

Who's watching your back?

Attacking CAPTCHAs for Fun and Profit

Gursev Singh Kalra
APPSEC DC | April 4, 2012



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

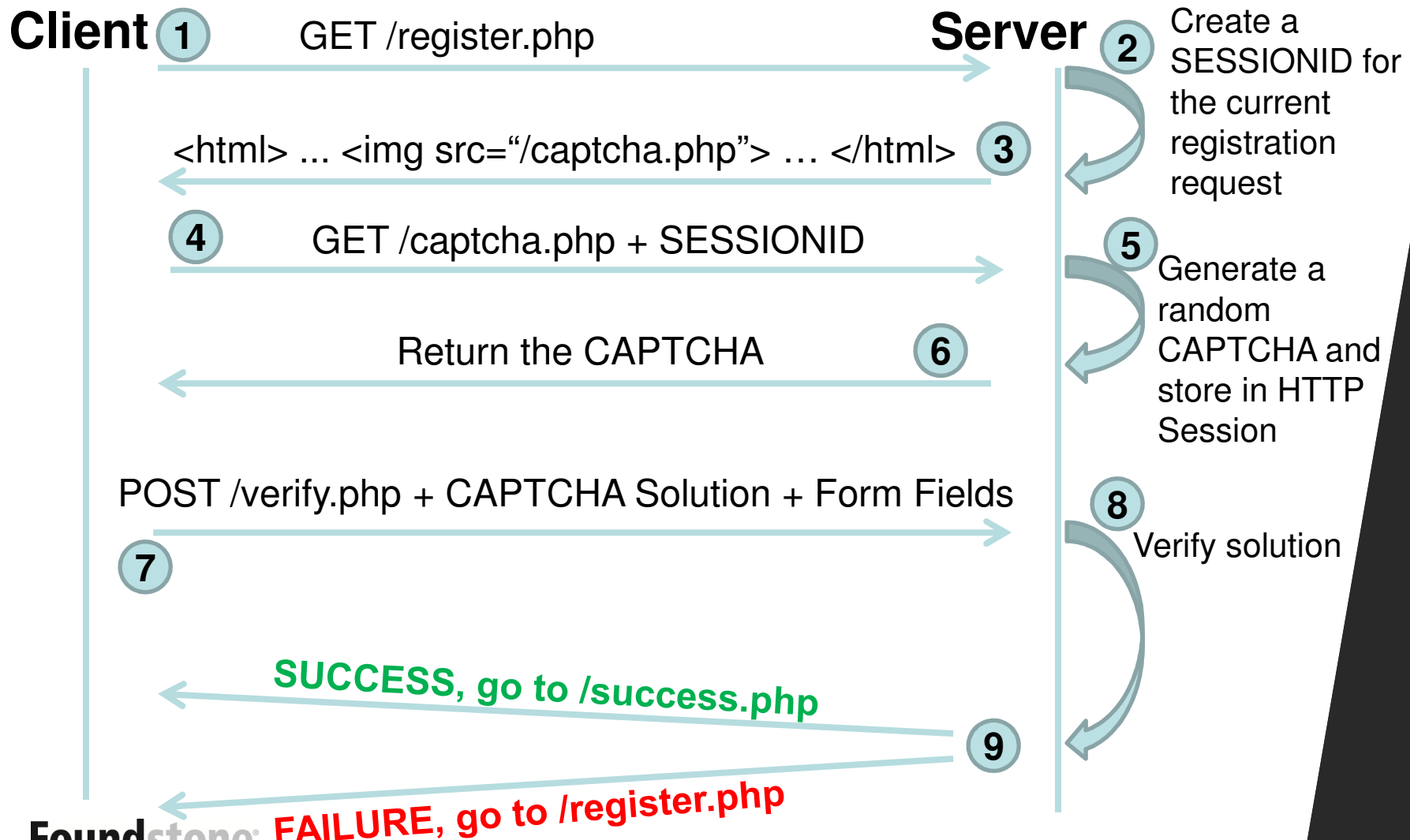
CAPTCHA Schemes

- 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





From Here On...



Breaching
the Client
Side Trust

Server Side
Attacks

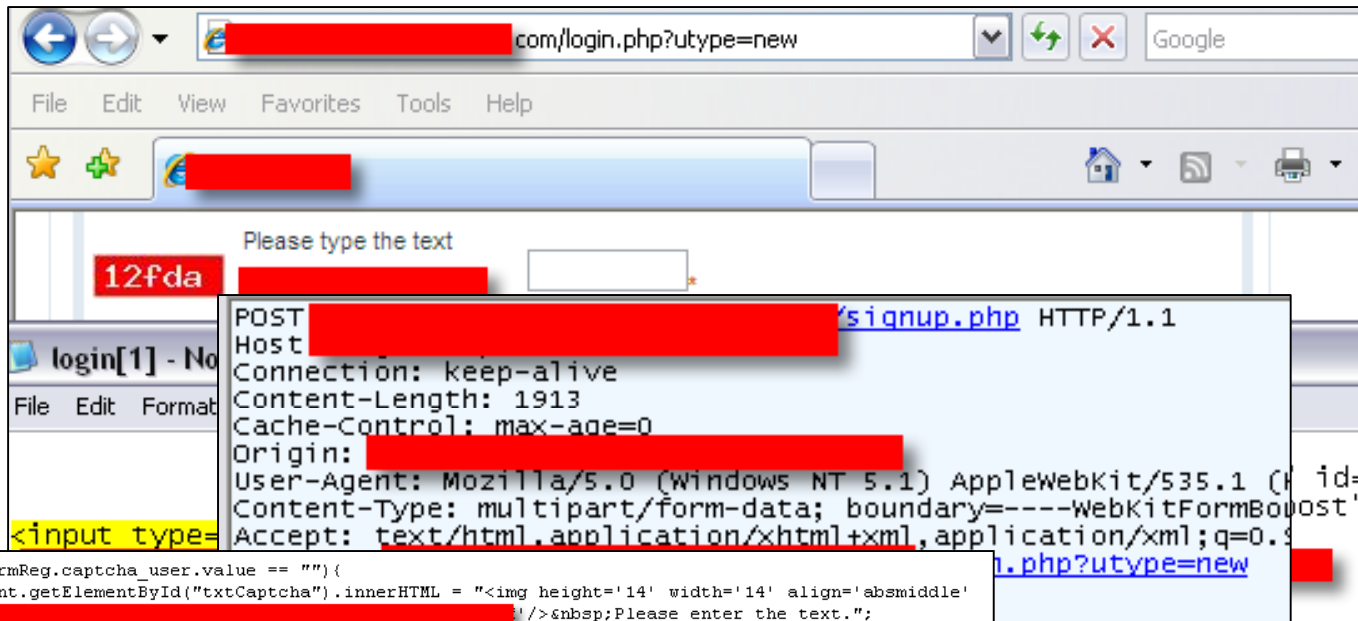
Attacking
CAPTCHA
Schemes
with
Tesseract

Let's Play
Nice



Breaching the Client Side Trust

Hidden Fields, Client Side Storage and More



The screenshot shows a web browser window with the address bar displaying a URL ending in `/login.php?utype=new`. The page content includes a text input field with the placeholder text "Please type the text" and a red box containing the text "12fda". A developer console window is open, showing a POST request to `signup.php` with the following headers:

```
POST /signup.php HTTP/1.1
Host: [redacted]
Connection: keep-alive
Content-Length: 1913
Cache-Control: max-age=0
Origin: [redacted]
User-Agent: Mozilla/5.0 (Windows NT 5.1) AppleWebKit/535.1 (KHTML, like Gecko) Chrome/18.0.938.63 Safari/535.1
Content-Type: multipart/form-data; boundary=----WebKitFormBoundary[redacted]
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
```

The console also shows the body of the request, which includes a hidden field `captcha_user` and a `captcha` field. The JavaScript code for the captcha is visible in the background:

```
if( document.frmReg.captcha_user.value == "" ){
    document.getElementById("txtCaptcha").innerHTML = "<img height='14' width='14' align='absmiddle' src='[redacted]' /><br>Please enter the text.";
    return false;
}
else {
    document.getElementById("txtCaptcha").innerHTML = "";
    if(document.frmReg.captcha_user.value == document.frmReg.captcha.value) {
        document.getElementById("txtCaptcha").innerHTML = "";
    }
    else {
        document.getElementById("txtCaptcha").innerHTML = "<img height='14' width='14' align='absmiddle' src='[redacted]' /><br>Please enter the correct vaue.";
        return false;
    }
}
```

Hidden Fields, Client Side Storage and More

The image displays a web application interface for a CAPTCHA challenge. The main window shows a form with a text input field and a "submit" button. The text input field contains the CAPTCHA text "Gx5TR". The form is labeled "CAPTCHA" in red text. Above the form, there is a "Confirm Password" field and a "Re-type" field.

The "Source" window shows the HTML source code of the page. The code includes a hidden field named "optcha" with a value of "709878". The code also shows the CAPTCHA image being generated by a PHP script. The code is as follows:

```
<td width="40%">To help prevent spam, please type the text you  
<span class="error">*</span>  
</td>  
<td width="5%">&nbsp;</td>  
<td width="40%" rowspan="2" style="text-align:left">  
<td>&nbsp;</td>  
</tr>  
<tr>  
<td><input name="number" type="text" class="form_textfield" size="15" maxlength="5" value="" />  
</td>  
</tr>
```

The command prompt window shows the output of a script, which is "Gx5TR". The command prompt is running the command "C:\WINDOWS\system32\cmd.exe - irb". The output is as follows:

```
^C  
ish(main):264:0 'R3g1VFI=' .d64  
=> "Gx5TR"  
ish(main):265:0>
```


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

Numeric Identifier	CAPTCHA	Solution
0	95C7A	95C7A
1	58413	58413
2	9D3BF	9D3BF
3	49F1C	49F1C
4	ABB87	ABB87
...		
99999	D498A	D498A

CAPTCHA Rainbow Tables

Attacking Static CAPTCHA Identifier

Alphanumeric Identifier	CAPTCHA	Solution
uJSqsPvjxc6	95C7A	95C7A
9WzrowjPEql	58413	58413
nm8SfvtEwpP	9D3BF	9D3BF
fespW5LVqNQ	49F1C	49F1C
dgLSB1CKJRJ	ABB87	ABB87
...		
QmJF3TQazch	D498A	D498A

CAPTCHA Rainbow Tables

Attacking Dynamic CAPTCHA Identifiers

CAPTCHA MD5	CAPTCHA	Solution
68ecb8867cd7457421c2eca3227bffbd	95C7A	95C7A
84a78d24bc9637fcfb152f723b6e8e27	58413	58413
84125db583d64c346d97a74fa9e53848	9D3BF	9D3BF
C6a1ed9477846568cdea62c97e389811	49F1C	49F1C
E9fa81f69debe45bded7bba4743a8a23	ABB87	ABB87
...		
B9df819f6174d6577661e12859226366	D498A	D498A

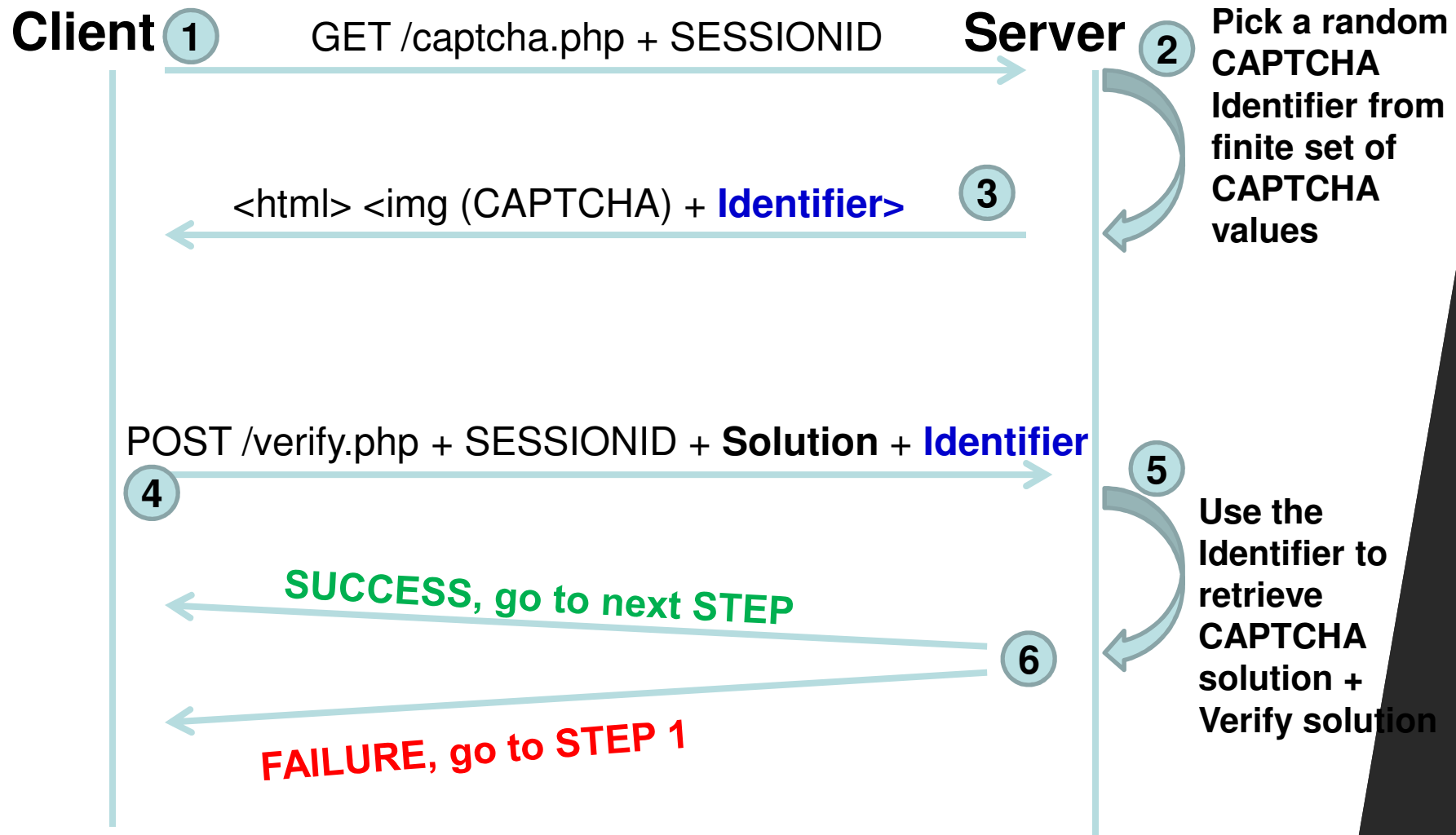


CAPTCHA Rainbow Tables

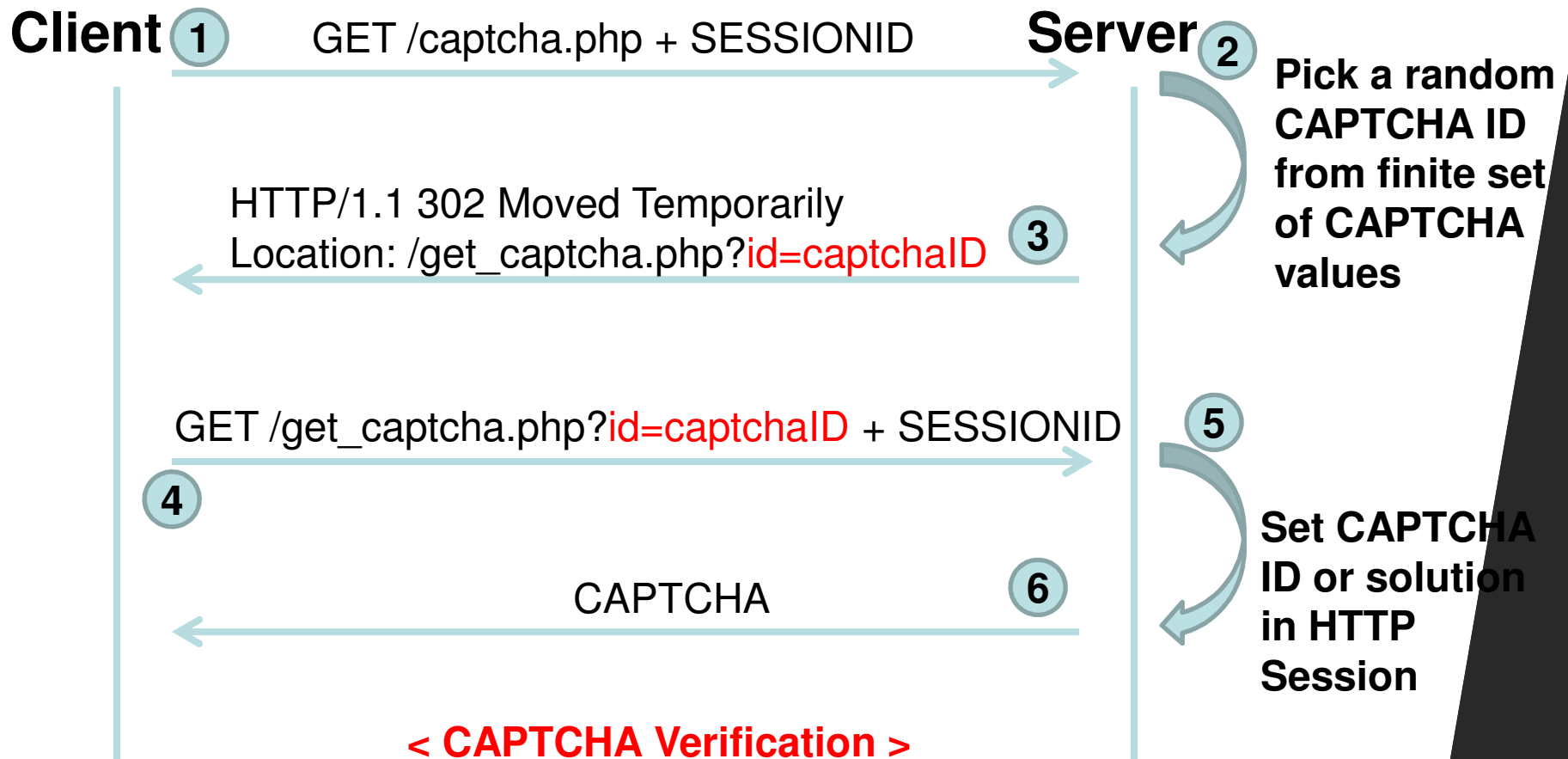
Dynamic Identifiers and Changing Images

Write your custom solvers!

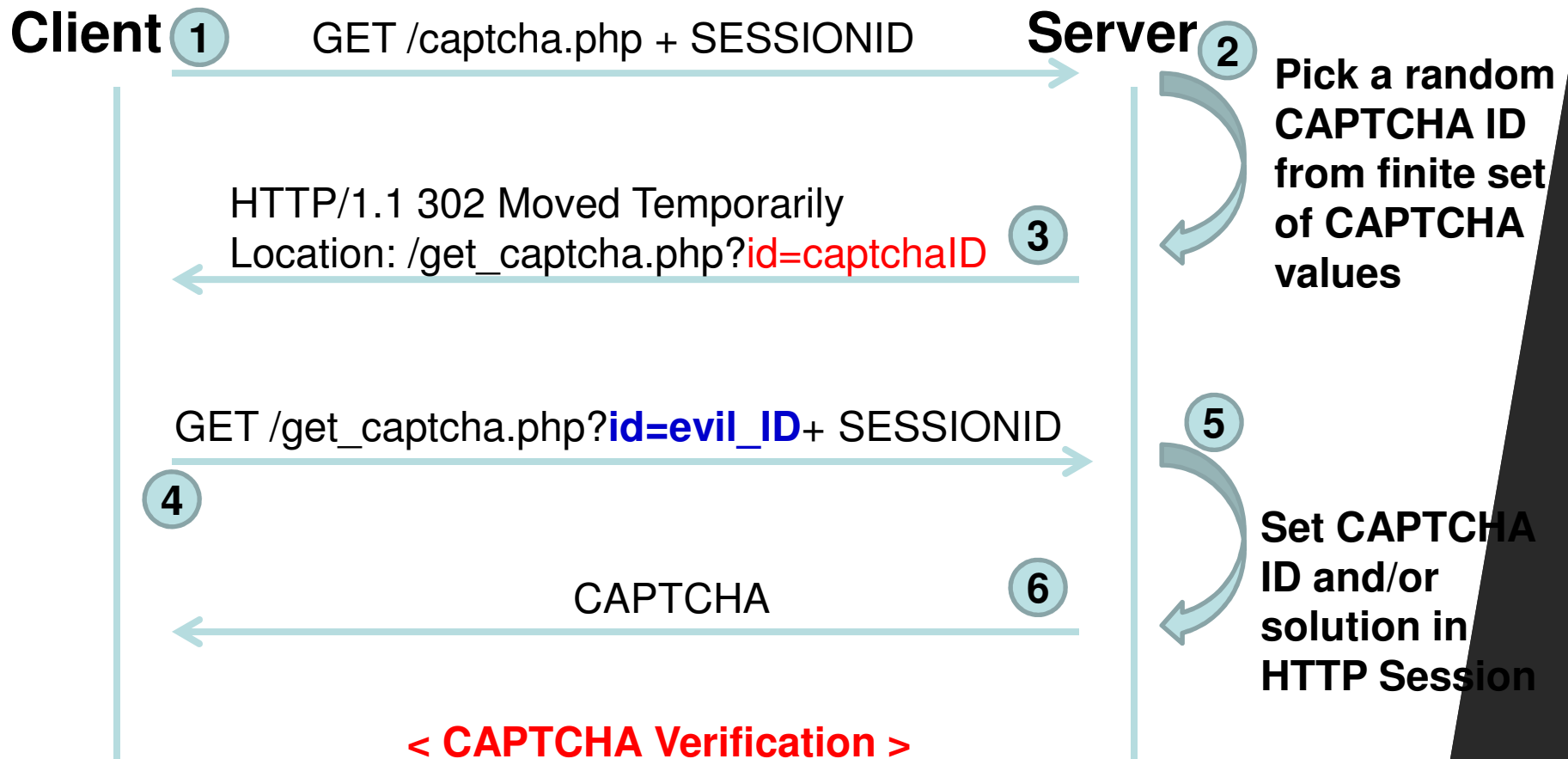
Chosen CAPTCHA Identifier Attack



CAPTCHA Fixation Attack



CAPTCHA Fixation Attack





Persistent CAPTCHAs

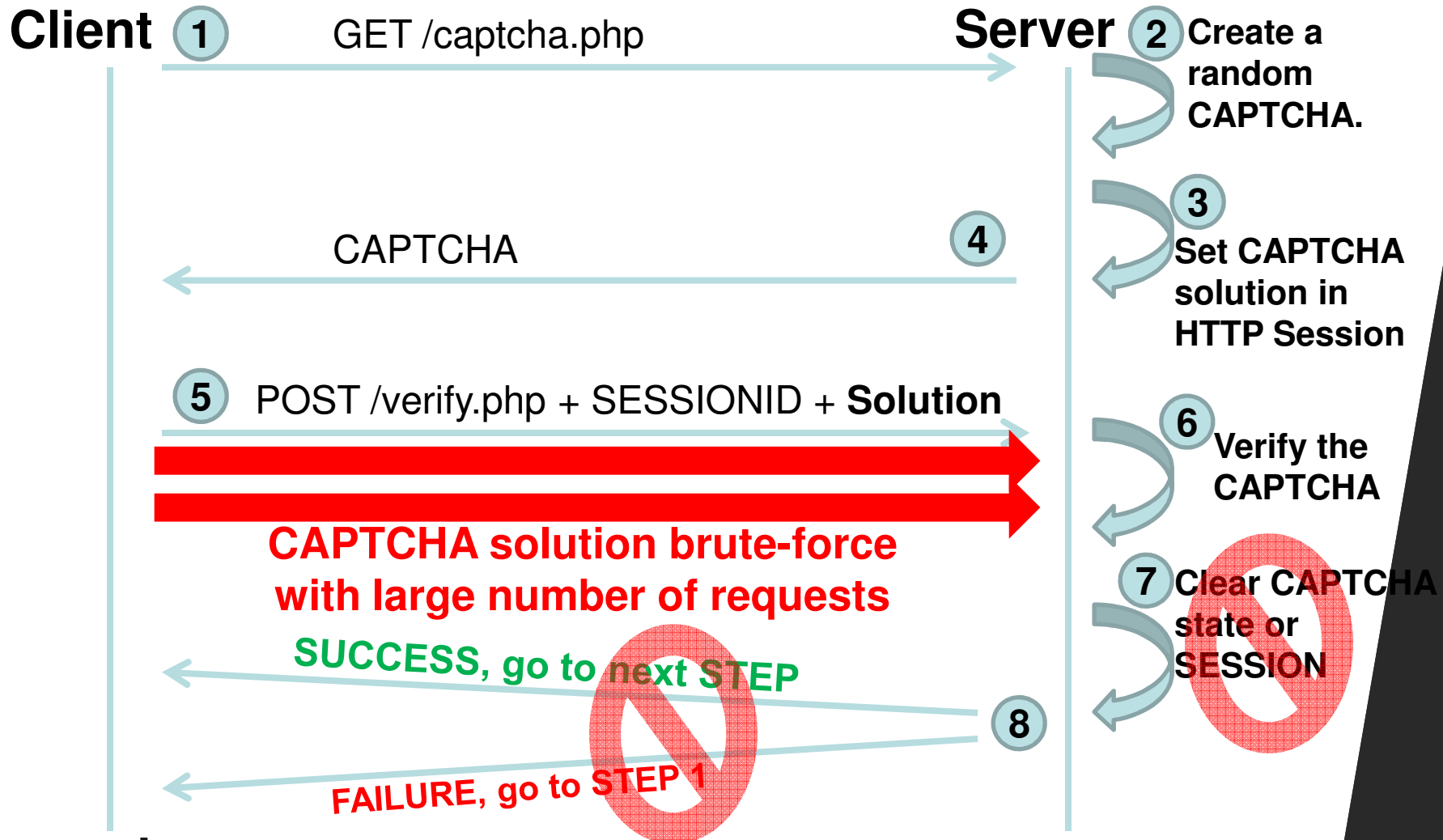
Same CAPTCHA was returned for any number of registration attempts

CAPTCHAs can be brute-forced

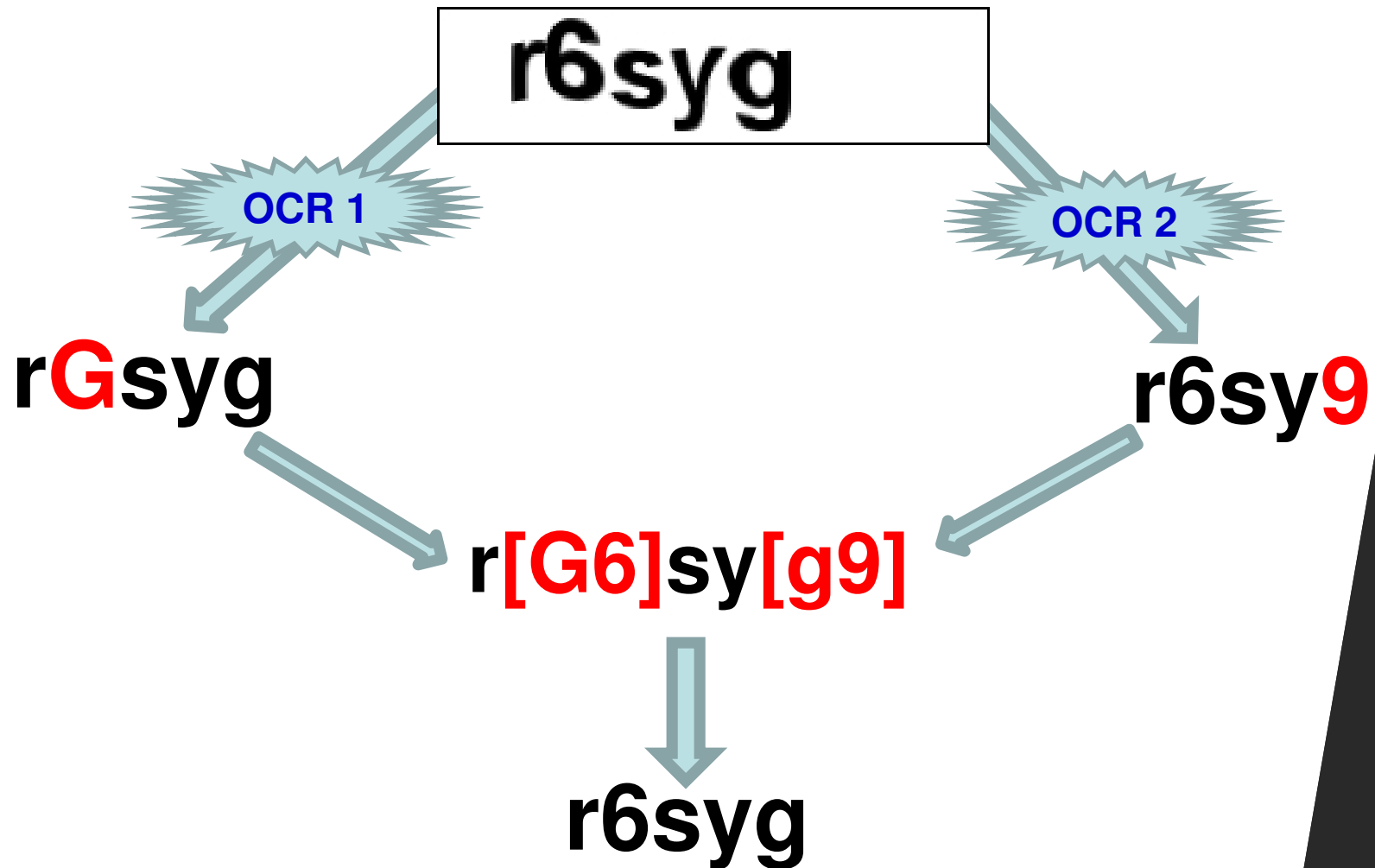
CAPTCHA Re-Riding Attack



In Session CAPTCHA Brute-Force



OCR Assisted CAPTCHA Brute-Force



OCR Assisted CAPTCHA Brute-Force

Solve CAPTCHA with an OCR

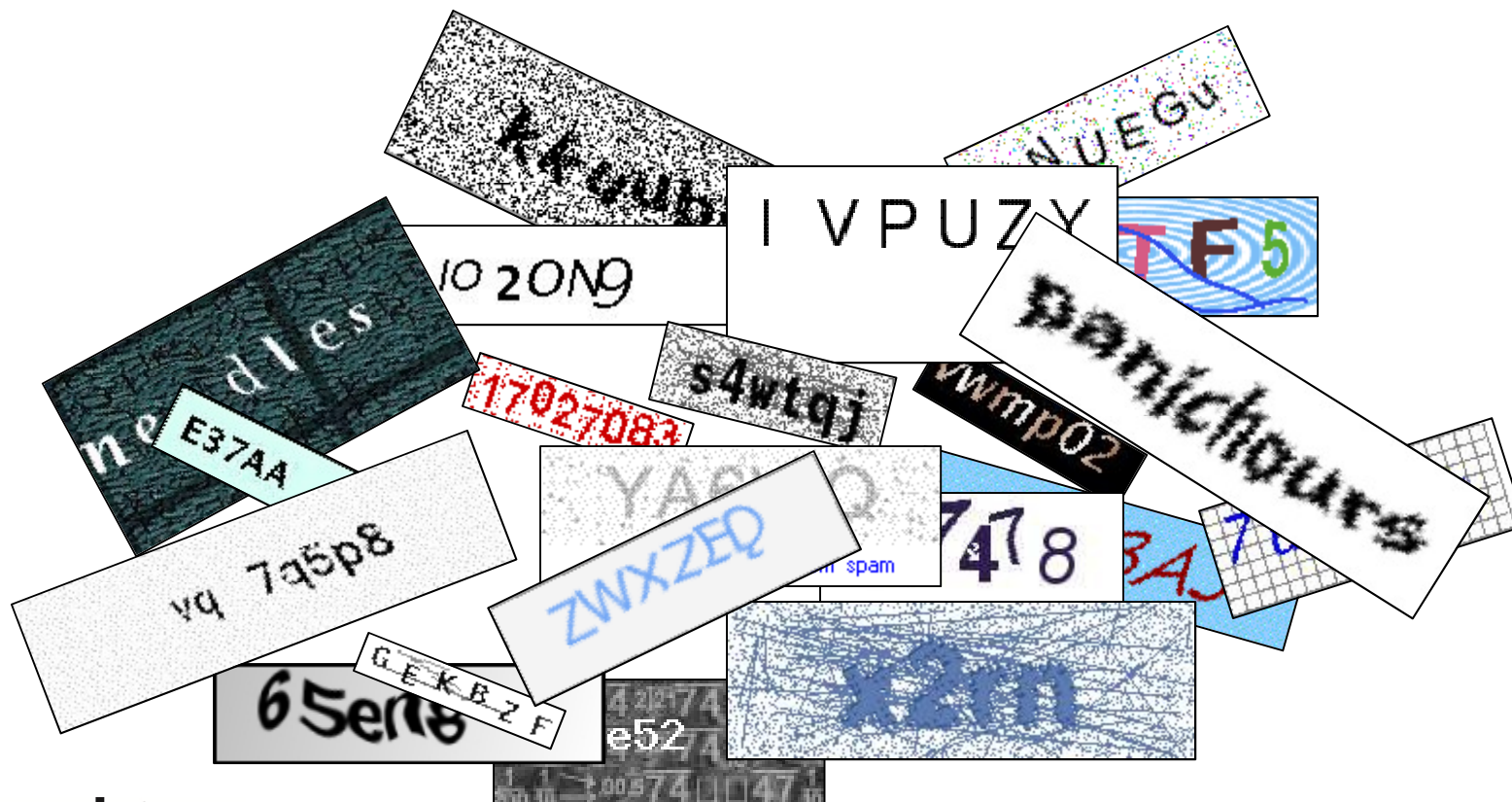
Bruteforce characters over the sample space

Continue.... Or better refresh SessionID for a new CAPTCHA!?

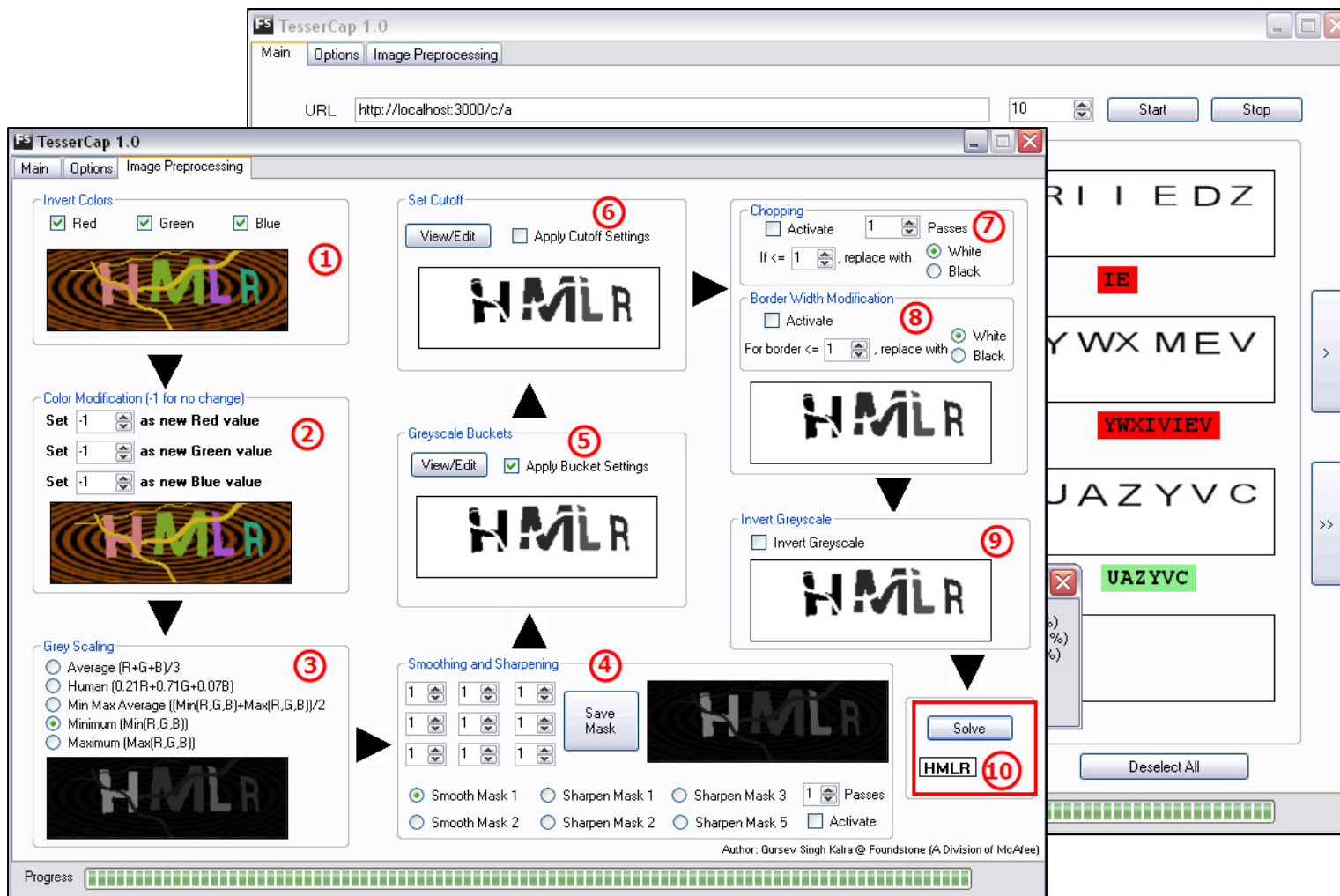


Attacking CAPTCHAs with Tesseract

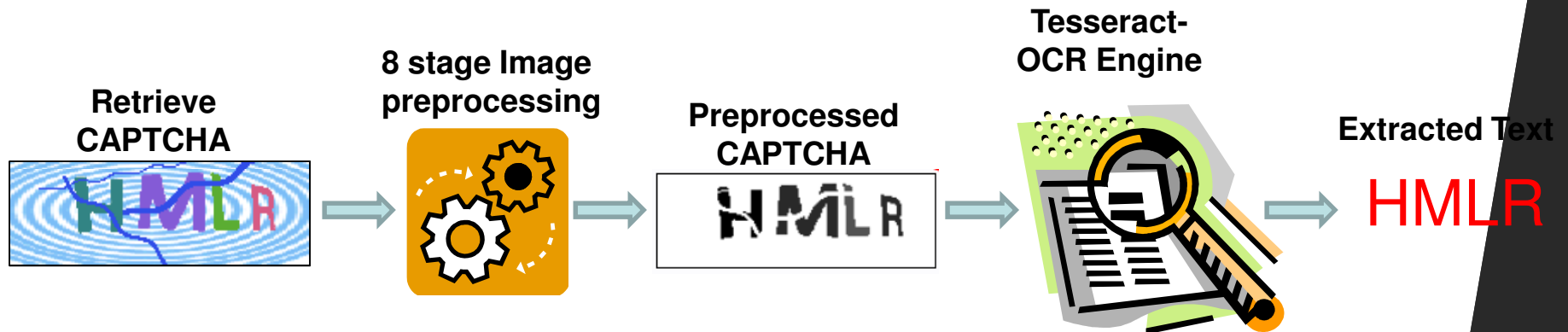
The Victims



The Weapon – TesseractCap



TesseractCap Introduction





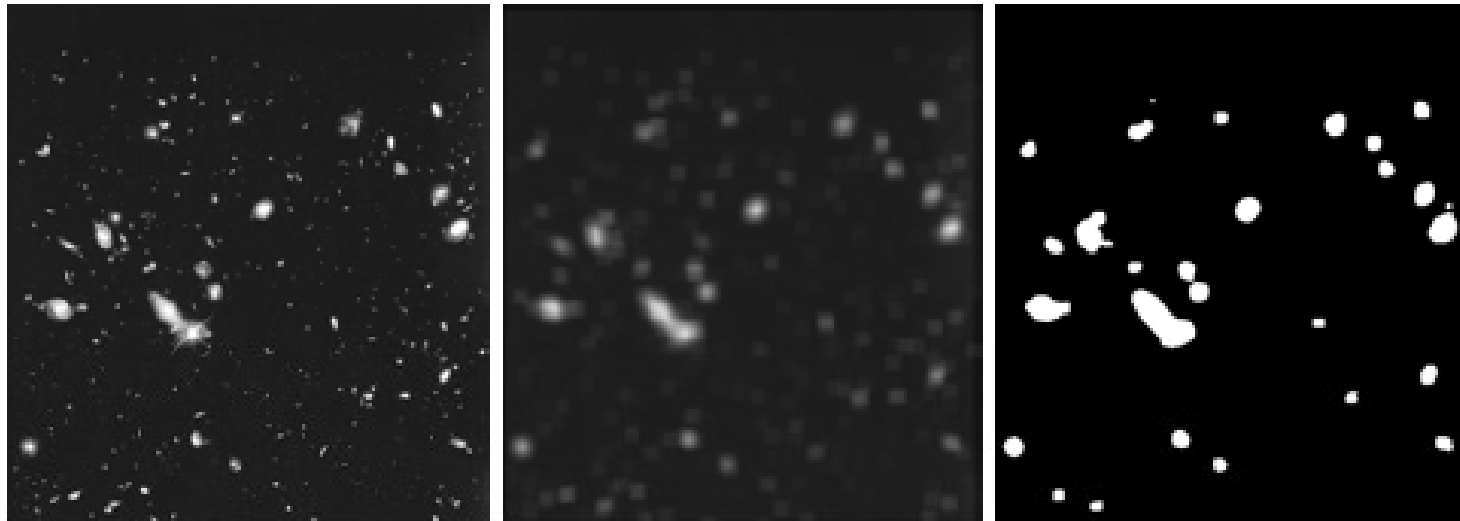
TesserCap Demonstrations

Spatial Filters

$$\frac{1}{9} \times \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array} \quad \frac{1}{16} \times \begin{array}{|c|c|c|} \hline 1 & 2 & 1 \\ \hline 2 & 4 & 2 \\ \hline 1 & 2 & 1 \\ \hline \end{array}$$

This Image: Digital Image Processing, Second Edition By Gonzalez and Woods

Spatial Filters in Action



a b c

FIGURE 3.36 (a) Image from the Hubble Space Telescope. (b) Image processed by a 15×15 averaging mask. (c) Result of thresholding (b). (Original image courtesy of NASA.)

This Image: Digital Image Processing, Second Edition By Gonzalez and Woods

TesserCap Results

CAPTCHA Provider	Accuracy
Captchas.net	40-50%
Opencaptcha.com	20-30%
Snaphost.com	60+%
Captchacreator.com	10-20%
www.phpcaptcha.org	10-20%
webspamprotect.com	40+%
ReCaptcha	0%

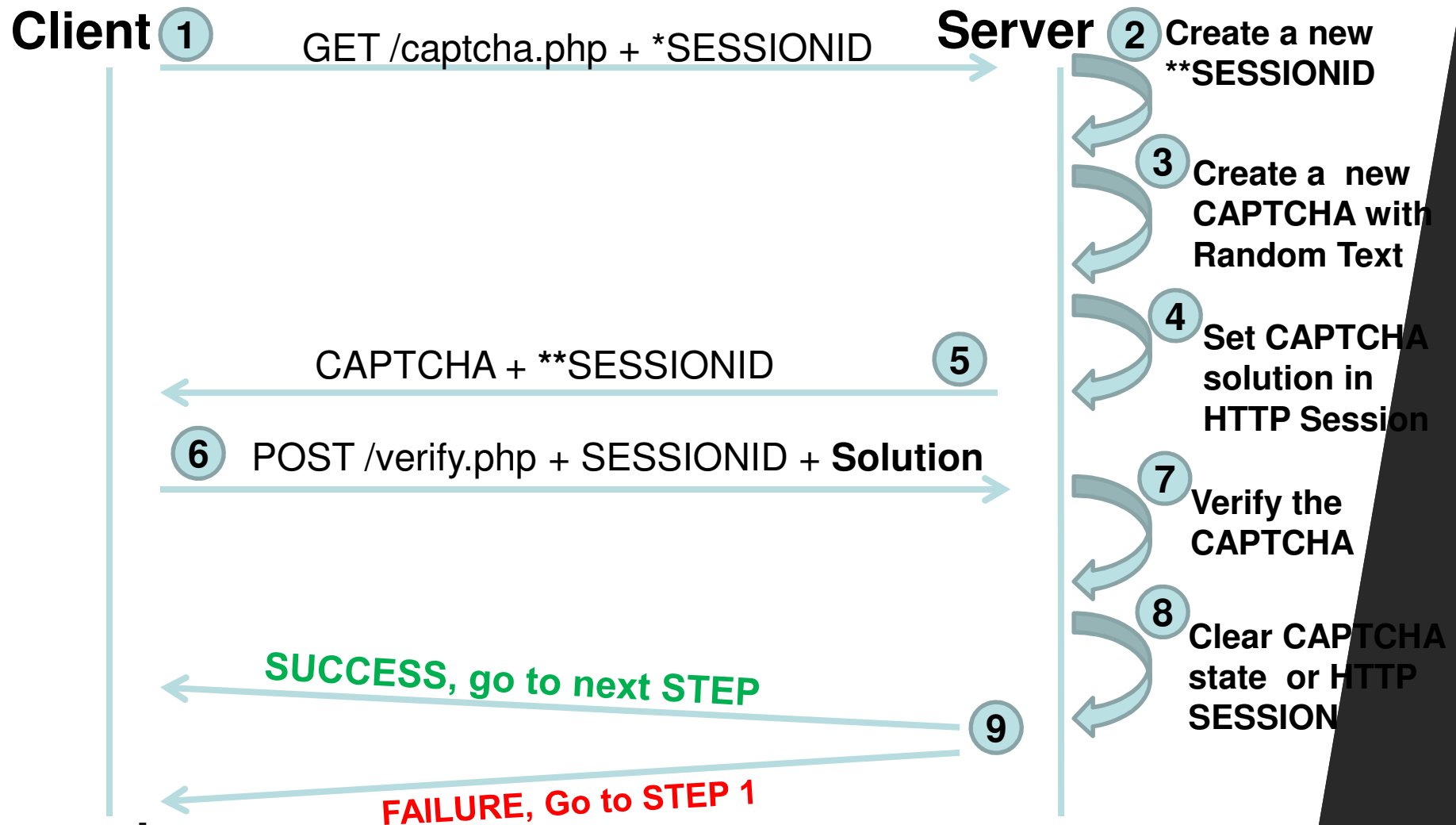
TesserCap Results

Website	Accuracy	Quantcast Rank
Wikipedia	20-30%	7
Ebay	20-30%	11
Reddit.com	20-30%	68
CNBC	50+%	121
Foodnetwork.com	80-90%	160
Dailymail.co.uk	30+%	245
Megaupload.com	80+%	1000
Pastebin.com	70-80%	32,534
Cavenue.com	80+%	149,645



Let's Play Nice a.k.a. Conclusion

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>