



I'm not a human: Breaking the Google reCAPTCHA

Suphannee Sivakorn, Jason Polakis, and Angelos D. Keromytis

[suphannee, polakis, angelos]@cs.columbia.edu
Columbia University, New York NY, USA

Since their inception, captchas have been widely used for preventing fraudsters from performing illicit actions. Nevertheless, economic incentives have resulted in an arms race, where fraudsters develop automated solvers and, in turn, captcha services tweak their design to break the solvers. Recent work, however, presented a generic attack that can be applied to any text-based captcha scheme. Fittingly, Google recently unveiled the latest version of reCaptcha. The goal of their new system is twofold; to minimize the effort for legitimate users, while requiring tasks that are more challenging to computers than text recognition. ReCaptcha is driven by an “advanced risk analysis system” that evaluates requests and selects the difficulty of the captcha that will be returned. Users may be required to click in a checkbox, or solve a challenge by identifying images with similar content.

In this paper, we conduct a comprehensive study of reCaptcha, and explore how the risk analysis process is influenced by each aspect of the request. Through extensive experimentation, we identify flaws that allow adversaries to effortlessly influence the risk analysis, bypass restrictions, and deploy large-scale attacks. Subsequently, we design a novel low-cost attack that leverages deep learning technologies for the semantic annotation of images. Our system is extremely effective, automatically solving 70.78% of the image reCaptcha challenges, while requiring only 19 seconds per challenge. We also apply our attack to the Facebook image captcha and achieve an accuracy of 83.5%. Based on our experimental findings, we propose a series of safeguards and modifications for impacting the scalability and accuracy of our attacks. Overall, while our study focuses on reCaptcha, our findings have wide implications; as the semantic information conveyed via images is increasingly within the realm of automated reasoning, the future of captchas relies on the exploration of novel directions.

1 Understanding Recaptcha

The reCaptcha service [1] offered by Google, is the most widely used captcha service, and has been adopted by many popular websites for preventing automated bots from conducting nefarious activities. Google announced [2] that deployment of a new reCaptcha mechanism designed to be more human-friendly and secure.

Widget

When visiting a webpage protected by reCaptcha, a widget is displayed (shown in Figure 1(a)). The widget's JavaScript code is obfuscated, to prevent analysis from third parties. When the widget loads, it collects information about the user's browser which will be sent back to the server. Furthermore, it performs a series of checks for verifying the user's browser.

Workflow

Once the user clicks in the checkbox, a request is sent to Google containing (i) the `Referrer`, (ii) the website's `sitekey` (obtained when registering for reCaptcha), (iii) the cookie for `google.com`, and (iv) the information generated by the widget's browser checks (encrypted). The request is then analyzed by the *advanced risk analysis* system, which decides what type of captcha challenge will be presented to the user.

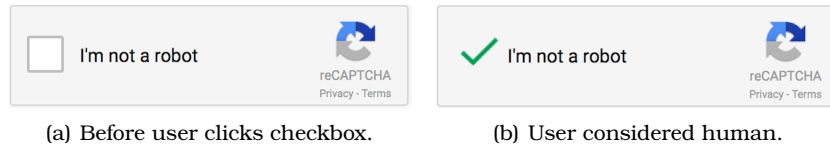


Figure 1: The reCAPTCHA widget.

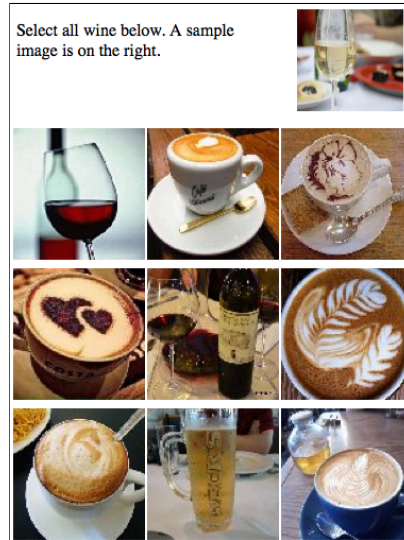


Figure 2: Similar images challenge by reCAPTCHA.

Challenge Type

The different type of challenges varies from user to user. Harder challenges will be presented if a specific user has low reputation or requests multiple challenges or provides several wrong answers many times. In our experiments we came across the following versions of reCAPTCHA:

- **“No captcha reCAPTCHA”** [Figure 1]. The new user-friendly version is designed to completely remove the difficulty of solving captchas. Upon clicking the checkbox in the widget, if the advanced risk analysis system consider the user have high reputation, the challenge will consider to be solved and no action required from the user. For the remainder of the paper, we will refer to this type of captcha as the *checkbox* captcha.
- **Image reCAPTCHA** [Figure 2]. This new version is built on the notion that identifying images with similar content. The challenge contains a sample image and 9 candidate images, and the user is requested to select those that are similar to the sample. The challenge usually contains a keyword describing the content of the images that the user is required to select. The number of correct images varies between 2 and 4.
- **Text reCAPTCHA** [Table 1 (a) to (e)]. These distorted texts are returned when the advanced risk analysis consider the user to have a lower reputation. (e) is fallback captcha which will be selected when the *User-Agent* fails certain browser checks, the widget automatically fetches and presents a challenge of this type, before the checkbox is clicked. Over the period of the following 6 months, text captchas appeared to be gradually “phased out”, with the image captcha now being the default type returned, as these captchas are harder for humans to solve despite being solvable by bots [3, 4].

Solution

Once the challenge has been presented to the user, it has to be answered within 55 seconds. Otherwise, the user is required to click on the checkbox again to receive a new challenge. Once the user clicks, an HTML field called

Table 1: Examples of remaining versions of reCaptcha.


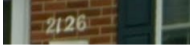



(a)		(b)	
(c)	special 	(d)	
(e)			

Table 2: Tracking cookie creation and “training” behavior.

Network	Web Surfing	Account	Threshold
Departmental	Frequent	No	9 th day
Departmental	Moderate	No	9 th day
ToR	Frequent	No	9 th day
ToR	Moderate	No	9 th day
Any	None	No	9th day

recaptcha-token is populated with a token. If the user is deemed legitimate and not required to solve a challenge, the token becomes valid on Google’s side. The token is submitted to the website when completing the desired action. The website sends a verification request through the reCaptcha API which contains: (i) a shared secret, (ii) the response token and, optionally, (iii) the user’s IP address. The response indicates if the verification was a success.

2 Analyzing Risk Analysis System

In this section we explore Google’s advanced advanced risk analysis and identify how various characteristics of user and user’s browser information affect it. We follow a black-box testing approach to identify how different aspects of our system and testing environment influence the risk analysis process. Our goal is to issue requests for captchas that will be considered legitimate by the advanced risk analysis system and, thus, receive checkbox captchas that can be solved with a single click.

2.1 Browsing History

Google tracking cookie plays crucial rule in determining the difficulty of challenge that is presented to the user. We constructed experiment and aim to quantify the minimum amount of browsing history required for a specific cookie that appeared to be from legitimate users to be presented a checkbox captcha. We explored multiple network connection setups, and activities to generate browsing history. For example we conduct the experiment in both our departmental network and Tor which US exit node. We automatically searches Google for certain terms, follows links from the results, watches video on Youtube, searches on Google Maps, visites popular websites that contains Google plus plugins and widgets. Surprisingly we are able to obtain a checkbox captcha after the beginning of the 9th day from the cookie’s creation, without requiring any browsing activities and type of network connection as shown in Table 2. Our experiment also revealed that each cookie can receive up to 8 checkbox captchas in a day.

2.2 Browser Environment

The reCaptcha widget executes a series of checks for detecting suspicious browser attributes or behavior. While the widgets JavaScript is obfuscated to prevent analysis, de-obfuscated code has been released¹ providing indications about the type of checks conducted. Here we explore how aspects of our automated browser environment affects the outcome of the risk analysis.

¹<https://github.com/neuroradiology/InsideReCaptcha>

Table 3: Combinations of mismatching information between what our system uses and what the User-Agent contains.

Component	9-day Cookie	System runs	User-Agent reports	Captcha
Browser	✓	Firefox/36.0	{Mobile/8C148 Safari/6533.18.5, Chrome/42.0.2311.135 Safari/537.36}	image
Browser version	✓	Firefox/36.0	Firefox/{10.0, 35.0, 36.0, 3.0.12}	checkbox
Browser version	✗	Firefox/36.0	Firefox/{10.0, 35.0, 36.0}	image
Browser version	-	Firefox/36.0	Firefox/1.0.4	fallback
Browser version	✗	Chrome/42.0	Chrome/{15.0.861.0, 4.0.212.1}	image
Browser version	-	Chrome/42.0	Chrome/3.0.191.3	fallback
Engine version	-	Chrome/42.0; AppleWebKit/537.36	AppleWebKit/{528.10, 530.5, 531.3}	fallback
Engine version	✗	Chrome 42/0; AppleWebKit/537.36	AppleWebKit/{532 and up}	image
Engine version	✓ ✗	Firefox/36.0; Gecko/20100101	Gecko/20040914	image
Browser/Engine	-	Chrome 42/0; AppleWebKit/537.36	Chrome 42/0; Gecko/20100101	fallback
Browser/Engine	✓ ✗	Firefox/36.0; Gecko/20100101	Firefox/36.0; AppleWebKit/537.36	image
Platform	✓	Linux x86_64	{(Macintosh; Intel Mac OS X 10.8); (Android; Mobile); (Windows NT 6.3);}	checkbox
-	-	-	wrong format or incomplete information	fallback
-	✓	Linux x86_64; Firefox/36.0	Mozilla/5.0 (iPhone; U; CPU like Mac OS X; en) AppleWebKit/420 (KHTML, like Gecko) Version/3.0 Mobile/1A543a Safari/419.3	checkbox

Canvas Rendering

Canvas rendering is a known technique to fingerprint user across machines and browsers [5]. The reCaptcha's JavaScript code creates a Canvas element and draws a predefined composition. After the rendering is complete, the element is encoded in base64 and sent back with the other data when the user clicks the checkbox. This piece of information can be used to browser rendering ability and determine the browser version and later compared to detect the discrepancies with the reported user-agent.

User-Agent

To identify how user-agent influence the user's reputation, We have grouped certain variations that exhibit the same behavior. We found that if the User-Agent contains an outdated version of browser or browser engine, the widget automatically considers the environment suspicious and presents the user with a fallback captcha (Table 1-(e)). The same happens if the browser and engine versions are up-to-date, but dont correspond to the actual environment of the experiment (e.g., if we use Firefox but report Chrome). Fallback captchas are also returned when the User-Agent is mis-formatted.

Screen Resolution and Mouse

We experimented with multiple combinations of screen resolutions, and various mouse behavior configurations (the timing of movements and movement patterns).None of these had a negative effect on the risk analysis.

2.3 Site Restriction

The attack's scale can be increased if we solve captchas on a website we control (attacker.com) but associate the tokens with a target website (example.com). This would facilitate captcha-solving services that harvest and sell tokens to others, as it will reduce the network activity and, thus, cost of the attacks. A straightforward clickjacking attack was demonstrated [6]. To prevent the attack, reCaptcha was re-designed so that the token is tied to the website where the challenge was presented. Apart from checking the Referrer, the widget identifies the website through the `document.location.hostname`, which is read-only and cannot be intercepted for security reasons. We present a workaround for bypassing this restriction. We setup a virtual host on our server and set the `ServerName` and other necessary fields to correspond with example.com. By using `a2ensite`, and modifying the hosts file, we can run our website on the localhost and trick reCaptcha into associating our request to example.com. To complete our attack we also need to send the targets sitekey, which can be trivially obtained by getting page source of the website.

Table 4: Combinations for passing the image reCaptcha.

Image Selection	Constraint	Pass
n Correct + k Wrong	$k \leq 1$	✓
$(n - 1)$ Correct	$n > 2$	✓
$(n - 1)$ Correct + k Wrong	$k > 0$	✗

3 Easy captcha breaker

The goal is to create cookies which are to appear as originating from legitimate users and not automated bots. In each case, we create a cookie in a clean virtual machine, where our browser automation system store non-account google.com cookies.

3.1 Token harvesting

We also explored if creating a large number of cookies from a single IP address is prohibited. We are able to create over 63,000 cookies in a single day without triggering any mechanisms or getting blocked, and are only limited by the physical capabilities of the machine. This indicates that there is no mechanism to prohibit the creation of cookies from a single IP address. The only restriction we detected was triggered by a massive number of concurrent requests (i.e., for detecting DoS attacks). The lack of a safeguard can be justified by the fact that creating cookies at a large scale has not been required by attacks before. Indeed, we present a novel *misuse of tracking cookies*, which makes them a valuable commodity for fraudsters.

3.2 Evaluation

Next, we deployed our system using these cookies after they had “aged”, to identify how many checkbox captchas we can solve in a single day. We experimented with different captcha-request rates and observed the dropping in more higher rates. In the end, despite being blocked for short periods of time, at the optimal rate we receive approximately 2,500 checkbox captchas per hour, which drops to about 1,200 during peak hours. During weekdays, our results vary between 52,000 and 55,000. We observe less blocking during the weekend, and obtain over 59,000 checkbox captchas per day.

4 Image Captcha

In this section we explore the new Google image captcha and present various characteristics that we observed throughout our testing. Our goal is find an some characteristics that attacker can exploit in solving image captchas.

4.1 Solution flexibility

We explore whether reCaptcha has any flexibility when deciding if the given solution is correct. In any case, at least two images have to be selected before the response is sent. We manually solved image challenges using different combinations of the number of correct (n) and wrong (k) selections. In most cases (74%) we found the number of correct candidate images to be 2; the rest contain 3 and we also found two challenges with 4. Table 4, our experiments reveal that a user can pass the challenge even if a correct image is missed or a wrong selection is provided. Based on these results, we set our captcha breaking system to select 3 images for the solution; this strategy offers us a “free” selection when $n = 2$ and may fall within the “relaxation” limits for the remaining challenges.

Thus, while in practice accepted combinations for passing a challenge may be even more flexible, Table 4 shows the passing combinations that we found to always hold true.

About 125 results (0.65 seconds)



Image size:
100 x 100

Find other sizes of this image:
[All sizes](#) - [Small](#) - [Medium](#) - [Large](#)

Best guess for this image: [wine and blood](#)

[Riedel Vinum Cabernet/Merlot/Bordeaux Wine Glasses \(Set ...](#)

[www.wineenthusiast.com](#) › [Glassware](#) › [Wine Glasses](#) › [Red Wine Glasses](#) ▾

★★★★★ Rating: 4.8 - 52 reviews - \$54.90

The Riedel Vinum Cabernet / Merlot / Bordeaux wine glass is ideal for full-bodied, complex red wines that are high in alcohol and tannins. The generous size ...

[Amazon.com: Riedel Wine Series Cabernet/Merlot Glass ...](#)

[www.amazon.com](#) › ... › [Glassware & Drinkware](#) › [Champagne Glasses](#) ▾

Amazon.com: Riedel Wine Series Cabernet/Merlot Glass, Set of 2: Red Wine Glasses: Kitchen & Dining.

Figure 3: Google Reverse Image Search (GRIS) for obtaining the description of an image.

4.2 Image repetition

During our experiments we came across several cases of images being repeated across challenges. We are some completely identical challenges: for each pair, the images and their ordering were exactly the same in both challenges. This suggests that challenges are not created “on-the-fly” but selected from a relatively small pool of challenges which is periodically updated.

We also found that images were being repeated across challenges. However, in most cases they had different MD5 values. Thus, we conducted a comparison using *perceptual hashes*, to identify identical even if they have different MD5 values. In all cases the images we detected seemed visually identical. We identified that 20% is a redundant images. The most re-used candidate image was seen 12 times.

5 Image captcha breaker

Recent advancements in the area of computer vision have demonstrated impressive results [7, 8, 9, 10], and image annotation services that leverage such techniques have emerged. As these services are bound to become even more widely available, we explore if they can be used for solving image captchas.

To solve an image captcha, our system has to automatically identify which of the given images are semantically similar to the sample image. Upon receiving a challenge we extract the sample and candidate images, and the *hint* that describes the content of the sample image (e.g., “wine”). Next, all images are passed to an image annotation module.


5.1 Image annotation services and libraries

There are several free online services and libraries that offer functionality, ranging from assigning tags (keywords) to providing free-form descriptions of images services. Example outputs are shown in Table 5.

GRIS

The Google Reverse Image Search, built by Krizhevsky et al. [11], offers the ability to conduct a search-based on an image. If the search is successful it may return a “*best guess*” description of the image (see Figure 3) along with a list of websites where the image is contained, and other available sizes of that image. While this is not part of Google’s public API, we identified the format of the search URL so our module can replicate the functionality. When conducting the reverse search for the 9 candidate images, we also collect the *page titles* of the webpages

Table 5: Example output from each image annotation module, with tags sorted in descending order of confidence.

	GRIS	Alchemy	Clarifai	TDL	NeuralTalk	Caffe
	wine and blood	wine, glass	glass, red wine, wine, merlot, liquid, bottle, still, glassware, alcohol, drink, wineglass, beverage, pouring, white wine, cabernet, taste, leaded glass, dining, party, vino	red wine, goblet, wine bottle, punching bag, beer glass, perfume, balloon	a glass of wine sitting on top of a table	red wine, wine, alcohol, drug of abuse, drug, red wine, punching bag, beaker, cocktail shaker, table lamp

that contain the image, as an extra piece of information. If available, we also obtain a higher resolution version of each image, as it increases the accuracy of the image annotation modules. If a non-English description is returned, we convert it to English through the automatic language detection feature of Google Translate.

Clarifai

is built on the deconvolutional networks by Zeiler et al. [12], and returns a set of 20 tags describing the image along with a confidence score for each tag.

Alchemy²

is also built upon deep learning, and offers an API for image recognition. For each submitted image, the service returns a set of tags and a confidence score for each tag. In our experiments, images received up to 8 tags, which tend to be specific (e.g., “wine”).

TDL³

Srivastava and Salakhutdinov [10] have released a system for demonstrating the image classification capabilities of their deep learning system. For each image, 8 tags are returned along with a confidence score.

NeuralTalk

Karpathy and Fei-Fei [8] developed NeuralTalk, a Recurrent Neural Network architecture for generating free-form descriptions of an image’s contents. We break the returned description down into individual words.

Caffe

Jia et al. [13] have released Caffe, a deep learning framework, which we also leverage for processing images locally. Caffe returns a set of 10 labels; 5 with the highest confidence scores and 5 that are more specific as keywords but may have lower confidence scores.

5.2 Tag Classifier

The returned tags do not always exactly match the description (i.e., hint) given by reCaptcha for a challenge. To overcome this, we (unsupervised) machine learning approach and leverage machine learning to develop a classifier that can “guess” the content of an image based on a subset of the tags. Specifically, we opted for the Word2Vec word vectors proposed by Mikolov et al. [14] for finding the similarity between tags and hints. During the training of our classifier, we modeled and represented each word (tag and hint) as a real vector in vector space. Each tag

²<http://www.alchemyapi.com>

³<http://deeplearning.cs.toronto.edu/>

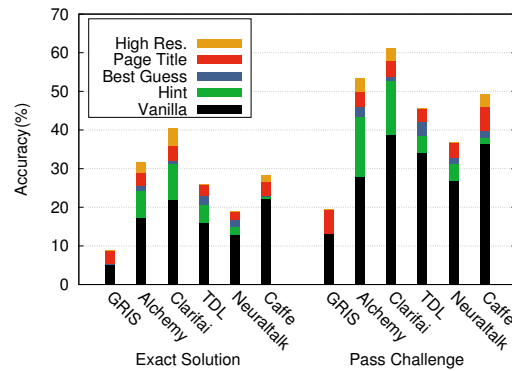


Figure 4: Accuracy of simulated attack for different combinations of modules and data against the image reCaptcha.

assigned to an image is paired with the correct hint and all tags and hint are given as input to the model. Once the classifier has been trained, it can be used to predict the similarity of the captcha's hint and the tags by computing the cosine similarity between their corresponding word vectors, with the goal of identifying subsets of tags from each image that have been associated with the hint during the training phase. Thus, our classifier allows our system to select images with similar content even if the annotation system does not return tags that exactly match the given hint.

5.3 History Module

Many images are actually re-used across challenges. As such, we manually create a labelled dataset with images and their tag from challenges we collect. Each image is annotated with the hint given in the challenge that describes the content (e.g., cat, soup). We also maintain a `hint_list` that contains the hints we have seen.

5.4 Solution

Each module assigns the candidate images to one of 3 sets: `select`, `discard` or `undecided`. First, we collect information for all the images through GRIS. Next, if a hint is not provided, we search for the sample image in the labelled dataset to obtain one if possible. The history module searches for the candidate images in our labelled dataset and, if found, compares their tag to the hint and adds those that match to the `select` list. The remaining images are compared to the `hint_list` and added to the `discard` list if there is a match. An image annotation module then processes all the images and assigns them tags. If one of the tags matches the hint the image is added to the `select` set. If it matches one of the other tags in the `hint_list` it is added to the `discard` set. A similar process is conducted when we leverage the best guess and page title results for each image. Once all the modules have completed, the system processes the results and merges the sets from the modules. Information is given a different "weight" (e.g., title pages have the lowest confidence) overcome cases where modules assign the same image to a different set. If there is not an adequate number of images in the `select` set, the system picks the remaining images from the `undecided` set. That is done either through our tag classifier, or by selecting the images that have the most overlapping (i.e., common) tags with the sample image.

5.5 Evaluation

Attack simulation

We simulated our attack on the dataset we have downloaded. Figure 4 breaks down the accuracy for each module and type of information. Here, we used a `hint_list` with the hints that we had come across in our experiments, but did not use the history module or tag classifier images.

As we showed in Table 4, reCaptcha is flexible and a solution is considered correct even if it contains a wrong image along with the correct ones. As such, we configured the system to select 3 images for the solution so as to fall within the “relaxed” limits. The *Pass Challenge* bars represent the outcome of the attacks.

We started with a baseline measurement for the “vanilla” version of each module which selects images based on the overlapping tags; for GRIS, this also entails using the hint provided in the challenge and the best guess returned by the reverse image search. In general, the success rate for GRIS is limited by the number of candidate images for which we can obtain a best guess description. For the other modules, the baseline attack selects the 3 images that have the most common (overlapping) tags with the sample image. When using the hint, best guess and page titles, the Alchemy module passed 49.9% of the challenges, while Clarifai passed 58%. Caffe is also very effective, solving 45.9% of the challenges. The hint has a significant effect in most cases, increasing the accuracy by 1.5-15.5% depending on the annotation system.

We explored how the attack’s accuracy is impacted by supplying the image annotation module with higher resolution versions of the images. We were able to automatically obtain a higher resolution version of 2,909 images from the 700 challenges. Out of those, 371 corresponded to the sample image. The high resolution images increased the attacks’ success, with Alchemy and Clarifai passing 53.4% and 61.2% of the challenges respectively. TDL is less accurate achieving 45%, while Caffe increases to 49.1%.

We also measured the number of challenges our system would pass if there was no flexibility. Since in most cases the solution consists of 2 images, we tuned the system to select 2 images for each challenge. The *Exact Solution* bars in Figure 4 present the results, and we can see that all the image annotation services were quite effective in identifying the correct images. Clarifai is the most effective as it selected the exact set of images in 40.2% of the challenges, while Alchemy reached 31.5% and Caffe 28.3%.

Tag classifier

To quantify the effectiveness of our tag classifier as part of our captcha-breaking system, we followed a 10-fold validation approach for training and testing our classifier on the dataset of 700 labelled image captchas. In our first experiment, we skipped the other image selection steps, and relied solely on the classifier for selecting the images. For each image, the classifier received as input the hint and the set of tags, and returned a “similarity” score; we selected the 3 images with the highest score. Our attack provided an exact match solution for 26.28% ($\sigma = 7.09$), and passed 44.71% ($\sigma = 6.39$) of the challenges. In the second experiment, we incorporated our classifier into our system, and used the classifier-based selection as a replacement of the overlapping-based selection of images from the `undecided` set. When using the classifier, our attack’s average accuracy for Clarifai reached 66.57% ($\sigma = 7.53$), resulting in an improvement of about 5.3%. The classifier is more effective than the overlap approach, as it identifies specific subsets of tags that are associated with each hint, instead of the more simplistic metric of the number of common tags. Furthermore, the use of the classifier does not impact the performance of the attack as the duration is increased by ~ 0.025 sec.

Live attack

To obtain an exact measurement of our attack’s accuracy, we run our automated captcha-breaker against reCaptcha. We employ the Clarifai service as it shows the best result amount other services.

Labelled dataset. We created a labelled dataset to exploit the image repetition. We manually labelled 3,000 images collected from challenges, and assigned each image a tag describing the content. We selected the appropriate tags from our `hint_list`. We used pHash for the comparison, as it is very efficient, and allows our system to compare all the images from a challenge to our dataset in 3.3 seconds.

We ran our captcha-breaking system against 2,235 captchas, and obtained a 70.78% accuracy. The higher accuracy compared to the simulated experiments is, at least partially, attributed to the image repetition; the history module located 1,515 sample images and 385 candidate images in our labelled dataset.

Average run time. Our attack is very efficient, with an average duration of 19.2 seconds per challenge. The most time consuming phase is running GRIS, consuming phase, as it searches for all the images in Google and processes the results, including the extraction of links that point to higher resolution versions of the images.

Offline mode. We also evaluated our attack in an *offline mode*, where we did not use any online annotation services or Google’s reverse image search; we relied solely on the local library, our labelled dataset, and our skip-gram classifier. with the two libraries, NeuralTalk and Caffe. When using Caffe and our classifier, our system solved



Figure 5: Image captcha by Facebook.

41.57% ($\sigma = 4.28$) of the image captchas, while the attack duration increased to 20.9 seconds per challenge. While NeuralTalk is similarly accurate at $\sim 40\%$, it introduces a large increase to the attack's duration; specifically, the duration of the attack increases to 117.8 seconds, as NeuralTalk requires an average of 110.9 seconds to process the 10 images. However, leveraging the capabilities of a GPU for the computations will improve the performance and reduce the duration.

Thus, adversaries can deploy accurate and efficient attacks against the image reCaptcha without relying on external services, which may require payment for processing large collections of images or report suspicious actions.

6 Applicability

The basis of our attack is can also be applied to other schemes, by extracting the semantics of images, we can construct attacks against other image-based captchas, such as Facebook captcha (Figure 6). The image captcha is shown to users when they send messages to other users that contain suspicious URLs, they must pass an image captcha first. Facebook's image captcha follows the same approach with reCaptcha, where users have to identify which images (out of 12 candidates) have content that match a given hint. There are, however, some differences. Facebook resizes the images dynamically in HTML, allowing access to the high resolution versions. Also, no sample image is shown. The system allows the same flexibility rules as reCaptcha. The number of correct images varies from 2 to 10, with 5-7 being the most common cases. Accordingly, we tweak our solution algorithm to only select the images contained in the `select` set, and not to opt for a specific number of images.

Over 200 Facebook image captchas, our attack with Clarifai achieves the highest accuracy with 83.5%. The higher accuracy, compared to reCaptcha, is due to two characteristics. First, the higher resolution of the candidate images and second, the use of completely unrelated images when creating the challenge facilitates the discarding of the incorrect options. On the other hand, reCaptcha opts for images that belong to the same category (e.g., all are some type of food) which renders the distinction more difficult.

7 Economic analysis

Since captcha-breaking is driven by monetary incentives, we evaluate our findings from an economic perspective and compare our attack to a captcha-solving service.

7.1 Image captcha

We compare our performance to that of Decaptcher, the (self-reported) oldest captcha-solving service. We selected Decaptcher for two reasons. First, it supports the image reCaptcha, charging \$2 per 1000 solved captchas. Second,

Table 6: Returned errors from Decaptcher service

Detail	Out of 700 challenges
Error	
System overload error	147 (21.00%)
Timeout error	88 (12.57%)
Success	
Exact match	258 (36.85%)
Pass challenges	321 (44.30%)

previous work [15] found it to be the most accurate solving service (tied with another service), rendering it a suitable candidate for comparison.

We submitted the 700 image captchas to Decaptcher and measured the response time and accuracy (taking into account the solution flexibility). Interestingly, Decaptcher rejected many captchas we submitted.

Interestingly, some of our submitted challenges rejected due to the service being overloaded, and had to be re-submitted at a later time, and received a time-out error as the solvers did not provide an answer in the time window allocated by the service. 258 challenges (36.85%) were an exact match. When taking into account the flexibility, 321 (44.3%) of the captchas were solved. The average solving time for the challenges that received a solution was 22.5 seconds. While the accuracy may increase over time as the human solvers become more accustomed to the image reCAPTCHA, it is evident that our system is a cost-effective alternative. Nonetheless, *our completely offline captcha-breaking system is comparable to a professional solving service in both accuracy and attack duration, with the added benefit of not incurring any cost on the attacker.*

7.2 Checkbox captcha

Assuming a selling price of \$2 per 1,000 solved captchas, our token harvesting attack could accrue \$104 - \$110 daily, per host (i.e., IP address). By leveraging proxy services and running multiple attacks in parallel, this amount could be significantly higher for a single machine.

8 Ethics and disclosure

We have disclosed a report with our findings and recommendations to Google, in an effort to assist them in making reCAPTCHA more robust to automated attacks. Following our disclosure, reCAPTCHA altered the safeguards and the risk analysis process to mitigate our large-scale token harvesting attacks. They also removed the solution flexibility and sample image from the image captcha for reducing the attack's accuracy. We have also informed Facebook, but have not been notified of any changes. Overall, we hope that sharing our findings will help initiate the much-needed discussion between researchers and industry regarding the future of captchas.

Acknowledgements

This work was supported by the NSF under grant CNS-13-18415. Author Suphannee Sivakorn is also partially supported by the Ministry of Science and Technology of the Royal Thai Government. Any opinions, findings, conclusions, or recommendations expressed herein are those of the authors, and do not necessarily reflect those of the US Government or the NSF.

References

- [1] L. Von Ahn, B. Maurer, C. McMillen, D. Abraham, and M. Blum, "reCAPTCHA: Human-based character recognition via web security measures," *Science*, vol. 321, no. 5895, 2008.
- [2] Google Online Security Blog, "Are you a robot? Introducing "No CAPTCHA reCAPTCHA"," <http://googleonlinesecurity.blogspot.com/2014/12/are-you-robot-introducing-no-captcha.html>.

- [3] E. Bursztein, J. Aigrain, A. Moscicki, and J. C. Mitchell, "The end is nigh: Generic solving of text-based CAPTCHAs." in *USENIX WOOT '14*.
- [4] I. J. Goodfellow, Y. Bulatov, J. Ibarz, S. Arnaud, and V. Shet, "Multi-digit number recognition from street view imagery using deep convolutional neural networks," in *CoRR '13*.
- [5] K. Mowery and H. Shacham, "Pixel perfect: Fingerprinting canvas in html5," in *W2SP '12*.
- [6] E. Homakov. The No CAPTCHA problem. <http://homakov.blogspot.in/2014/12/the-no-captcha-problem.html>.
- [7] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: A neural image caption generator," in *CoRR '14*.
- [8] A. Karpathy and L. Fei-Fei, "Deep visual-semantic alignments for generating image descriptions," in *CoRR '14*.
- [9] M. M. Kalayeh, H. Idrees, and M. Shah, "NMF-KNN: Image Annotation Using Weighted Multi-view Non-negative Matrix Factorization," in *CVPR '14*.
- [10] N. Srivastava and R. Salakhutdinov, "Multimodal learning with deep boltzmann machines," *Journal of Machine Learning Research*, vol. 15, pp. 2949–2980, 2014.
- [11] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS '12*.
- [12] M. D. Zeiler, G. W. Taylor, and R. Fergus, "Adaptive deconvolutional networks for mid and high level feature learning," in *ICCV '11*.
- [13] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding."
- [14] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *CoRR '13*.
- [15] M. Motoyama, K. Levchenko, C. Kanich, D. McCoy, G. M. Voelker, and S. Savage, "Re: CAPTCHAs: understanding captcha-solving services in an economic context," in *USENIX Security '10*.