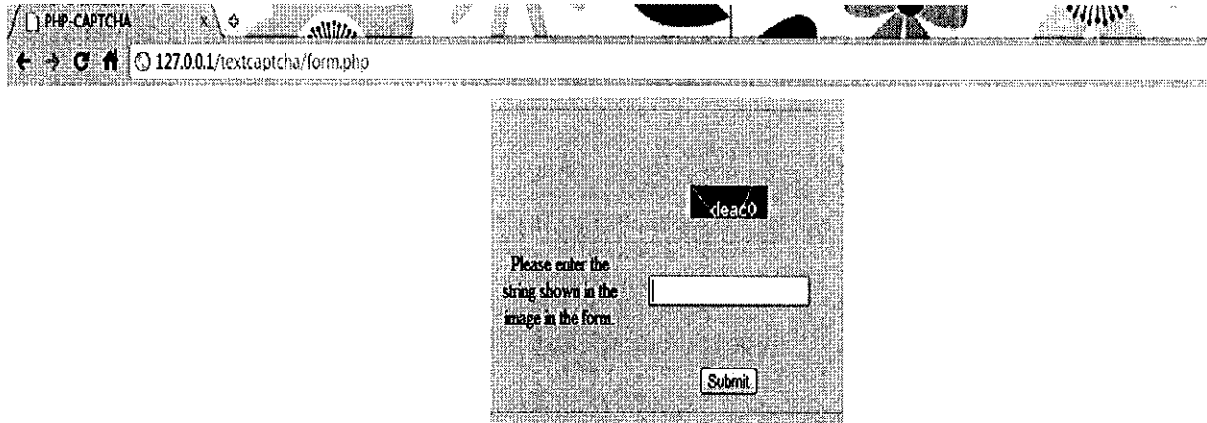


Figure 8: Text CAPTCHA Snapshot

Code Snippet:

php_captcha.php:

```
<?php
session_start();

$RandomStr = md5(microtime());// md5 to generate the random string

$ResultStr = substr($RandomStr,0,5);//trim 5 digit
```

```
$NewImage =imagecreatefromjpeg("img.jpg");//image create by existing image and  
as back ground
```

```
$LineColor = imagecolorallocate($NewImage,233,239,239);//line color
```

```
$TextColor = imagecolorallocate($NewImage, 255, 255, 255);//text color-white
```

```
imageline($NewImage,1,1,40,40,$LineColor);//create line 1 on image
```

```
imageline($NewImage,1,100,60,0,$LineColor);//create line 2 on image
```

```
imagestring($NewImage, 5, 20, 10, $ResultStr, $TextColor);// Draw a random string  
horizontally
```

```
$_SESSION['key'] = $ResultStr;// carry the data through session
```

```
header("Content-type: image/jpeg");// out out the image
```

```
imagejpeg($NewImage);//Output image to browser
```

```
?>
```

form.php

```
<?php
```

```
session_start();
```

```
?>
```

```
<HTML>
```

```
<HEAD>
```

```
<TITLE>PHP-CAPTCHA </TITLE>
```

```
<META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=iso-  
8859-1">
```

```
</HEAD>
```

```
<BODY onLoad="return focuson();">
```

```

<script language="javascript">
function focuson()
{ document.form1.number.focus()}

function check()
{
if(document.form1.number.value==0)
{
alert("Please enter your Category Name");
document.form1.number.focus();
return false;
}
}

</script>
<?php

if(isset($_REQUEST['Submit'])){
$key=substr($_SESSION['key'],0,5);
$number = $_REQUEST['number'];
if($number!=$key){
echo '<center><font face="Verdana, Arial, Helvetica, sans-serif"
color="#FF0000> Validation string not valid! Please try again!</font></center>';}
else{
echo '<center><font face="Verdana, Arial, Helvetica, sans-serif" color="#66
Your string is valid!</font></center>';}
}
?>
<form name="form1" method="post" action="form.php" onsubmit="return
check();">
<table width="342" align="center" cellspacing="0" bgcolor="#D4D0C8">

```

```

<tr>
  <td colspan="4" align="center"><hr></td>
</tr>
<tr>
  <td width="8" align="center">&nbsp;</td>
  <td width="330" align="right" valign="top">&nbsp;</td>
  <td width="330" align="right" valign="top">&nbsp;</td>
  <td width="2" align="center">&nbsp;</td>
</tr>
<tr>
  <td align="center">&nbsp;</td>
  <td align="right" valign="top">&nbsp;</td>
  <td align="right" valign="top">&nbsp;</td>
  <td align="center">&nbsp;</td>
</tr>
<tr>
  <td align="center">&nbsp;</td>
  <td align="center">&nbsp;</td>
  <td align="center"></td>
  <td align="center">&nbsp;</td>
</tr>
<tr>
  <td align="center">&nbsp;</td>
  <td align="center">&nbsp;</td>
  <td align="center">&nbsp;</td>
  <td align="center">&nbsp;</td>
</tr>
<tr>
  <td align="center">&nbsp;</td>

```

```
<td align="center"> Please enter the string shown in the image in the form.<br></td>
```

```
<td align="center"><input name="number" type="text" id="number"></td>
```

```
<td align="center">&nbsp;</td>
```

```
</tr>
```

```
<tr>
```

```
<td align="center">&nbsp;</td>
```

```
<td align="center">&nbsp;</td>
```

```
<td align="center">&nbsp;</td>
```

```
<td align="center">&nbsp;</td>
```

```
</tr>
```

```
<tr>
```

```
<td align="center">&nbsp;</td>
```

```
<td align="center">&nbsp;</td>
```

```
<td align="center"><input name="Submit" type="submit" value="Submit"></td>
```

```
<td align="center">&nbsp;</td>
```

```
</tr>
```

```
<tr>
```

```
<td colspan="4" align="center"><hr></td>
```

```
</tr>
```

```
</table>
```

```
</form>
```

```
</BODY>
```

```
</HTML>
```

2. Image CAPTCHA



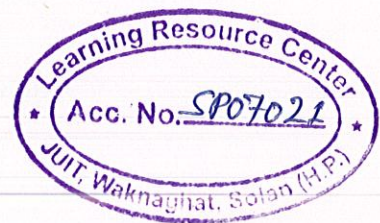
Figure 9: Image CAPTCHA Snapshot

Code Snippet:

Image.php

```
<html>
<title>Example Service: Signup Page</title>
<head>

<script type="text/javascript">
function HumanCheckComplete(isHuman)
{
    if (isHuman)
    {
```



```
        formElt = document.getElementById("mainForm");
        formElt.submit();
    }
    else
    {
        alert("Please correctly identify the cats.");
        return false;
    }
}
```

</script>

</head>

<body>

<h2>

Image CAPTCHA

</h2>

<p>

<form action="ExampleService-PHP.php" method="get" id="mainForm">

User Name: <input type="text" name="UserName">

Favorite Color: <input type="text" name="FavoriteColor">

<script type="text/javascript"

src="//challenge.asirra.com/js/AsirraClientSide.js"></script>

<script type="text/javascript">

asirraState.SetEnlargedPosition("top");

// You can control the aspect ratio of the box by changing this
constant

```
        asirraState.SetCellsPerRow(6);  
    </script>  
  
<br><input type="button" value="Submit"  
onclick="javascript:Asirra_CheckIfHuman(HumanCheckComplete)">  
  
</form>  
</body>  
</html>
```

3. MAPTCHA

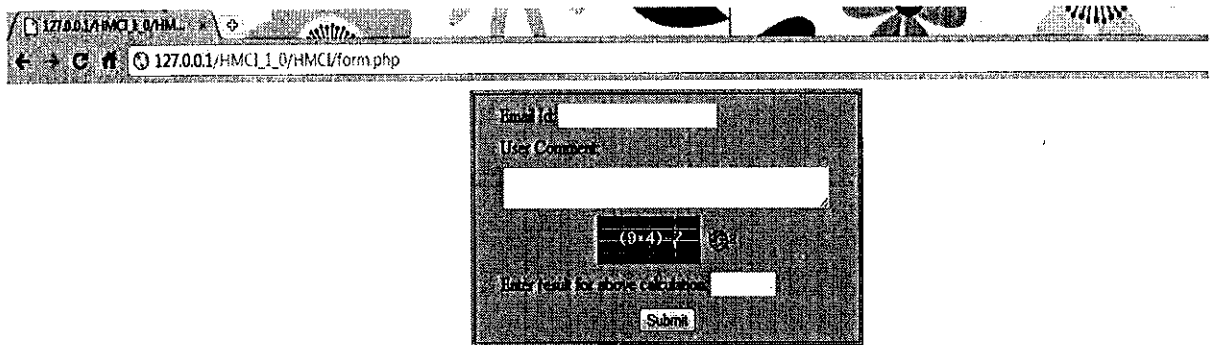


Figure 10: MAPTCHA Snapshot

Code Snippet:

captcha.php

```
<?php  
session_start();  
  
$num1=rand(0,9);
```

```
$num2=rand(0,9);
```

```
$num3=rand(0,9);
```

```
$op1=rand(0,2);
```

```
$op2=rand(0,2);
```

```
if($op1=="0") $op3="+";
```

```
if($op1=="1") $op3="-";
```

```
if($op1=="2") $op3="*";
```

```
if($op2=="0") $op4="+";
```

```
if($op2=="1") $op4="-";
```

```
if($op2=="2") $op4="*";
```

```
if($op3=="*")
```

```
{
```

```
if($op4=="*")
```

```
    $mcap>(".$num1.$op3.$num2.")$.op4.$num3;
```

```
else if($op4=="+" || $op4=="-")
```

```
    $mcap(".$num1.$op3.$num2.")$.op4.$num3;
```

```
}
```

```
else if($op3=="+" || $op3=="-")
```

```
{
```

```
if($op4=="+" || $op4=="-")
```

```
    $mcap(".$num1.$op3.$num2.")$.op4.$num3;
```

```
else if($op4=="*")
```

```
    $mcap=$num1.$op3."(".$num2.$op4.$num3.)";
```

```
}
```

```
if($op3=="+" && $op4=="+")
```

```
    $res=($num1+$num2)+$num3;
```

```
else if($op3=="+" && $op4=="-")
```

```
    $res=($num1+$num2)-$num3;
```

```
else if($op3=="+" && $op4=="*")
```

```
$res=$num1+($num2*$num3);
```

```
else if($op3=="-" && $op4=="+")
```

```
$res=($num1-$num2)+$num3;
```

```
else if($op3=="-" && $op4=="-")
```

```
$res=($num1-$num2)-$num3;
```

```
else if($op3=="-" && $op4=="*")
```

```
$res=$num1-($num2*$num3);
```

```
else if($op3=="*" && $op4=="+")
```

```
$res=($num1*$num2)+$num3;
```

```
else if($op3=="*" && $op4=="-")
```

```
$res=($num1*$num2)-$num3;
```

```
else if($op3=="*" && $op4=="*")
```

```
$res=($num1*$num2)*$num3;
```

```
$_SESSION['result'] = $res;
```

```
$secure = $_SESSION['result'];
```

```
$a=rand(150,255);
```

```
$width = 100;
```

```
$height = 40;
```

```
$image = ImageCreate($width, $height);
```

```
$linecol = ImageColorAllocate($image, rand(155,200), rand(155,200), rand(155,200));
```

```
$textcol = ImageColorAllocate($image, rand(200,255),rand(200,255),rand(200,255));
```

```
$bgcol = ImageColorAllocate($image, rand(0,155),rand(0,155), rand(0,155));
```

```
ImageFill($image, 0, 0, $bgcol);
```

```
//Add randomly generated string in white to the image
```

```
ImageString($image, 10, 20, 10, $mcap, $textcol);
ImageRectangle($image,0,0,$width-1,$height-1,$bgcol);
imageline($image, 0, $height-30, $width, $height-30, $linecol);
imageline($image, 0, $height-20, $width, $height-20, $linecol);
imageline($image, 0, $height-10, $width, $height-10, $linecol);

imagedashedline($image, $width/2, 0, $width/2, $height, $linecol);
imagedashedline($image, $width-25, 0, $width-25, $height, $linecol);
imagedashedline($image, $width-75, 0, $width-75, $height, $linecol);

header("Content-Type: image/jpeg");
ImageJpeg($image);
ImageDestroy($image);
?>
```

form.php

```
<?php
session_start();
?>
<script type="text/javascript">
function refresh()
{
    var ran=Math.floor(Math.random()*10);
    var path=document.getElementById("imgsrc").src;
    document.getElementById("imgsrc").src=path+"?rand="+ran;
}
</script>

<form name="form1" method="post" action="validate.php">
```

```

<table border="1" width="380" align="center" cellspacing="2" cellpadding="2"
bgcolor="#BEBEBE"><tr><td>
<table bgcolor="#BEBEBE" align="center">
<tr>
<td>Email Id:<input type="text"></td>
</tr>
<tr>
<td>User Comment:</td>
</tr>
<tr>
<td><textarea cols=37 rows=2></textarea></td>
</tr>
<tr>
<td align="center">&nbsp;&nbsp;&nbsp;
<a style="color: green; font-size: 10px; text-decoration: none;"
href="http://www.hscripts.com">H</a></td>
</tr>
<tr>
<td>Enter result for above calculation:<input name="number" type="text"
size="5"></td>
</tr>
<td align="center">
<input name="Submit" type="submit" value="Submit"></td> </tr>
</table>
</td>
</tr>
</table>
</form>

```

validate.php

```
<form name="form1" method="post" action="form.php">
<div align="center">
<input name="Submit" type="submit" value="back"></div>
</form>
<div align="center">
<?php
@session_start();
$key=$_SESSION['result'];
$imag = $_POST['number'];
//echo "$key===== $imag";
if($imag==$key)
{
    echo ("Verification success");
}
else{
    echo "You have entered wrong verification code!!<br>
        Please go back and enter proper value.";}
?>
</div>
```

4. Animated MAPTCHA

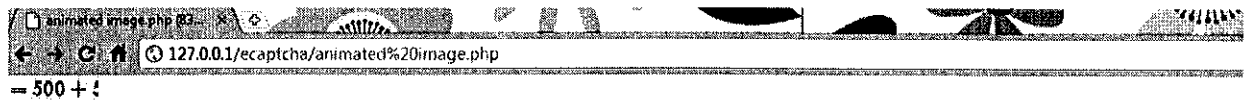


Figure 11: Animated MAPTCHA Snapshot 1

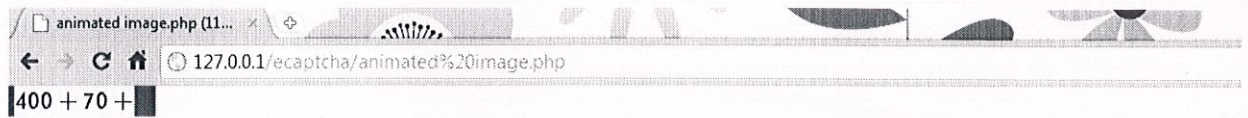


Figure 12: Animated MAPTCHA Snapshot 2



Figure 13: Animated MAPTCHA Snapshot 3

Code Snippet:

animated image.php

```
<?php  
  
header('Content-Type: image/gif');  
header('Cache-control: no-cache, no-store');  
  
# Config  
  
$height = 24;
```

```
/* Image height (pixels). Width and number of frames are automatically set
*/
```

```
$delay = 8;
```

```
/* delay = Time of visualization of each frame 1/100 sec.
```

```
delay = 4 hundredth of second means 25 fps (frames per second) High values = slow
animation.
```

```
Firefox can correctly visualize 25 fps. Internet Explorer cannot visualize more than 12
fps (delay = 8).
```

```
*/
```

```
$font = 'sans.ttf';
```

```
/* default font, You can change, but it should be not correctly visualized
```

```
*/
```

```
$ttf = 1;
```

```
/* ttf = 1 TTF font is used
```

```
ttf = 0 TTF font not used. 16 x 100 pixel steady image
```

```
*/
```

```
$anim_mode = 3;
```

```
/* anim_mode
```

```
0 pulsing
```

```
1 rolling balls
```

```
2 sliding sum
```

```
3 random animation
```

```
*/
```

```
#####
```

```
if ($anim_mode == 3) {$anim_mode = rand(0,2)}
```



```

// // Set font size
$fontsize = $height * 0.7;
$tot_frames = $height;

srand((double)microtime()*1000000);
$string = rand(1,9)*100; //Genero il primo numero
$string2=rand(1,9)*10; //Genero il secondo numero
$string4=rand(1,9); //Genero il terzo numero
$string3="$string + $string2 + $string4";
$somma = $string3;

$pass=$string+$string2+$string4;
$somma2 = "$string3 = $string3";

// Set image width
$textbox = imagettfbbox($fontsize, 0, $font, $somma) or die('Error in imagettfbbox
function');

if ($ttf == 0)
{
$width = 100;
$height = 16;
$delay = 12;
if ($anim_mode == 2)

{ $width = 80;
}

}

else {

```

```

if($anim_mode < 2)

{ $width = (abs($textbox[4] - $textbox[0]))*1.1;
}
else {
$width = (abs($textbox[4] - $textbox[0]))*0.8;
}

}

$х = $width*0.035;
$у = $height - $height/4;

$anim_len = 10;
$start_dummy = rand(0,10);
$end_dummy = rand($start_dummy+$anim_len,30);

$files = array();

function get_gif_header($gif_data) {
$header = array();
$header["signature"] = substr($gif_data,0,3);
$header["version"] = substr($gif_data,3,3);
$header["logical_screen_width"] = substr($gif_data,6,2);
$header["logical_screen_height"] = substr($gif_data,8,2);
$header["packed"] = substr($gif_data,10,1);
$header["background_color_index"] = substr($gif_data,11,1);
$header["pixel_aspect_ratio"] = substr($gif_data,12,1);
$packed = ord($header["packed"]);
if(($packed >> 7) & 0x1) {

```

```

    $gct = $packed & 3;
    $gct_size = 3 * pow(2,$gct+1);
    $header["global_color_table"] = substr($gif_data,13,$gct_size);
}
return $header;
}

```

```

function strip_gif_header($gif_data) {
    $without_header = "";
    $header_len = 0;
    $header = get_gif_header($gif_data);
    foreach ($header as $k=>$v)
        $header_len += strlen($v);
    return substr($gif_data,$header_len,strlen($gif_data)-$header_len);
}

```

```

function get_gif_image_data($gif_data) {
    $no_header = strip_gif_header($gif_data);
    $no_header = substr($no_header,0,strlen($no_header)-1);
    return $no_header;
}

```

```

function get_gif_image_descriptor($image_data) {
    $header = array();
    $header["image_separator"] = substr($image_data,0,1);
    $header["image_left_position"] = substr($image_data,1,2);
    $header["image_top_position"] = substr($image_data,3,2);
    $header["image_width"] = substr($image_data,5,2);
    $header["image_height"] = substr($image_data,7,2);
    $header["packed"] = substr($image_data,9,1);
    $packed = ord($header["packed"]);
    if (($packed >> 7) & 0x1) {

```

```

    $lct = $packed & 3;
    $lct_size = 3 * pow(2,$lct+1);
    $header["local_color_table"] = substr($image_data,10,$lct_size);
}
return $header;
}
function strip_gif_image_descriptor($imgdata) {
    $descriptor = get_gif_image_descriptor($imgdata);
    $len = 0;
    foreach ($descriptor as $k=>$v)
        $len += strlen($v);
    return substr($imgdata,$len,strlen($imgdata)-$len);
}
function make_gifanim($gifs) {
    global $delay;
    $head0 = get_gif_header($gifs[0]);
    $head0["packed"] = chr( ord($head0["packed"]) & (7 << 4) );
    $head0["background_color_index"] = chr(0);
    $head0["pixel_aspect_ratio"] = chr(0);
    unset($head0["global_color_table"]);
    $anim_gif = implode("", $head0);
    $extra_info = array( chr(0x21), chr(0xff) ,chr(0x0B), "NETSCAPE2.0",chr(0x03),
chr(0x01), chr(0x00).chr(0x00), chr(0x00) );
    $anim_gif .= implode("", $extra_info);
    foreach ($gifs as $gif) {
        $header = get_gif_header($gif);
        $imgdata = get_gif_image_data($gif);
        $image_header = get_gif_image_descriptor($imgdata);
        $image_only = strip_gif_image_descriptor($imgdata);
        $control_block = array();
        $control_block["extension_introducer"] = chr(0x21);

```

```

$control_block["graphic_control_label"] = chr(0xF9);
$control_block["block_size"] = chr(4);
$control_block["packed"] = chr(0);
$control_block["delay"] = chr($delay).chr(0);
$control_block["transparent_color_index"] = chr(0);
$control_block["terminator"] = chr(0);
if (!isset($image_header["local_color_table"]) &&
isset($header["global_color_table"])) {
    $image_header["local_color_table"] = $header["global_color_table"];
    $size_gct = (ord($header["packed"]) & 3);
    $image_header["packed"] = chr( ord($image_header["packed"]) | (0x1 << 7) |
($size_gct) );
}
$anim_gif .= implode("", $control_block).implode("", $image_header).$image_only;
}
$anim_gif .= chr(0);
return $anim_gif;
}

```

```

$cur2_x = $tot_frames*2;

```

```

for ($f=0;$f<$tot_frames;$f++) {
    $im = imagecreate($width, $height)
    or die("Cannot Initialize new GD image stream");

```

```

// Some colors

```

```

$white = imagecolorallocate($im, 255, 255, 255);
$gray = imagecolorallocate($im, 238, 238, 238);
$gray2 = imagecolorallocate($im, 200, 200, 200);
$black = imagecolorallocate($im, 0, 0, 0);
$red = imagecolorallocate($im, 255, 0, 0);

```

```

$green = imagecolorallocate($im, 0, 255, 0);
$blu = imagecolorallocate($im, 0, 0, 255);
$lightblu = imagecolorallocate($im, 221, 231, 244);

// Background color - You can change it
$colore_sfondo = $gray;

// Textcolor - You can change it
$colore_testo = $black;

// Pulsing animation color - You can change it
$colore_anim1 = $red;

// Rolling animation color - You can change it
$colore_anim2 = $blu;

ImageFill($im, 0, 0, $colore_sfondo);

if ($f > $start_dummy && $f < $end_dummy)

ImageFill($im, 0, 0, $colore_sfondo);

if ($anim_mode == 0) {

if ($tff == 1)
{ ImageTtfText($im, $fontsize, 0, $x, $y, $colore_testo, $font, $somma);
}
else {
ImageString($im, 4, 3, 1, $somma, $black);
}
}

```

```

}

if ($cur_x < $tot_frames) {
  imagefilledrectangle($im, -2, 0, $cur_x - 2, $height, $colore_anim1);
  imagefilledrectangle($im, $cur_x + ($width -
$tot_frames + 1), 0, $width + 2, $width, $colore_anim1);
}
else {
  imagefilledrectangle($im, -2, 0, $cur2_x - 2, $height, $colore_anim1);
  imagefilledrectangle($im, $cur2_x + ($width -
$tot_frames + 1), 0, $width + 2, $width, $colore_anim1);
}

$cur_x = $cur_x + 2;
$cur2_x = $cur2_x - 2;

} else { if ($anim_mode == 1) {

if ($tff == 1)
{ ImageTtfText($im, $fontsize, 0, $x, $y, $colore_testo, $font, $somma);
}
else {
  ImageString($im, 4, 3, 1, $somma, $black);
}

imagefilledellipse($im, $cur3_x, $height/2, $height, $height, $colore_anim2);
  imagefilledellipse($im, $width/2 + $cur3_x, $height/2, $height, $height,
$colore_anim2);
  imagefilledellipse($im, $width + $cur3_x, $height/2, $height, $height, $colore_anim2);
}
}

```

```
    imagefilledellipse($im,    -$width/2+$scur3_x,    $height/2,    $height,    $height,  
$colore_anim2);
```

```
    $scur3_x = $scur3_x + ($width/$height);
```

```
} else {
```

```
if ($ttf == 1)
```

```
{ ImageTtfText($im,$fontsize,0,$x-$scur4_x,$y,$colore_testo,$font,$somma2);  
}
```

```
else {
```

```
ImageString($im, 4, $x-$scur4_x, 1, $somma2, $black);
```

```
}
```

```
$scur4_x = $scur4_x + ($width/$height)/0.64;
```

```
}
```

```
}
```

```
ob_start();
```

```
imagegif($im);
```

```
$files[] = ob_get_clean();
```

```
imagedestroy($im);
```

```
}
```

```
echo make_gifanim($files);
```

```
session_start();
```

```
$_SESSION['code'] = $pass
```

```
?>
```


image.php

```
<?php

#http://rodomontano.altervista.org/engcaptcha.php

# Config

/* Set thick colored lines in background.
   0 no linear background.
   1 linear background.
*/
$LinearBackground = 0;

/* Set granular noise.
   0 no granular noise.
   1 granular noise.
*/
$GranularNoise = 0;

/* Set linear noise.
   0 no square noise.
   1 square noise.
*/
$LinearNoise = 0;

/* Set square noise .
   0 no square noise.
   1 square noise.
*/
$SquareNoise = 0;
```

```
/* Horizontal Character distortion.
```

```
0 no distortion.
```

```
1 distortion.
```

```
*/
```

```
$distortion_hor = 0;
```

```
/* Vertical Character distortion.
```

```
0 no distortion.
```

```
1 distortion.
```

```
*/
```

```
$distortion_ver = 0;
```

```
/* Character color select.
```

```
0 All characters are black.
```

```
1 random color.
```

```
*/
```

```
$CharacterColorMode = 0;
```

```
/* subset of Security Code
```

```
0 User has to type the whole code.
```

```
1 User has to type a subset of Code: only red characters
```

```
NOTE - This option is obviously not compatible with random color of characters
```

```
*/
```

```
$subset = 0;
```

```
/* Character Phase Shift
```

```
0 no staggering.
```

```
1 staggering.
```

```
*/  
$staggering = 0;
```

```
/* Character size select.
```

```
0.5 character size randomizes from 50 to 100%.
```

```
0.6 character size randomizes from 60 to 100%.
```

```
etc
```

```
1 fixed size.
```

```
*/
```

```
$CharacterSizeMode = 1;
```

```
/* Character rotation angle. The script will randomize the rotation of the  
characters between this angle in degrees, positive and negative. Set to  
zero for no rotation.
```

```
0 no rotation
```

```
20 rotation of 20°
```

```
*/
```

```
$CharacterRotationMode = 0;
```

```
/* Default font. You can change it. Font must be in same directory of script
```

```
*/
```

```
$font = 'sans.ttf';
```

```
/* Type of fonts used. Fonts must be in same directory of script
```

```
0 no random font. Default font is always loaded
```

```
1 random fonts
```

```
*/
```

```
$randomfont = 0;
```

```

$fontlist = array
("sans.ttf","box.ttf","ball.ttf","shades.ttf","outlined.ttf","sp.ttf","dayplanner.ttf");

/* number and type of characters in the security code.
*/
$Codelength = 6;
$Characters = "23456789abcdefghijklmnopqrstuvwxy";

/* Set image height (width is automatically set)
*/
$height = 60;

/* Set Background color mode
0 White
1 Grey
2 random shading background
*/
$BackgroundColorMode = 0;

#####

// Create Securitycode
for ($i = 0; $i < $Codelength; $i++) {
    $SecurityCode[$i] = substr($Characters,mt_rand(0,strlen($Characters) - 1),1);
    $pass=$pass.$SecurityCode[$i];
    $rand = mt_rand(0,1);
    If ($i == 1) {$rand = 1;}
    if ($rand > 0)
    {$pass2=$pass2.$SecurityCode[$i];
    $select[$i] = $i;

```

```

    }
    else
    {
        $select[$i] = -1;
    }
}

if ($randomfont > 0) {$font = $fontlist[mt_rand(0,6)];}

// Set font size
$fontsize = $height * 0.6;

// Set Width and Create image
$textbox2 = imagettfbbox($fontsize, 0, $font, $pass) or die('Error in imagettfbbox
function');
$width = (abs($textbox2[4] - $textbox2[0]))*1.25;
$im = imagecreatetruecolor($width, $height);

// Create some colors
$white = imagecolorallocate($im, 255, 255, 255);
$grey = imagecolorallocate($im, 238, 238, 238);
$black = imagecolorallocate($im, 0, 0, 0);
$red = imagecolorallocate($im, 255, 0, 0);

// Create background

ImageFill($im, 0, 0, $grey);

if ($BackgroundColorMode < 1) {ImageFill($im, 0, 0, $white);}

```

```

if ($BackgroundColorMode > 1) {

// colors to fade
$red_start = mt_rand(0,255);
$red_end   = mt_rand($red_start,255);

$green_start = mt_rand(0,255);
$green_end   = mt_rand($green_start,255);

$blue_start = mt_rand(0,255);
$blue_end   = mt_rand($blue_start,255);

function dif ($start,$end)
{
    if ($start >= $end)
        $dif = $start - $end;
    else
        $dif = $end - $start;

    return $dif;
}

function draw($start,$end,$pos,$step_width)
{
    if ($start > $end)
        $color = $start - $step_width * $pos;
    else
        $color = $start + $step_width * $pos;

    return $color;
}

```

```

$dif_red = dif($red_start,$red_end);
$dif_green = dif($green_start,$green_end);
$dif_blue = dif($blue_start,$blue_end);

$step_red = $dif_red / $width;
$step_green = $dif_green / $width;
$step_blue = $dif_blue / $width;

$height = $height-1;
for ($pos=0; $pos<=$width; $pos++)
{
    $color = ImageColorAllocate($im,draw($red_start,$red_end,$pos,$step_red),
    draw($green_start,$green_end,$pos,$step_green),
    draw($blue_start,$blue_end,$pos,$step_blue));

    imageline($im,$pos,"0",$pos,$height,$color);
}

$height = $height+1;

}

/* generate colored thick random lines in background */

if ($LinearBackground > 0) {
    imagesetthickness($im, $height/10);
    for( $i=0; $i<$width*$height/200; $i++ ) {

```

```

        $Color = imagecolorallocate($im, mt_rand(100,250), mt_rand(100,250),
mt_rand(100,250));
        imageline($im, mt_rand(0,$width), mt_rand(0,$height),
mt_rand(0,$width), mt_rand(0,$height), $Color);
    }
}
$x = $width/18;
for ($i = 0; $i < $Codelength; $i++) {
    $Color = $black;
    if ($CharacterColorMode > 0) $Color = imagecolorallocate($im, mt_rand(0,250),
mt_rand(0,250), mt_rand(0,250));

    $textbox = imagettfbbox($fontsize, 0, $font, $SecurityCode[$i]) or die('Error in
imagettfbbox function');
    $y = ($height - $textbox[5])/2;
    $w = abs($textbox[4] - $textbox[0]);

    $Size = mt_rand($fontsize*$CharacterSizeMode,$fontsize);
    $Angle = mt_rand(-$CharacterRotationMode,$CharacterRotationMode);
    if ($staggering > 0) {
        $x = $x + rand($w-$w/18 , $w+$w/18);
        $y = rand($y-($height/6) , $y+($height/6));

    if ($subset < 1) {ImageTtfText($im,$Size,$Angle,$x-
$w,$y,$Color,$font,$SecurityCode[$i]); }
    else {

if ($select[$i] > -1) {ImageTtfText($im,$Size,$Angle,$x-
$w,$y,$red,$font,$SecurityCode[$i]); }
        else {ImageTtfText($im,$Size,$Angle,$x-
$w,$y,$Color,$font,$SecurityCode[$i]);}

```



```

}
}
else {

$y = $height - $height/4;
$x = $x + $w*1.1;
if ($subset < 1) {ImageTtfText($im,$Size,$Angle,$x-
$w,$y,$Color,$font,$SecurityCode[$i]); }
else {

if ($select[$i] > -1) {ImageTtfText($im,$Size,$Angle,$x-
$w,$y,$red,$font,$SecurityCode[$i]); }
else {ImageTtfText($im,$Size,$Angle,$x-
$w,$y,$Color,$font,$SecurityCode[$i]);}

}

}
}

//Vertical distortion amplitude and frequency,
// good values $sampl_y = 5 $freq_y = 10; 0 for no vertical distortion
$sampl_y = 5;
$freq_y = 10;

//Horizontal distortion amplitude and frequency,
// good values $sampl_x = 5 $freq_x = 5; 0 for no vertical distortion
$sampl_x = 5;
$freq_x = 10;
if ($distortion_ver > 0) {

```

```

//Apply vertical distortion
for ($i=0;$i<$width;$i+=2){
    imagecopy($im,$im,
        $xx+$i-2,$yy+sin($i/$freq_y)*$sampl_y, //dest
        $xx+$i,$yy, //src
        2,$height);
    }
}

if ($distortion_hor > 0) {
//Apply horizontal distortion
for ($i=0;$i<$height;$i+=1){
    imagecopy($im,$im,
        $xx+sin($i/$freq_x)*$sampl_x,$yy+$i-1, //dest
        $xx,$yy+$i, //src
        $width,1);
    }
}

// Apply square noise
if ($SquareNoise > 0) {
imagesetthickness($im, 1);
for($i = 0; $i <= $width; $i += $height/5) {
    @ImageLine($im, $i, 0, $i, $height, $black);
    }
for($i = 0; $i <= $height; $i += $height/5) {
    @ImageLine($im, 0, $i, $width, $i, $black);
    }
}

// Apply linear noise
if ($LinearNoise > 0) {

```

```

imagesetthickness($im, 1);
    for( $i=0; $i<$height; $i++ ) {
        imageline($im, mt_rand(0,$width), mt_rand(0,$height),
mt_rand(0,$width), mt_rand(0,$height), $black);
    }
}

```

```
// Apply granular noise
```

```

if($GranularNoise > 0) {
    for ($i=1;$i<($width*$width/10);$i++)
    {
        $cor_x = mt_rand(1,$width);
        $cor_y = mt_rand(1,$height);
        imagesetpixel($im,$cor_x,$cor_y,$black);
    }
}

```

```
// make image
```

```

header('Content-Type: image/jpeg');
    imagejpeg($im);
    imagedestroy($im);

```

```
//store code to verify
```

```

if($subset > 0) {$pass = $pass2;}
session_start();
$_SESSION['code'] = $pass
?>

```

page.php

```
<script language='JavaScript'>
  <!--
    function setFocus() {
      document.verifica.key.focus();
    }

  -->
</script>

<?
session_start();
$key=$_POST['key'];
?>

<html>
  <head>
    <meta http-equiv="content-type" content="text/html; charset=windows-1250">
    <title></title>
  </head>
  <body onLoad="setFocus()">
    <img src='image.php'>
    <form action="page.php" method="post" name="verifica" onsubmit="setFocus();">
    <input type="text" name="key" size=17 maxlength=15 ><br>
    <input type="submit" value=" Test Code ">
  </form>
</body>
</html>
```

```
<?
//print $stronzo;
if($key)
{
if($key==$code){?>
<script language="javascript">
<!--
alert("Exact Code!!!");

//-->
</script>
<?
}
else
{
?>
<script language="javascript">
<!--
alert("Sorry...Wrong Code!!! Exact code was <?print $code;?>. Try again.");
//-->
</script>
<?}
}
?>
```

5. Audio CAPTCHA

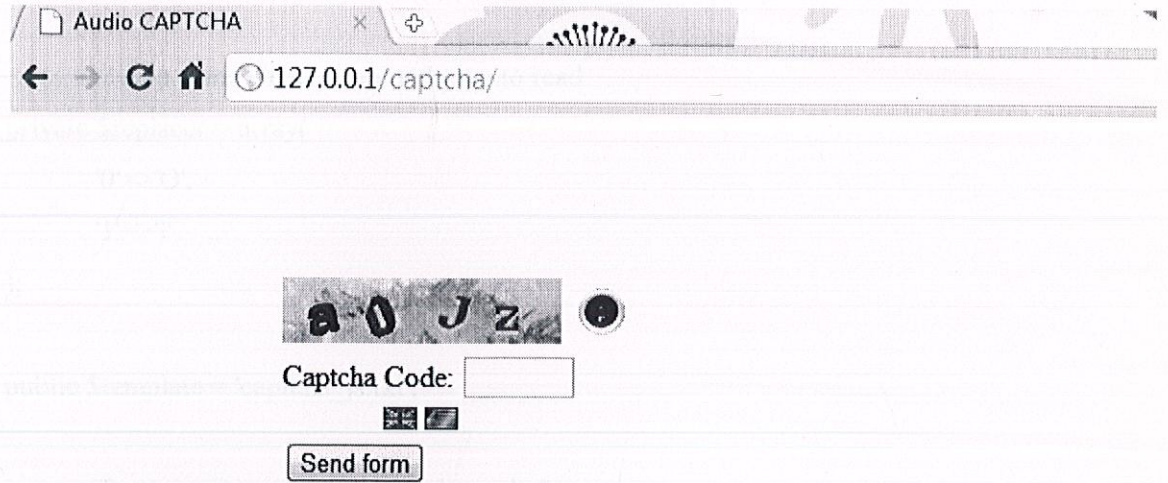


Figure 14: Audio CAPTCHA Snapshot

Code Snippet :

mp3captchaform.php

```
<?php
```

```
class mp3captcha {
```

```
    // default language
```

```
    public $language = 'uk';
```

```
    // Sample path: sounds/en/a.mp3
```

```
    // use charmap.php in sound dirs to map sounds
```

```
    public $mapping = false;
```

```
    // captcha characters
```

```

private                               $chars                               =
'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ23456789';

// map some characters confusing to read
private $replacer = array(
    '0'=>'O',
    '1'=>'l'
);

public $template = 'captcha_js.txt';

/* session name for the captcha code */
/* External change, set in image and form file to match */
public $session = 'captcha_mp3';
/* captcha code length */
public $codelength = 4;

/* ttf fonts to use - empty is autoscan fontdir */
public $fonts = array();

/* image width and height = fixed in class */
/* External change, set in image and form file to match */
public $width = 150;
public $height = 35;
/* image output type = gif jpg png */
public $type = 'jpg';
/* image transparant for backgroundless captcha */
public $transparant = 'FFFFFF';

public $formkey = 'captcha_code';

```

```

        /* bg images - empty and $backgrounddir exists = scan auto */
public $backgrounds = array();

        /* font size */
public $fontsize = 20;

        /* font colors */
public $colors = array('FF0000', '996600', '006699', '0000FF');

        /* font shade colors */
public $shades = array('FFFF00', '111111');
public $shadesize = 2;

        /* font rotation max - 0 till 60 */
public $rotate = 30;

        /* block on dnsbl ( deny ip in spam blacklist )
for more info on dnsbl lists see http://www.moensted.dk/spam/
http://www.sdsc.edu/~jeff/spam/cbc.html
http://www.declude.com/Articles.asp?ID=97
example:
array('zen.spamhaus.org','bl.spamcop.net','list.dsbl.org','tor.ahbl.org','opm.tornevall.org');
*/
public $forbidden = '403';
public $dnsbl = array();

        /* url and install dir will be set auto - override possible */
public $url = "";
public $installdir = "";

        /* private values */
private $mp3str = "";
private $mp3tag = "";
private $ccode = "";

```



```

private $deflang = "";
private $code = "";

        /* dir names set private - do not change the install structure */
private $dnsblname = 'dnsblsession';
private $templatedir = 'template';
private $sounddir = 'sounds';
private $backgrounddir = 'backgrounds';
private $fontdir = 'fonts';

/*****
*****
* @ public function mp3captcha()
* @ sets default language + var with captcha session value
*****
*****/

public function __construct($code = "") {
    if (!session_id()) {
        @session_start();
    }
    if ($code != "") {
        $this->ccode = $code;
    }
    $this->deflang = $this->language;
    $this->instaldir = str_replace("\\", '/', __FILE__);
    $this->instaldir = str_replace('/include/' . basename(__FILE__), "", $this-
>instaldir);
    if (substr($this->instaldir, -1) == '/') {
        $this->instaldir = substr($this->instaldir, 0, -1);
    }
}
}

```

```

/*****
*****
* @ public function mp3stitch()
* @ Couple soundfiles to the captcha code characters
*****
*****/

public function mp3stitch($code = "") {
    if ($code != "") {
        $this->cocode = $code;
    } else if ($this->cocode == "" && isset($_SESSION[$this->session])) {
        $this->cocode = $_SESSION[$this->session];
    }
    if ($this->cocode != "") {
        $mp3s = array();
        $this->cocode = strtolower($this->cocode);
        $this->sounddir = $this->instaldir . '/' . $this->sounddir;
        if (substr($this->sounddir, -1) == '/') {
            $this->sounddir = substr($this->sounddir, 0, -1);
        }
        // Choose language dir
        $sdir = $this->sounddir . '/' . strtolower($this->language);
        // If not present choose default language dir
        if (strlen($this->language) != 2 || !is_dir($sdir)) {
            $sdir = $this->sounddir . '/' . $this->deflang;
        }
        $scharmap = array();
        if ($this->mapping && is_file($sdir . '/charmap.php')) {
            include_once($sdir . '/charmap.php');
        }
        for ($i = 0; $i < strlen($this->cocode); $i++) {
            $sdir = array();

```

```

        $cdir[] = $sdir;
        if ($this->mapping && !empty($charmap) && $this->
>ccode[$i] != "") {
            foreach ($charmap as $key=>$val) {
                if (is_dir($this->sounddir . '/' . $key) &&
stristr($val, $this->ccode[$i])) {
                    $cdir[] = $this->sounddir . '/' . $key;
                }
            }
        }
        shuffle($cdir);
        if (is_file($cdir[0] . '/' . strtolower($this->ccode[$i]) . '.mp3')) {
            $mp3s[$i] = $cdir[0] . '/' . strtolower($this->ccode[$i]) .
'.mp3';
        }
    }

    // Captha length == Number of soundfiles
    if (strlen($this->ccode) == count($mp3s)) {
        foreach ($mp3s as $mp3) {
            // Actual mp3 join here
            $this->mp3add($mp3);
        }
        // added a 1.5 sec silent mp3 - to prevent abrupt sound ending (
new quicktime 7.x on IE 7 )
        if (is_file($this->sounddir . '/silent.mp3')) {
            $this->mp3add($this->sounddir . '/silent.mp3');
        }
    }
}

// Stream out
$this->mp3stream();

```

```

    }

    /**
     *
     * @ private function mp3add()
     * @ Open a mp3 - strip it tags and add it to string
     */
    private function mp3add($mp3 = "") {
        $mp3tmp = $this->mp3str;
        if ($mp3 != "" && is_file($mp3)) {
            $this->mp3str = file_get_contents($mp3);
            $this->mp3strip();
            $this->mp3str = $mp3tmp . $this->mp3str;
            return true;
        } else if ($mp3 == "") {
            return true;
        }
        return false;
    }

    /**
     *
     * @ private function mp3strip()
     * @ Strips begin and end tags from mp3 string
     * @ Set the mp3 header ( from first file )
     */
    private function mp3strip() {
        $i = 0;
        for ($i = 0; $i < strlen($this->mp3str); $i++) {

```

```

        if(ord(substr($this->mp3str, $i, 1)) == 255) {
            break;
        }
    }
    $mp3tmp = $this->mp3str;
    $this->mp3str = substr($this->mp3str, $i);
    if ($this->mp3tag == ") {
        $this->mp3tag = str_replace($this->mp3str, ", $mp3tmp);
    }
    if (strtolower(substr(substr($this->mp3str,(strlen($this->mp3str) - 128)), 0, 3))
== 'tag') {
        $this->mp3str = substr($this->mp3str, 0, (strlen($this->mp3str) -
129));
    }
}

```

```

/*****
*****
* @ private function mp3stream()
* @ Output the new mp3 file
*****
*****/

```

```

private function mp3stream() {
    $this->mp3str = $this->mp3tag . $this->mp3str;
    header('Expires: Mon, 1 Jan 2000 00:00:00 GMT');
    header('Cache-Control: no-cache, must-revalidate');
    header('Pragma: no-cache');
    header('Content-Transfer-Encoding: binary');
    header('Content-Disposition: inline; filename=captcha.mp3');
    header('Content-type: audio/mpeg');
    header('Cache-Control: post-check=0, pre-check=0');
}

```

```

header('Pragma: public');
header('Connection: close');
header('Content-Length: ' . strlen($this->mp3str));
echo $this->mp3str;
exit;
}

```

```

/*****
*****
* @ public function image
* @ output for the captcha image
*****
*****/

public function image() {
    $this->captchaCode();
    header('Expires: Mon, 1 Jan 2000 00:00:00 GMT');
    header('Last-Modified: ' . gmdate('D, d M Y H:i:s') . ' GMT');
    header('Cache-Control: no-store, no-cache, must-revalidate');
    header('Cache-Control: post-check=0, pre-check=0', false);
    header('Pragma: no-cache');
    if ($this->type == 'jpg') {
        header('Content-type: image/jpeg');
    } else if ($this->type == 'gif' && function_exists('imagecreatefromgif')) {
        header('Content-type: image/gif');
    } else {
        header('Content-type: image/png');
    }
    $this->width = (int) $this->width;
    $this->height = (int) $this->height;
    $image = imagecreatetruecolor($this->width, $this->height);

```

```

if ($this->backgrounddir != "" && $this->readbgs()) {
    shuffle($this->backgrounds);
    $bg = $this->backgrounds[0];
    $parts = pathinfo($bg);
    $ext = strtolower($parts['extension']);
    $bgimg = false;
    if ($ext == 'png') {
        $bgimg = imagecreatefrompng($bg);
    } else if ($ext == 'gif' && function_exists('imagecreatefromgif')) {
        $bgimg = imagecreatefromgif($bg);
    } else if ($ext == 'jpg' || $ext == 'jpeg') {
        $bgimg = imagecreatefromjpeg($bg);
    }
    list($bg_w, $bg_h) = getimagesize($bg);
    if ($bgimg) {
        imagecopyresampled($image, $bgimg, 0, 0, 0, 0, $this->width,
$this->height, $bg_w, $bg_h);
        imagedestroy($bgimg);
    }

} else if ($this->transparent != "") {
    $dec = $this->deccolors($this->transparent);
    $bgcolor = imagecolorallocate($image, $dec[0], $dec[1], $dec[2]);
    imagefill($image, 0, 0, $bgcolor);
    imagecolortransparent($image, $bgcolor);
}
if ($this->readfonts()) {
    $space = 0;
    if ($this->dnsbl()) {
        $this->code = $this->forbidden;
        $this->rotate = 0;
    }
}

```

```

        $this->fonts = array($this->fonts[0]);
        unset($_SESSION[$this->session]);
    }
    if ($this->fontsize * (strlen($this->code) + 2) > $this->width) {
        $size = round($this->width / (strlen($this->code) + 2));
        $xo = $size;
    } else {
        $size = $this->fontsize;
        $xo = round(($this->width - ($size * strlen($this->code))) / 2);
    }
    if ($xo > $size) {
        $space = ($xo * 1.5) / strlen($this->code);
        $xo -= round($space * (strlen($this->code) - 1) / 2);
    }
    $yo = round(($this->height - $size) / 2);
    for ($i = 0; $i < strlen($this->code); $i++) {
        shuffle($this->fonts);
        shuffle($this->colors);
        $xcor = $space * $i;
        $ycor = 0;
        $rotate = 0;
        if ($this->rotate != 0) {
            $rotate = rand(0, (int) $this->rotate);
            $rotate = (rand(1,2) == 2) ? $rotate * -1 : $rotate;
            $xcor = $size - (cos(deg2rad($rotate)) * $size) + $xcor;
            $ycor = (sin(deg2rad($rotate)) * $size) / 2;
        }
        if (!empty($this->shades)) {
            shuffle($this->shades);
            $this->shadesize = (int) $this->shadesize;
            $dec = $this->deccolors($this->shades[0]);

```



```

        $color = imagecolorallocate($image, $dec[0], $dec[1],
$dec[2]);

        imagettftext($image, $size, $rotate, ($size * $i + $xo +
$xcor), ($size + $yo + $ycor + $this->shadesize), $color, $this->font[0], $this-
>code[$i]);
    }
    $dec = $this->deccolors($this->color[0]);
    $color = imagecolorallocate($image, $dec[0], $dec[1],
$dec[2]);

    imagettftext($image, $size, $rotate, ($size * $i + $xo + $xcor),
($size + $yo + $ycor), $color, $this->font[0], $this->code[$i]);
    }
}
if ($this->type == 'jpg') {
    imagejpeg($image);
} else if ($this->type == 'gif' && function_exists('imagecreatefromgif')) {
    imagegif($image);
} else {
    imagepng($image);
}
imagedestroy($image);
exit;
}

```

```

/*****
*****
* @ public function post
* @ returns true/false - if a post is allowed - true
*****
*****/

public function post() {

```

```

        if ($_SERVER['REQUEST_METHOD'] == 'POST' &&
isset($_SESSION[$this->session])) {
            $cs = $_SESSION[$this->session];
            unset($_SESSION[$this->session]);
            if (!empty($this->dnsbl)) {
                if (!isset($_SESSION[$this->dnsblname]) || $_SESSION[$this->dnsblname] != 'oke') {
                    return false;
                }
            }
            if (isset($_POST[$this->formkey])) {
                if (!preg_match('/^[A-Z0-9]+$/iU', $_POST[$this->formkey]))
{
                    return false;
                }
                $cc = $_POST[$this->formkey];
                foreach ($this->replacer as $key=>$var) {
                    $cc = str_replace($key, $var, $cc);
                }
                if ($cs == strtolower($cc)) {
                    return true;
                }
            }
        }
        return false;
    }
}

```

```

/*****

```

```

*****

```

```

* @ private function captcha code

```

```

* @ sets a lowercase session and $this->code with the captcha code

```

```

*****
*****/
private function captchaCode() {
    $this->codeLength = (int) $this->codeLength;
    for ($i = 0; $i < $this->codeLength; $i++) {
        $this->code .= $this->chars[rand(0, strlen($this->chars) - 1)];
    }
    $_SESSION[$this->session] = strtolower($this->code);
}

/*****
*****/
* @private function readFonts
* @rebuilds array $this->fonts with font + pad ( from user input or auto from dir )
*****/
*****/
private function readFonts() {
    $ext = '.ttf';
    $dir = substr($this->fontDir, -1) != '/' ? $this->fontDir . '/' : $this->fontDir;
    $dir = $this->installDir . '/' . $dir;
    if (is_dir($dir) && !empty($this->fonts)) {
        for ($i = 0; $i < count($this->fonts); $i++) {
            if (!strstr($this->fonts[$i], $ext)) {
                $this->fonts[$i] .= $ext;
            }
            if (is_file($dir . $this->fonts[$i])) {
                $this->fonts[$i] = $dir . $this->fonts[$i];
            }
        }
    }
}

if (empty($this->fonts) && is_dir($dir)) {

```

```

        $this->fonts = glob($dir . '*' . $ext);
    }
    return empty($this->fonts) ? false : true;
}

private function readbgs() {
    $dir = substr($this->backgrounddir, -1) != '/' ? $this->backgrounddir . '/' :
$this->backgrounddir;
    $sdir = $this->instaldir . '/' . $dir;
    if (is_dir($sdir) && !empty($this->backgrounds)) {
        for ($i = 0; $i < count($this->backgrounds); $i++) {
            if (is_file($sdir . $this->backgrounds[$i])) {
                $this->backgrounds[$i] = $sdir . $this->backgrounds[$i];
            }
        }
    }
    if (empty($this->backgrounds) && is_dir($sdir)) {
        $this->backgrounds = glob($sdir . '{*.gif,*.jpg,*jpeg,*.png}' ,
GLOB_BRACE);
    }
    return empty($this->backgrounds) ? false : true;
}

```

```

/*****

```

```

*****

```

```

* @ private function deccolors

```

```

* @ convert a hex color string to rgb

```

```

* @ returns an array (r,g,b) colors

```

```

*****

```

```

*****/

```

```

private function deccolors($color) {

```

```

        $color = str_replace('#', '', $color);
        return array(hexdec(substr($color, 0, 2)), hexdec(substr($color, 2, 2)),
hexdec(substr($color, 4, 2)));
    }

/*****
*****
* @ public function dnsbl
* @ validate IP on DNSBL ( blacklists )
* @ sets $_SESSION[$this->dnsblname] - returns false on not listed
*****
*****/

    public function dnsbl() {
        if (isset($_SESSION[$this->dnsblname])) {
            return $_SESSION[$this->dnsblname] == 'oke' ? false : true;
        } else if (preg_match('/^([0-9]{1,3})\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3})/', $_SERVER['REMOTE_ADDR'])) {
            $reversip = implode('.', array_reverse(explode('.',
$_SERVER['REMOTE_ADDR'])));
            $win_os = strtolower(substr(PHP_OS, 0, 3)) == 'win' ? true : false;
            foreach ($this->dnsbl as $list) {
                if (function_exists('checkdnsrr')) {
                    if (checkdnsrr($reversip . '.' . $list . '.', 'A')) {
                        $_SESSION[$this->dnsblname] = $list;
                        return true;
                    }
                } else if ($win_os) {
                    $lookup = array();
                    @exec('nslookup -type=A ' . $reversip . '.' . $list . '.',
$lookup);
                    foreach ($lookup as $line) {

```



```

$langs = glob($this->instaldir . '/' . $this->sounddir . '/*');
$flags = array();
foreach($langs as $lang) {
    if (is_dir($lang)) {
        $flags[] = basename($lang);
    }
}
if (count($flags) <= 1) {
    $flags = array();
}
$imgurl = str_replace(str_replace('\\', '/',
$_SERVER['DOCUMENT_ROOT']), "", $this->instaldir);
$flagurl = $imgurl . '/images/flags/';
$imgurl = $imgurl . '/images/';
$tpl = str_replace('{IMGURL}', $imgurl, $tpl);
$flagout = "";
foreach($flags as $flag) {
    $flagout .= '<a href="' . $this->url . '?fclang=' . $flag . '"
onclick="return captchaLang(this);">';
    $flagout .= '</a>' . "\n";
}
$tpl = str_replace('{FLAGS}', $flagout, $tpl);
$tpl .= "\n<!-- End Mp3 Captcha Form (c) scripts.titude.nl 2008 --
>\n";
$html = $tpl;
}
return $html;
}

```

```

/*****
*****
* @ public function lanswitch
* @ Set language for sound
*****
*****/

public function langswitch() {
    if (isset($_GET['fclang']) && strlen($_GET['fclang']) == 2) {
        $_SESSION['fclang'] = $_GET['fclang'];
        if (isset($_GET['fchttp']) && $_GET['fchttp'] == 'xml') {
            echo "&nbsp;";
            exit;
        }
    } else if (isset($_SESSION['fclang']) && strlen($_SESSION['fclang']) == 2) {
        $this->language = $_SESSION['fclang'];
    }
}

}

?>

```

6. CAPTCHA based Email hacking prevention

Email security is one of the major areas of concern today. Our paper suggests, many hacking prevention methods along with the ways to retrieve a hacked account. It also presents an innovative CAPTCHA cum password technique that will save the crucial information from being disclosed to the hacker even if the account is hacked.

We have used two main algorithms:

6.1 Image Grid Algorithm

a) Set a static image grid

We use image mining algorithm to create a grid of images.

- b) Leave margins so as to wrap the images nicely
- c) Align the photographs
- d) Adding the slider

Combination of CAPTCHA and password is effective to thwart bot attacks and any human intrusion. Since image is used in the master password, it reduces the difficulty in remembering the password. Suggestions which were proposed, if implemented they will enhance the security of existing email service providers. Hence an efficient and easy way to recover a hacked email account is developed.

6.2 Image Mining Algorithm

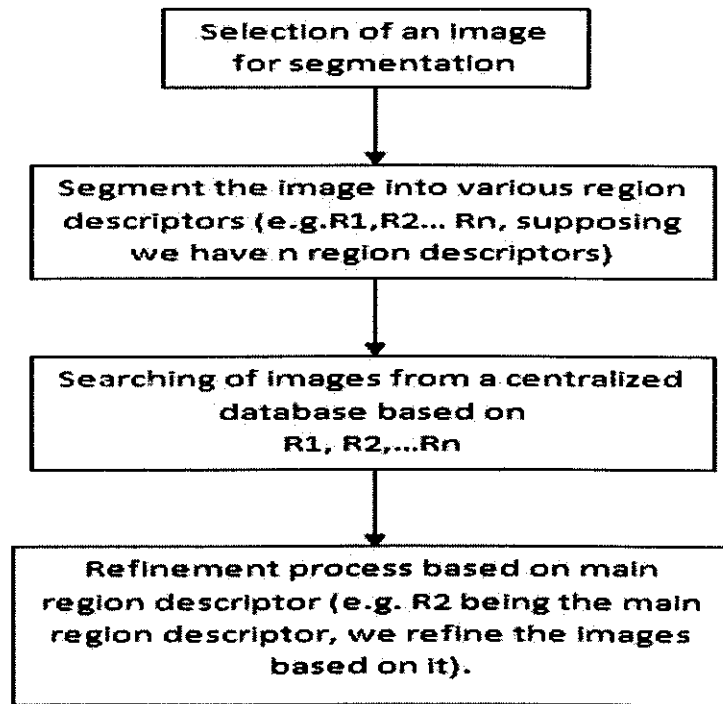


Figure 15: Image Mining Algorithm

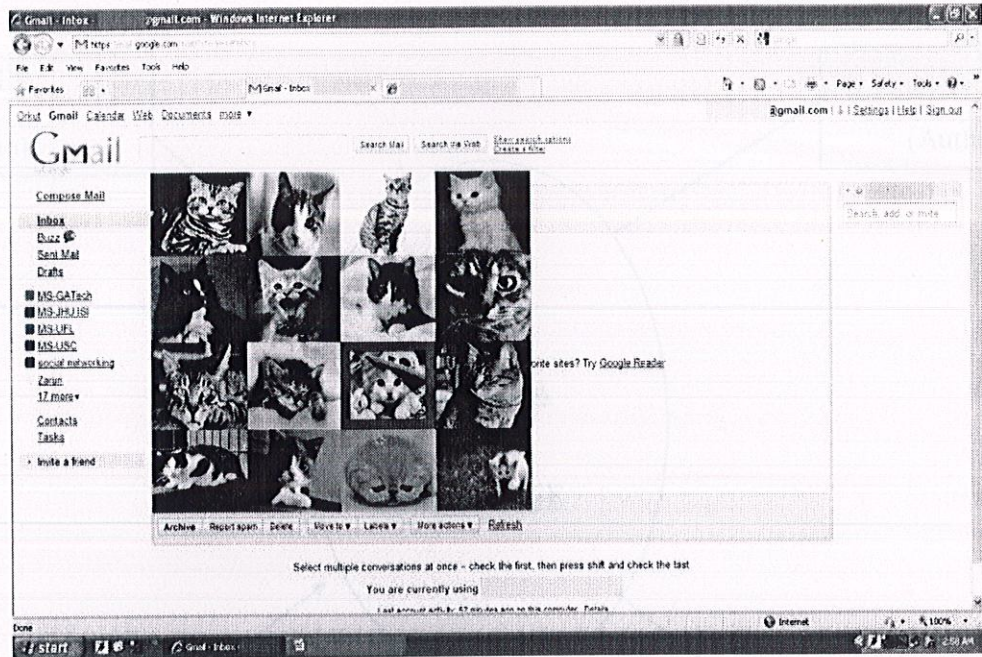


Figure 16: Email hacking prevention using Image Grid Algorithm

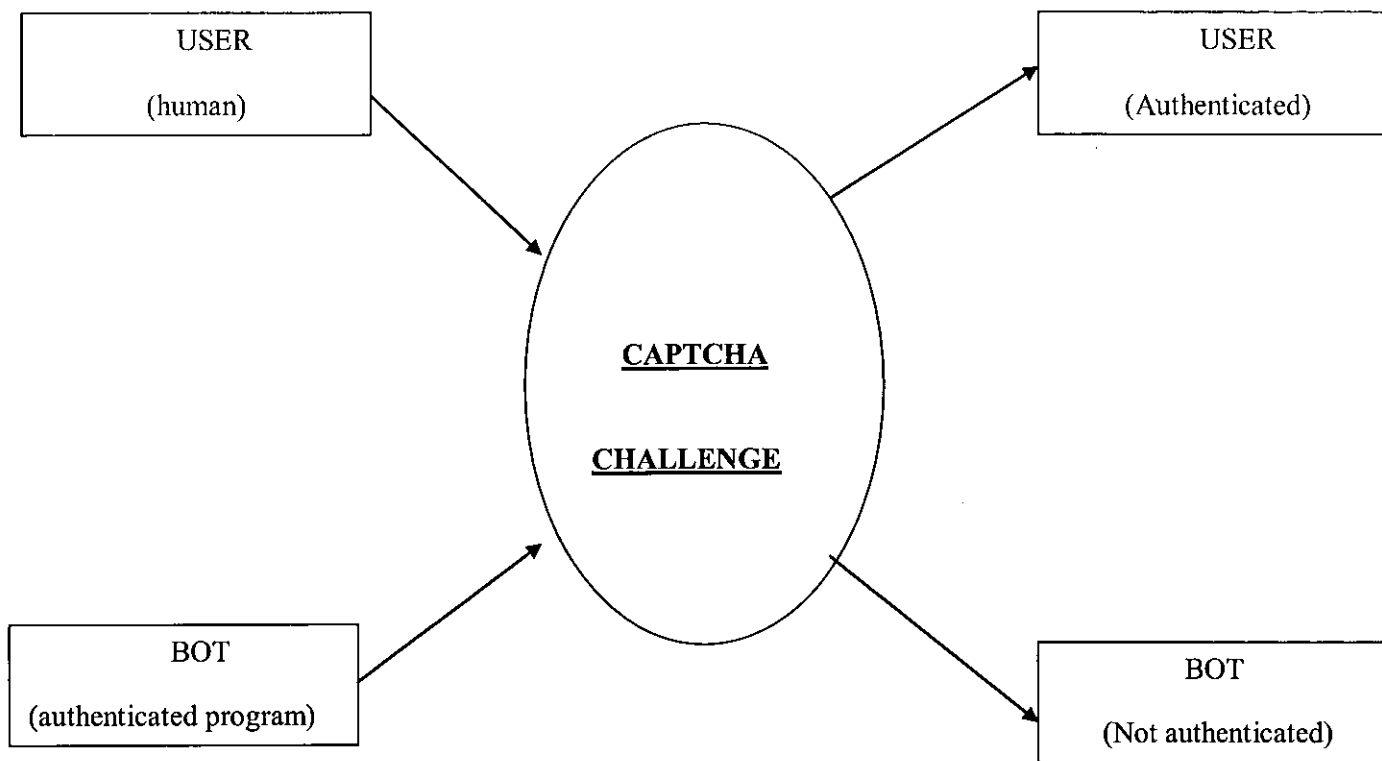


Figure 17: LEVEL 0 DFD

It represents the basic implementation of the CAPTCHA test. The challenge can be text based, image based or audio based.

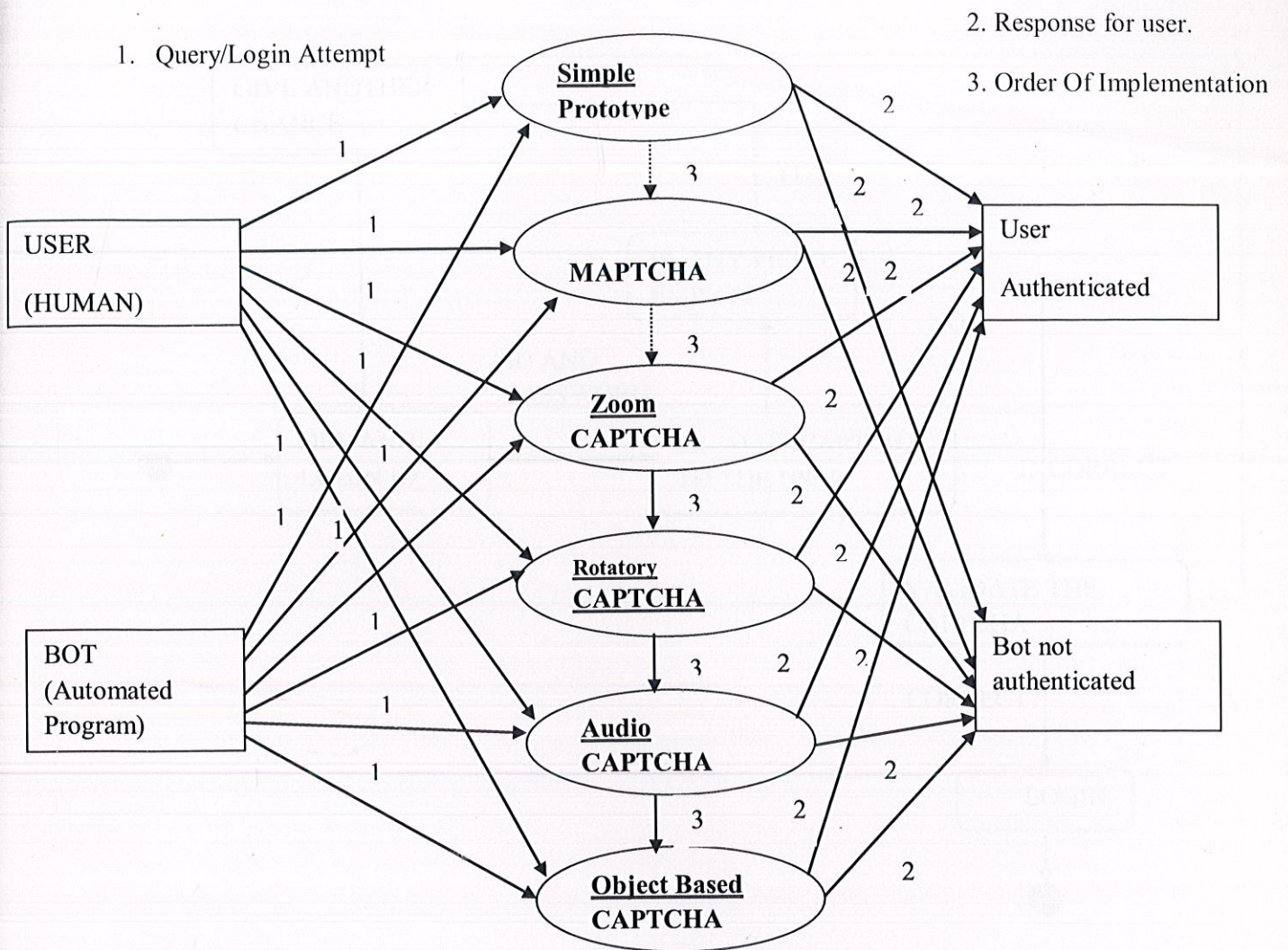


Figure 18: LEVEL 1 DFD

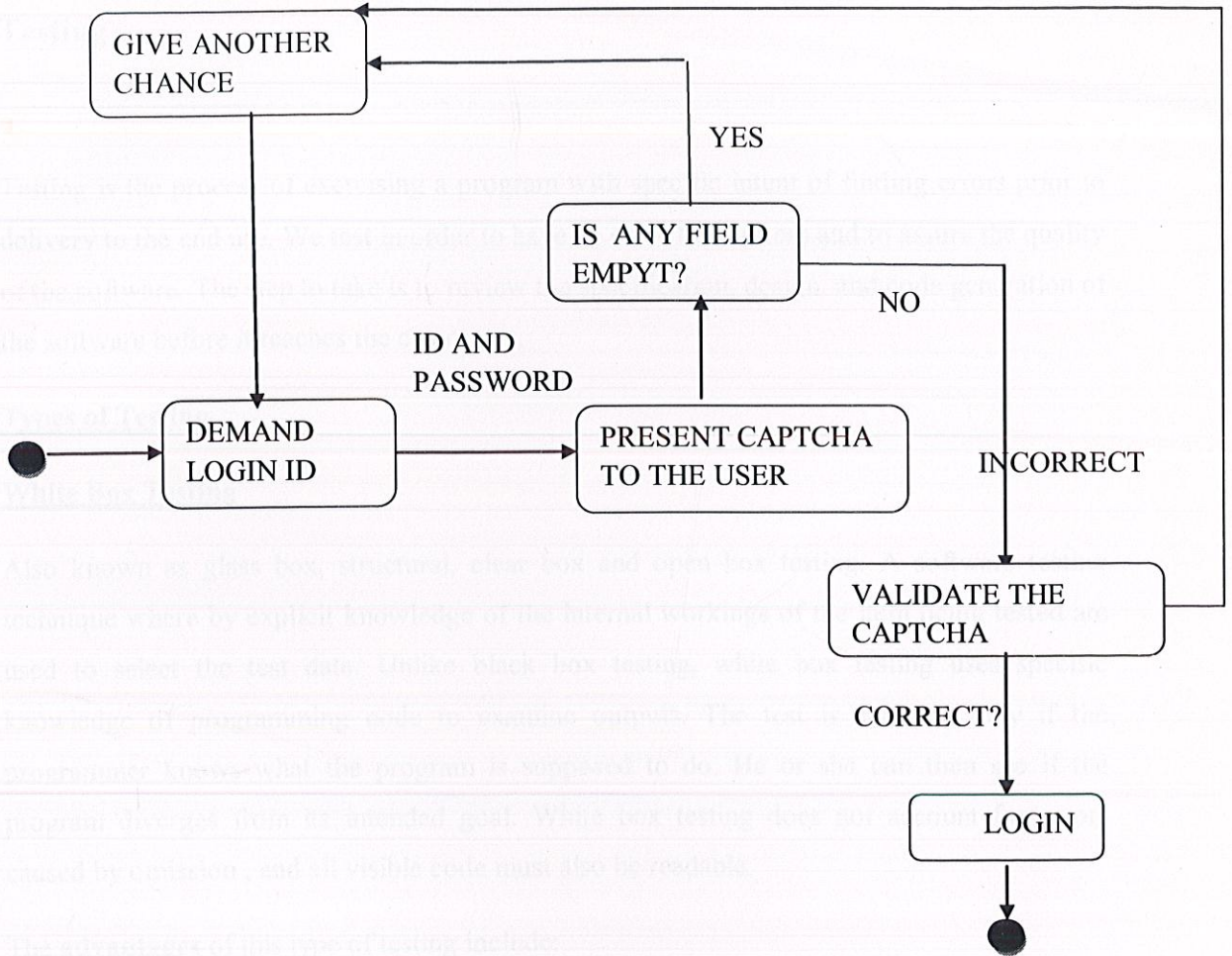


Figure 19: Activity Diagram (Login)

Chapter – V

Testing

Testing is the process of exercising a program with specific intent of finding errors prior to delivery to the end use. We test in order to have an error free system and to assure the quality of the software. The step to take is to review the specification, design, and code generation of the software before it reaches the customers.

Types of Testing

White Box Testing

Also known as glass box, structural, clear box and open box testing. A software testing technique where by explicit knowledge of the internal workings of the item being tested are used to select the test data. Unlike black box testing, white box testing uses specific knowledge of programming code to examine outputs. The test is accurate only if the programmer knows what the program is supposed to do. He or she can then see if the program diverges from its intended goal. White box testing does not account for errors caused by omission , and all visible code must also be readable.

The **advantages** of this type of testing include:

- As the knowledge of the internal coding structure is prerequisite, it becomes very easy to find out which type of input/data can help in testing the application effectively.
- The other advantage of white box testing is that it helps in optimizing the code.
- It helps in removing the extra lines of code, which can bring in the hidden defects.
- Forces test developer to reason carefully about the implementation.

The **disadvantages** of this type of testing include:

- As knowledge of code and internal structure is a prerequisite, a skilled tested is needed to carry out this type of testing, which increases the cost.

- And it is nearly impossible to look into every bit of code to find out hidden errors, which may create problems, resulting in failure of the application.
- Not looking at the code in a runtime environment. That's important for a number of reasons. Exploitation of vulnerability is dependent upon all aspects of the platform being targeted and source code is just of those components. The underlying operating system, the backend database being used, third party security tools, dependent libraries, etc, must all be taken into account. A source code review is not able to take these factors into account.
- Very few white-box tests can be done without modifying the program, changing values to force different execution paths, or to generate a full range of inputs to test a particular function.
- Miss cases omitted in the code.

Black Box Testing

Also known as functional testing. A software testing technique whereby the internal workings of the item being tested are not known by the tester. For example, in a black box test on software design the tester only knows the inputs and what the expected outcomes should be and not how the program arrives at those outputs. The tester does not examine the programming code and does not need any further knowledge of the program other than its specifications.

The **advantages** of this type of testing include:

- The test is unbiased because the designer and tester are independent of each other.
- The tester does not need knowledge of any specific programming language.
- The test is done from the point of view of the user, not the designer.
- Test cases can be designed as soon as the specifications are complete.

The **disadvantages** of this type of testing include:

- The test can be redundant if the software designer has already run a test case.
- The test cases are difficult to design.

- Testing every possible input stream is unrealistic because it would take an inordinate amount of time; therefore, many program paths will go untested.

For a complete software examination, both white box and black box tests are required.

Testing Levels

Testing can be done on the following levels:

- Unit testing tests the minimal software component, or module. Each unit (basic component) of the software is tested to verify that the detailed design for the unit has been correctly implemented. In an object-oriented environment, that is usually at the class level, and the minimal unit tests that include the constructors and destructors.
- Integration testing exposes defects in the interfaces and interaction between integrated components (modules). Progressively larger groups of tested software components corresponding to elements of the architectural design are integrated and tested until the software works as a system.
- System testing tests a completely integrated system to verify that it meets its requirements.
- System integration testing verifies that a system is integrated to any external or third party systems defined in the system requirements.

Regression Testing

Regression testing is a commonly used activity whose purpose is to determine whether the modifications made to a software system have introduced new faults. Regression testing is the process of testing changed to computer programs to make sure the older program still works with the new changes.

sTesting and debugging of the system is one of the main steps in the process of any project development.

For making the system user friendly and removing the chances of any errors/flaws, we tested the CAPTCHA's with various techniques.

- Testing while development :

- In the process of development of any type of CAPTCHA, the main emphasis was laid on the fact to make it secure and to safeguard it from attacks. We checked out CAPTCHA's for any such vulnerability and implemented various methods like MD5 functions to encrypt the data before passing it on to next page. Moreover, we also destroyed any piece of textual information after it was written to the image and CAPTCHA was made so that no information leak could occur.

- Testing after implementation :

- After making any kind of CAPTCHA, we tried to break it in any possible way, like the brute force attack where the bot tries to guess the text in the image but it always came out to be secure.
- Also, we put our CAPTCHA's to real time test and put them online for testing purposes.

- Testing in future:

- After implementing our main CAPTCHA server, we intend to provide feedback, suggestion and query forms to the actual user who will be the main user of our CAPTCHA's and his/her response and feedback is what mainly matters.

Chapter – VI

Conclusion

We started the project in our seventh semester – and the project has been interesting right from the beginning. The project requires basic information about the existing security techniques used on social networking/ e-mail websites on the world wide web. Also, along with the hardened information on the existing CAPTCHA's, we were required to look into possible methods to implement CAPTCHA's minus the flaws in the already deployed CAPTCHA's.

During our research on the existing CAPTCHA used on various websites – we stumbled upon the possibility of an image CAPTCHA used differently. To support the basis of our theory, we published a paper under the guidance of Professor Nitin Chanderwal.

We have managed to keep up with the goals we set for ourselves in the beginning, and we have successfully implemented a text CAPTCHA, an image CAPTCHA and an audio CAPTCHA. Also we have come up with the implementation of a more secure animated MAPTCHA.

Future of CAPTCHAs

We have now reached the state where computers are so good at letter recognition that any system that lets the majority of humans through is going to be susceptible to bots. More recently researcher's have concentrated their efforts on uping the difficulty of recognition by changing to photos instead of words.

So what can we expect from the CAPTCHAs in future? Is CAPTCHA even worth implementing still if it is so easily worked around? How can CAPTCHA fight back? Do u think it will be slowly replaced with other technologies and methods?

The future of CAPTCHA will need to be imperceptible to the user or innocuous enough not to impede the exchange of information.

For now, CAPTCHAs usually operate on visual and auditory senses. In the future when computers are undoubtedly more advanced, we can also operate on touch and taste. In the coming millennia we may even operate on spatial senses.

REFERENCES

The official CAPTCHA site

www.captcha.net

Last Batchelder. Honeypots: Better than CAPTCHAs?

www.people.w3.org/~cmsmcq/blog/?p=23

Connor, ML. 100 Line simple CAPTCHA JSP

www.jroller.com/mlconnor/

Repetition breaks Google Audio CAPTCHA

<http://www.peakpositions.com/seonews/repetition-breaks-google-audio-captcha.htm>

Jennifer Tam, Jiri Simsa (2009): Breaking Audio CAPTCHAs

http://www.captcha.net/Breaking_Audio_CAPTCHAs.pdf

G. Mori and J. Malik. "Recognizing Objects in Adversarial Clutter: Breaking a Visual CAPTCHA," In Computer Vision and Pattern Recognition CVPR'03, June 2003.

Wikipedia. CAPTCHA

<http://en.wikipedia.org/wiki/CAPTCHA>

Wikipedia, PHP

<http://en.wikipedia.org/wiki/PHP>

PUBLICATIONS

E- Mail Hacking Prevention Using CAPTCHAs Published at the SAM-2010 Conference in Las Vegas. Indexed in DBLP.