Objective Captcha

Darko Obradovic

Kaiserslautern.pm

http://kaiserslautern.pm.org
darko@kaiserslautern.pm.org

DFKI GmbH

http://www.dfki.uni-kl.de/~obradovic
darko.obradovic@dfki.de

with contributions from Florian Jostock, Kai Tombers and Fabian Zimmermann
Outline

- Introduction
- Text Captchas
- Objective Captcha
  - Idea
  - Usage
  - Architecture
  - Customisation
- Conclusion
Introduction

• CAPTCHA™ = Completely Automated Public Turing test to tell Computers and Humans Apart

• challenge-response system to protect web site access against bots
Introduction

- types of captchas:
  - visual, audio, semantic

- general requirements:
  - automated generation of new challenges
  - no „security by obscurity“!
  - challenge algorithms should be published
  - chance to solve a captcha by guessing should be less than 1%
  - delete incorrectly answered queries
  - prevent multiple guessing by same IP address
Text Captchas

- text captchas are most widely spread today
- common techniques:
  - dictionary words or arbitrary strings
  - text deformation, rotation, ...
  - background texturing
  - font variations
  - and many more...
- Perl support:
  - Authen::Captcha
  - GD::SecurityImage
Text Captcha Examples

5Z28AF
52ZL4P

ST JBD82X
83MNFXTS

5KS
NIB

5DH 2U2

McGuro0
uobBWp

FREE
GIRL

QHD8KC
KCD32T

Chocolat
Eurotrip
Text Captcha Riddles

dynamiT

originaTic N

kiTec
Text Captcha Problems

• mature OCR techniques:
  – very active research discipline
  – free availability of advanced OCR code (e.g. integrated OCR library in PHP)
  – specialised captcha crack tools with very high success rates exist

• with increased security, readability for humans becomes more and more difficult
The „Objective“ Idea

• use object images instead of characters!
• algorithmic creation on the fly
• variability of:
  – positioning
  – line length
  – proportions
  – optional elements
  – alternative elements
**Object Example 1: Glass**

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Glass 1" /></td>
<td><img src="image2" alt="Glass 2" /></td>
<td><img src="image3" alt="Glass 3" /></td>
<td><img src="image4" alt="Glass 4" /></td>
</tr>
<tr>
<td><img src="image5" alt="Glass 5" /></td>
<td><img src="image6" alt="Glass 6" /></td>
<td><img src="image7" alt="Glass 7" /></td>
<td><img src="image8" alt="Glass 8" /></td>
</tr>
<tr>
<td><img src="image9" alt="Glass 9" /></td>
<td><img src="image10" alt="Glass 10" /></td>
<td><img src="image11" alt="Glass 11" /></td>
<td><img src="image12" alt="Glass 12" /></td>
</tr>
</tbody>
</table>
Object Example 2: Bike
„Objective“ Tuning

• vulnerable to image recognition techniques:
  – count pixels, edges, enclosed areas, ...
  – compare feature vectors (SVM)

• use background patterns:
  – exploits human pattern recognition capability
  – hardly recognisable by computers
  – interferes with most features
How To Use It

- set up paths in oc_conf.pl once
- Captcha->form(3, 6, 1)
  - creates 3 objects with backgrounds
  - returns form elements in a table
  - offers 6 choices for each object
  - handles id and response automatically
  - just include this call in your CGI form!
- Captcha->check_response()
  - checks response in submitted query
CGI Form Example
Customisation

• add your own objects and backgrounds!
• draw objects/backgrounds with GD::line(), GD::rectangle(), GD::ellipse(), GD::arc(), ...
• implement position/length variance with – Captcha::fuzzy(fuzziness, x, y, ...)  
• implement options/alternatives with – Captcha::probe(probability)
sub draw {
    my ($self) = @_; 
    my $img = $self->{image}; 
    my $fg = $self->{fg}; 

    # define center point 
    my ($x, $y) = fuzzy(15, 100, 115); 

    # draw tree bole 
    my $bole_width = fuzzy(1, 3); 
    my $bole_height = fuzzy(10, 55); 
    $img->rectangle($x - $bole_width, $y, 
                    $x + $bole_width, $y + $bole_height, $fg); 

    # draw tree crown 
    my $crown_width = fuzzy(15, 50); 
    my $crown_height = fuzzy(10, 35); 
    $img->ellipse($x, $y - $crown_height, 
                  2 * $crown_width, 2 * $crown_height, $fg); 
}
Code Example: Tree
Further Possibilities

• harden against image recognition:
  – rotation, graphic filters, texturing, ...

• more objects, more backgrounds!
  classes to recognise rise multiplicatively:
  – 6 objects * 3 backgrounds = 18 classes
  – 12 objects * 6 backgrounds = 72 classes

• unique objects for your site protect you from generalised attacks
General Problems

• accessibility problem:
  – visual and audio captchas always exclude impaired users
  – only semantic captchas are fully accessible

• AI research will catch up

• social attacks always break captchas
  – delegate captchas to your own site's users

• waste of user time
  – 150,000 hours per day estimated
  – „reCAPTCHA“ project digitalises books
Related Links

- PWNtcha - captcha decoder
  - http://sam.zoy.org/pwntcha/
- Breaking a Visual CAPTCHA
- Inaccessibility of CAPTCHA
  - http://www.w3.org/TR/turingtest/
  - http://recaptchacha.net/
Questions?

Discussion!

Submit your plugins!