Who's watching your back?

Attacking CAPTCHAs for Fun and Profit

Gursev Singh Kalra
ISSW2012
Who Am I

Principal Consultant with Foundstone McAfee

Tools (TesserCap, SSLSmart, and many internal)

Security Research, Web Applications, Networks, Mobile Applications…. and more

Ruby, C#, Rails
Research Scope

Quantcast Top 1 Million
- 200+ CAPTCHA schemes analyzed
- Scores of Websites for Implementation

CAPTCHA Schemes
- Known OCR Engines for Classification
- Custom Image Preprocessing

CAPTCHA Implementations
- Register User Pages
- Recover Account/Password Pages
- Contact Us and Feedback Pages
CAPTCHAs: More Than Just the Image

1. Client GET /register.php
2. Server Create a SESSIONID for the current registration request
3. <html> ... <img src="/captcha.php"> ... </html> Return the CAPTCHA
4. GET /captcha.php + SESSIONID
5. Generate a random CAPTCHA and store in HTTP Session
6. Return the CAPTCHA
7. POST /verify.php + CAPTCHA Solution + Form Fields
8. Verify solution
9. SUCCESS, go to /success.php
   FAILURE, go to /register.php
From Here On…

Breaching the Client Side Trust

Server Side Attacks

Attacking CAPTCHA Schemes with TesserCap

Let’s Play Nice
Breaching the Client Side

Trust
Hidden Fields, Client Side Storage and More
Hidden Fields, Client Side Storage and More
Arithmetic CAPTCHAs

Please answer this simple math question.

\[ 8 + 2 = \_\_\_\_\_\_\_\_ \]  

Post comment

What Is \( 6 + 4? \) (required)

\[ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\]
Server Side Attacks
CAPTCHA Rainbow Tables
Implementation Flaws

CAPTCHAs are not generated at runtime

Limited number of CAPTCHAs

CAPTCHAs are assigned static index values to be referenced for verification and assignment

Observations

• One of the most popular implementation
• Seen On very high traffic websites
CAPTCHA Rainbow Tables
Attacking Static CAPTCHA Identifier

<table>
<thead>
<tr>
<th>Numeric Identifier</th>
<th>CAPTCHA</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>95C7A</td>
<td>95C7A</td>
</tr>
<tr>
<td>1</td>
<td>58413</td>
<td>58413</td>
</tr>
<tr>
<td>2</td>
<td>9D3BF</td>
<td>9D3BF</td>
</tr>
<tr>
<td>3</td>
<td>49F1C</td>
<td>49F1C</td>
</tr>
<tr>
<td>4</td>
<td>ABB87</td>
<td>ABB87</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>999999</td>
<td>D498A</td>
<td>D498A</td>
</tr>
</tbody>
</table>
## CAPTCHA Rainbow Tables
Attacking Static CAPTCHA Identifier

<table>
<thead>
<tr>
<th>Alphanumeric Identifier</th>
<th>CAPTCHA</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>uJSqsPvjxc6</td>
<td>95C7A</td>
<td>95C7A</td>
</tr>
<tr>
<td>9WzrowjPEql</td>
<td>58413</td>
<td>58413</td>
</tr>
<tr>
<td>nm8SfvtEwpP</td>
<td>9D3BF</td>
<td>9D3BF</td>
</tr>
<tr>
<td>fespW5LVqNQ</td>
<td>49F1C</td>
<td>49F1C</td>
</tr>
<tr>
<td>dgLSB1CKJRJ</td>
<td>ABB87</td>
<td>ABB87</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>QmJF3TQazcH</td>
<td>D498A</td>
<td>D498A</td>
</tr>
<tr>
<td>CAPTCHA MD5</td>
<td>CAPTCHA</td>
<td>Solution</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------</td>
<td>----------</td>
</tr>
<tr>
<td>68ecb8867cd7457421c2eca3227bffbd</td>
<td>95C7A</td>
<td>95C7A</td>
</tr>
<tr>
<td>84a78d24bc9637fcfb152f723b6e8e27</td>
<td>58413</td>
<td>58413</td>
</tr>
<tr>
<td>84125db583d64c346d97a74fa9e53848</td>
<td>9D3BF</td>
<td>9D3BF</td>
</tr>
<tr>
<td>C6a1ed9477846568cdea62c97e389811</td>
<td>49F1C</td>
<td>49F1C</td>
</tr>
<tr>
<td>E9fa81f69debe45bde97bba4743a8a23</td>
<td>ABB87</td>
<td>ABB87</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>B9df819f6174d6577661e12859226366</td>
<td>D498A</td>
<td>D498A</td>
</tr>
</tbody>
</table>
Write your custom solvers!
Chosen CAPTCHA Identifier Attack

Client 1: GET /captcha.php + SESSIONID

Server 2: Pick a random CAPTCHA Identifier from finite set of CAPTCHA values

3: <html> <img (CAPTCHA) + Identifier>

4: POST /verify.php + SESSIONID + Solution + Identifier

5: Use the Identifier to retrieve CAPTCHA solution + Verify solution

6: SUCCESS, go to next STEP

Failure, go to STEP 1
**CAPTCHA Fixation Attack**

**Client**

1. GET /captcha.php + SESSIONID

2. Pick a random CAPTCHA ID from finite set of CAPTCHA values

3. HTTP/1.1 302 Moved Temporarily
   Location: /get_captcha.php?id=captchaID

4. GET /get_captcha.php?id=captchaID + SESSIONID

**Server**

5. Set CAPTCHA ID or solution in HTTP Session

6. CAPTCHA

< CAPTCHA Verification >
**CAPTCHA Fixation Attack**

**Client** 1. GET /captcha.php + SESSIONID

**Server**

2. Pick a random CAPTCHA ID from finite set of CAPTCHA values

3. HTTP/1.1 302 Moved Temporarily
   Location: /get_captcha.php?id=captchaID

4. GET /get_captcha.php?id=evil_ID+ SESSIONID

5. Set CAPTCHA ID and/or solution in HTTP Session

6. CAPTCHA

< CAPTCHA Verification >
Persistent CAPTCHAs

Same CAPTCHA was returned for any number of registration attempts

CAPTCHAs can be brute-forced
**CAPTCHA Re-Riding Attack**

1. Client GET /captcha.php + SESSIONID
2. Server Create a random CAPTCHA.
3. Set CAPTCHA solution in HTTP Session
4. CAPTCHA
5. Client POST /verify.php + SESSIONID + Solution
6. Verify the CAPTCHA
7. Clear CAPTCHA state or SESSION
8. If SUCCESS, go to next STEP;
   If FAILURE, go to STEP 1;
GET /captcha.php

Set CAPTCHA solution in HTTP Session

Create a random CAPTCHA.

Set CAPTCHA solution in HTTP Session

Verify the CAPTCHA

Clear CAPTCHA state or SESSION

In Session CAPTCHA Brute-Force

1. GET /captcha.php

2. Create a random CAPTCHA.

3. Set CAPTCHA solution in HTTP Session

4. CAPTCHA

5. POST /verify.php + SESSIONID + Solution

6. Verify the CAPTCHA

7. Clear CAPTCHA state or SESSION

8. CAPTCHA solution brute-force with large number of requests

SUCCESS, go to next STEP

FAILURE, go to STEP 1
OCR Assisted CAPTCHA Brute-Force

r6syg

OCR 1

rGsyg

OCR 2

r6sy9

r[G6]sy[g9]

r6syg
OCR Assisted CAPTCHA Brute-Force

Solve CAPTCHA with an OCR

Bruteforce characters over the sample space

Continue…. Or better refresh SessionID for a new CAPTCHA!?
Attacking CAPTCHAs with TesserCap
The Weapon – TesserCap
TesseractCap Introduction

Retrieve CAPTCHA

8 stage Image preprocessing

Preprocessed CAPTCHA

Tesseract-OCR Engine

Extracted Text

HMLR

www.foundstone.com © 2010, McAfee, Inc.
TesserCap Demonstrations
Spatial Filters

\[
\frac{1}{9} \times \begin{array}{ccc}
1 & 1 & 1 \\
1 & 1 & 1 \\
1 & 1 & 1 \\
\end{array}
\times
\begin{array}{ccc}
1 & 1 & 1 \\
1 & 1 & 1 \\
1 & 1 & 1 \\
\end{array}
\]

\[
\frac{1}{16} \times \begin{array}{ccc}
2 & 4 & 2 \\
1 & 2 & 1 \\
\end{array}
\times
\begin{array}{ccc}
1 & 2 & 1 \\
\end{array}
\]

This Image: Digital Image Processing, Second Edition By Gonzalez and Woods
Spatial Filters in Action

FIGURE 3.36 (a) Image from the Hubble Space Telescope. (b) Image processed by a $15 \times 15$ averaging mask. (c) Result of thresholding (b). (Original image courtesy of NASA.)
## TesserCap Results

<table>
<thead>
<tr>
<th>CAPTCHA Provider</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Captchas.net</td>
<td>40-50%</td>
</tr>
<tr>
<td>Open captcha.com</td>
<td>20-30%</td>
</tr>
<tr>
<td>Snap host.com</td>
<td>60+%</td>
</tr>
<tr>
<td>Captcha creator.com</td>
<td>10-20%</td>
</tr>
<tr>
<td><a href="http://www.phpcaptcha.org">www.phpcaptcha.org</a></td>
<td>10-20%</td>
</tr>
<tr>
<td>Webspamprotect.com</td>
<td>40+%</td>
</tr>
<tr>
<td>ReCaptcha</td>
<td>0%</td>
</tr>
</tbody>
</table>
TesseractCap Results

<table>
<thead>
<tr>
<th>Website</th>
<th>Accuracy</th>
<th>Quantcast Rank</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wikipedia</td>
<td>20-30%</td>
<td>7</td>
</tr>
<tr>
<td>Ebay</td>
<td>20-30%</td>
<td>11</td>
</tr>
<tr>
<td>Reddit.com</td>
<td>20-30%</td>
<td>68</td>
</tr>
<tr>
<td>CNBC</td>
<td>50+%</td>
<td>121</td>
</tr>
<tr>
<td>Foodnetwork.com</td>
<td>80-90%</td>
<td>160</td>
</tr>
<tr>
<td>Dailymail.co.uk</td>
<td>30+%</td>
<td>245</td>
</tr>
<tr>
<td>Megaupload.com</td>
<td>80+%</td>
<td>1000</td>
</tr>
<tr>
<td>Pastebin.com</td>
<td>70-80%</td>
<td>32,534</td>
</tr>
<tr>
<td>Cavenue.com</td>
<td>80+%</td>
<td>149,645</td>
</tr>
</tbody>
</table>
Let’s Play Nice
a.k.a. Conclusion
Create a new **SESSIONID

Create a new CAPTCHA with Random Text

Set CAPTCHA solution in HTTP Session

Verify the CAPTCHA

Clear CAPTCHA state or HTTP SESSION

SUCCESS, go to next STEP

FAILURE, Go to STEP 1

A Secure CAPTCHA Implementation
A Secure CAPTCHA Implementation

- No client “influence on” or “knowledge about” the CAPTCHA content
- Random with a large sample space
- High on complexity to perform image preprocessing, segmentation and classification
- The client should not have direct access to the CAPTCHA solution
- No CAPTCHA reuse
Thank You!

Gursev Singh Kalra (@igursev)

gursev.kalra@foundstone.com
http://gursevkalra.blogspot.com
http://blog.opensecurityresearch.com