FEATURE

EVADING SPAMASSASSIN WITH OBFUSCATED TEXT IMAGES

Battista Biggio, Giorgio Fumera, Ignazio Pillai, Fabio Roli and Riccardo Satta
University of Cagliari, Italy

Most spam filters consist of a set of modules which analyse different characteristics of an email (sender's address, header, body, attachments) to determine whether to label it as spam or legitimate mail (ham). In many filters the module devoted to the analysis of the email's textual content is based on statistical text categorization techniques. The application of such techniques to the spam-filtering task has been widely investigated by the machine-learning community over the past ten years (see for instance [1–4]).

To circumvent filtering modules based on text analysis, spammers started to embed their messages into attached images – this trick is called image-based spam (or image spam). Moreover, text images are often obfuscated using different techniques to render OCR tools ineffective without compromising human readability.

To deal with image spam, modules based either on OCR tools or on low-level image processing techniques have been introduced in spam filters. However, there is not yet a clear understanding of the effectiveness of image spam with obfuscated text in evading anti-spam filters that are based on OCR tools. As a consequence, there is also a lack of clear guidelines for the development of filtering modules against image spam.

In this work we focus on *SpamAssassin* [5], one of the most well known and widespread open-source spam filters, and provide a thorough and systematic analysis of its vulnerability to image spam with obfuscated text when an OCR-based filtering module is used. To this aim, we used a dataset of real spam emails with artificially generated images. The images were obtained by reproducing three kinds of the most commonly used text obfuscation techniques observed in real spam emails, and by varying the degree of obfuscation of each image in a suitable range.

We assessed both the performance of the whole *SpamAssassin* filter and that of the stand-alone OCR-based module. Two open source OCR tools used in *SpamAssassin* were considered: gocr and tesseract. Finally, we evaluated the *SpamAssassin* performance on a dataset of real spam emails with real attached images. The results of our analysis provide some useful insights into the effectiveness of OCR-based modules in spam filters, as well as some suggestions for the development of effective image spam-filtering modules.

2 SPAMASSASSIN ARCHITECTURE

SpamAssassin is made up basically of a set of 'if-then' rules, each one of which is dedicated to a different characteristic of an email which can be useful to determine either its spamminess or its hamminess. A score is associated with each rule: higher scores denote a higher degree of spamminess. The scores of the rules which fire (rules whose antecedent is true) are summed up, and if the sum exceeds a predefined threshold the email is labelled as spam (see Figure 1). The score of each rule belongs to a given range (possibly different for each rule).

To improve the effectiveness of the filter on their specific email traffic, users can change the score range either manually or by using a built-in procedure named mass-check [6], which must be carried out on a user-defined dataset. The mass-check procedure sets the score range by taking into account the detection rate and the reliability of each rule. It is worth pointing out that the scores of *SpamAssassin* rules can either be binary or continuous-valued.

The *SpamAssassin* architecture is easy for users to understand and is extremely flexible, since it is possible to plug and unplug rules without changing the global configuration of the filter, i.e. without changing the score of the rules (although it should be pointed out that leaving weights unchanged as the rule set is modified may not be the best choice).

Among the set of rules available in *SpamAssassin*, there are currently four plug-ins dedicated to image spam: OCR plug-in, OCRtext, FuzzyOCR and BayesOCR.

The OCR plug-in looks for a set of predefined keywords in the text extracted by an OCR tool from the image attached to an email (two open source OCR tools can be used: gocr [7] and tesseract [8]). A binary scoring system is used: if at least one keyword is found, the plug-in sets its score to the value of 3, otherwise its score is set to 0 (these are the predefined values). OCRtext uses the same idea as the OCR plug-in, but also checks some other properties of the attached image unrelated to content (such as aspect ratio and file size). FuzzyOCR works similarly to OCRtext, but

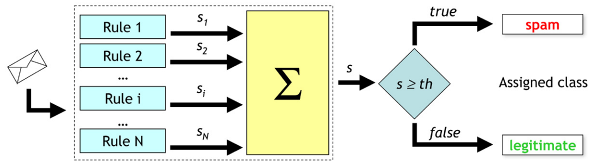


Figure 1: SpamAssassin architecture. The scores associated with the firing rules are summed up and if the sum exceeds a predefined threshold the email is labelled as spam, otherwise it is labelled as ham.

carries out a fuzzy matching between keywords and the extracted text, which in principle makes it more robust to OCR errors. Its scoring system is different as well. BayesOCR sends the extracted text to the text classification module of *SpamAssassin*, using the corresponding score. It is based on the work described in [9].

In our experiments we chose to use the OCR plug-in since it does not perform any pre-processing on the attached image, and carries out a simple keyword matching between the extracted text and a list of keywords in its database. This allowed us to assess directly how obfuscation techniques affect the performance of an OCR-based filtering module, and consequently the overall performance of the spam filter, without the need for taking into account the influence of other techniques (like image pre-processing or text categorization).

Finally, it should be pointed out that *SpamAssassin* has four different working configurations obtained by using or not using the text classifier based on the naïve Bayes classification technique (which is often used in spam filters, see for instance [1, 3]), and by using or not using Internet-related rules, e.g. DNS blacklist checking. In our experiments we assessed the performance of all four working configurations, using the default scoring system for each rule.

3 AUTOMATIC GENERATION OF SPAM IMAGES WITH OBFUSCATED TEXT

To carry out a systematic analysis of the effectiveness of image spam with obfuscated text in evading detection by a spam filter with an OCR-based tool, it is necessary to collect a large dataset of spam emails with attached images including representative kinds of obfuscation techniques.

Unfortunately, however, no freely available benchmark dataset with these characteristics exists. In particular, freely available datasets of spam emails were collected when image spam was not yet widespread and thus contain at most a negligible number of messages with attached images. Moreover, even in a personal archive of spam emails it is difficult to find a representative set of obfuscation levels. For this reason we developed a software module for generating artificial spam images characterized by different kinds of obfuscation technique observed in real spam images, and by a degree of obfuscation which can be fine tuned. In particular, we focused on the three kinds of obfuscation techniques widely used by spammers.

3.1 Implementation of text obfuscation techniques

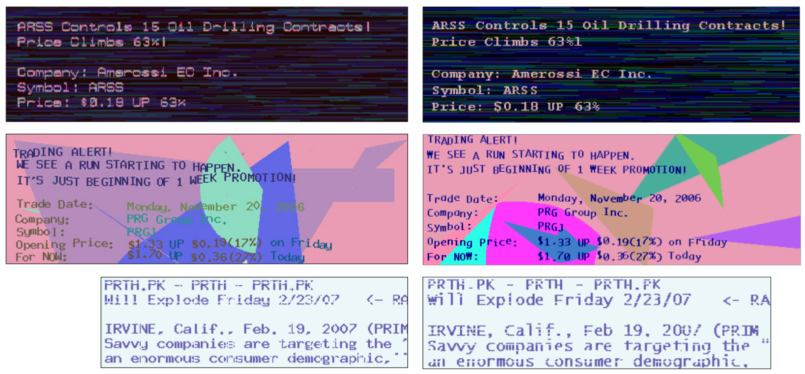
The first obfuscation technique (see Figure 2, top row) consists of making both the text and background colours non-uniform. In particular, the colour of each text pixel is chosen randomly, while the background is made up of horizontal segments of random width and one pixel height, whose colour is also chosen randomly. A Gaussian distribution is used for the grey-level value Y of each text pixel. The corresponding RGB values are then generated as:

$$Y = 0{:}299 * R + 0{:}587 * G + 0{:}114 * B \qquad (1)$$

A Gaussian distribution is also used for the width and the colour of each background segment, which was set as described above for text pixels. All the parameters of the above-mentioned probability distributions (mean and variance) were set as described in Section 3.2, with the aim of obtaining images similar to the ones observed in real spam emails, with different degrees of text obfuscation with respect to an OCR tool.

The second obfuscation technique consists of misaligning text characters over a non-uniform background made up of random shapes of different random colours (see Figure 2, middle row). Each text character was shifted vertically by a random amount (positive or negative). The perimeter of each background shape was obtained by connecting with straight lines pairs of points chosen randomly, until a given percentage of the image area was covered (different shapes can overlap). The colours of the background shapes were set as described above for the first obfuscation technique.

The third obfuscation technique consists of drawing horizontal segments of random length over a clean text image with uniform background. The segment colour is identical to the background colour: this results in cutting the



```
ARSS Controls 15 Oil Drilling Contracts!
Price Climbs 63%!

Company: Amerossi EC Inc.
Symbol: ARSS
Price: $0.18 UP 63%
```

```
ARSS Controls 15 Oil Drilling Contracts!
Price Climbs 63%1

Company: Amerossi EC Inc.
Symbol: ARSS
Price: $0.18 UP 63%
```

```
TRADING ALERT!
WE SEE A RUN STARTING TO HAPPEN.
IT'S JUST BEGINNING OF 1 WEEK PROMOTION!

Trade Date:      Monday, November 20, 2006
Company:         PRG Group Inc.
Symbol:          PRGJ
Opening Price:   $1.33 UP $0.19(17%) on Friday
For NOW:         $1.70 UP $0.36(27%) Today
```

```
TRADING ALERT!
WE SEE A RUN STARTING TO HAPPEN.
IT'S JUST BEGINNING OF 1 WEEK PROMOTION!

Trade Date:      Monday, November 20, 2006
Company:         PRG Group Inc.
Symbol:          PRGJ
Opening Price:   $1.33 UP $0.19(17%) on Friday
For NOW:         $1.70 UP $0.36(27%) Today
```

```
PRTH.PK - PRTH - PRTH.PK
Will Explode Friday 2/23/07   <- RA

IRVINE, Calif., Feb. 19, 2007 (PRIM
Savvy companies are targeting the "
an enormous consumer demographic, "
```

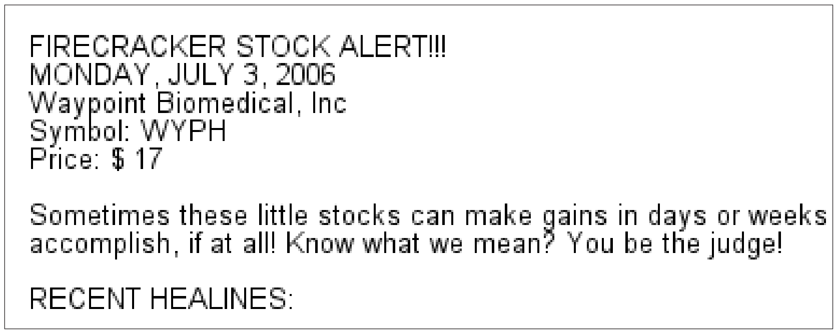
```
PRTH.PK - PRTH - PRTH.PK
Will Explode Friday 2/23/07   <- RA

IRVINE, Calif., Feb 19. 2007 (PRIM
Savvy companies are targeting the "
an enormous consumer demographic.
```

Figure 2: Examples of real spam images using three different text obfuscation techniques (left), and spam images generated with the proposed algorithm (right). It is easy to see that artificially generated spam images are very similar to the real ones.

text, breaking and splitting characters. The length of each segment, as well as the average horizontal and vertical distance between different segments, is chosen randomly from a Gaussian distribution.

3.2 Definition of the degree of obfuscation

For the purposes of this work, the parameters which control the three obfuscation techniques were set with the following rationale: first, we wanted to obtain images similar to the ones observed in real spam emails. In particular, the text embedded into such images, although obfuscated, must be readable by a human being. Second, a range of parameter values had to be defined to obtain different degrees of text obfuscation with respect to an OCR tool. This was accomplished as follows: we evaluated the performance of two OCR tools, gocr and tesseract, in terms of the word error rate (WER), which is a common measure of OCR performance [10]. For a given image with embedded text, WER is defined as the fraction of words not recognized correctly by the OCR. A word is considered correctly recognized only if all its characters are recognized correctly (in the right sequence).


FIRECRACKER STOCK ALERT!!!
MONDAY, JULY 3, 2006
Waypoint Biomedical, Inc
Symbol: WYPH
Price: $ 17

Sometimes these little stocks can make gains in days or weeks
accomplish, if at all! Know what we mean? You be the judge!

RECENT HEALINES:

Figure 3: Example of an image with clean text (degree of obfuscation = 0%).

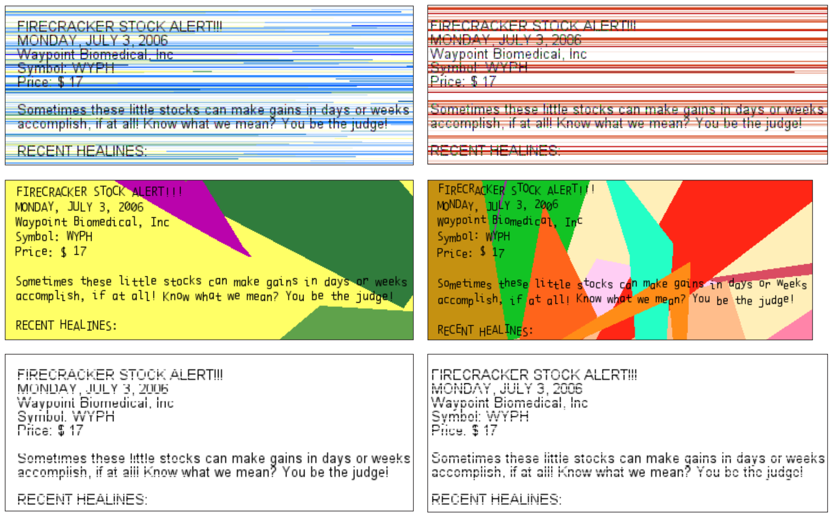


Figure 4: Examples of images obtained from the one shown in Figure 3 using the three obfuscation techniques considered in this work, and a degree of obfuscation equal to 50% (left) and 100% (right).

In these experiments we used a text composed of 80 different words (which is the typical length of an image spam text), excluding punctuation and accents.

For each obfuscation technique we assessed which of the corresponding parameters exhibited a significant correlation with WER. To this aim, we computed WER as a function of each single parameter, setting all the others to constant values (different constant values were evaluated). Afterwards, parameters which turned out not to be significantly correlated to WER were set to constant values so that the images obtained looked as similar as possible to real spam images (without compromising human readability).

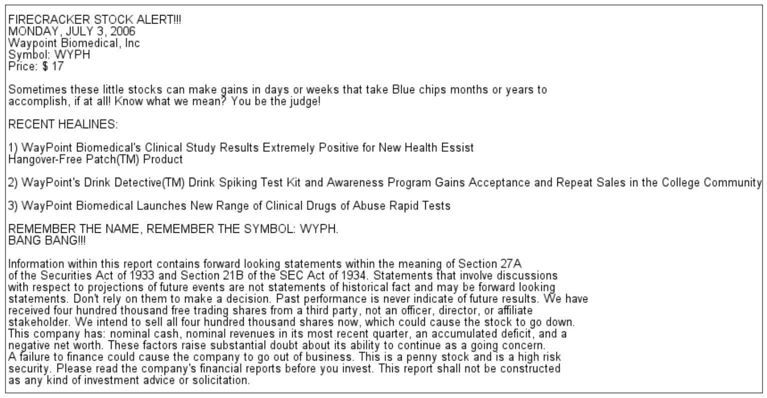
For parameters correlated to WER a range of values was defined so that the values at the end of the range, corresponding to the highest degree of obfuscation, led to images that were still readable by human beings. The degree of obfuscation was then formally defined as a percentage: to obtain an image with, say, a 60% degree of obfuscation, each parameter (among the ones correlated with WER) of the first and second obfuscation techniques was set to $l + 0{:}6 * (u - l)$, where l and u denote respectively the lower and upper values of the corresponding range (assuming that l corresponds to the lowest obfuscation caused by that parameter). The parameters of the third obfuscation technique were set similarly, but using a logarithmic scale for their range (the reason is that the relationship between the parameters and WER proved to be non-linear for this obfuscation technique).

Examples of the images obtained at different degrees of obfuscation for the image in Figure 3 are shown in Figure 4.

4 EXPERIMENTAL ANALYSIS

In this section we present experiments aimed at evaluating the effectiveness of image spam with obfuscated text in evading detection by the *SpamAssassin* filter equipped with an OCR plug-in.

Two experiments were carried out. In the first experiment, artificial spam images generated as described in Section 3 were used to assess systematically the performance of the single OCR plug-in and of the whole *SpamAssassin* filter as a function of the degree of obfuscation. In the second experiment a real dataset of image spam was used. Version 3.1.3 of *SpamAssassin* was used, with default settings and additional packages related to common collaborative spam-filtering networks, including RAZOR [11], PYZOR [12] and DCC [13]. For the OCR plug-in, the default keywords database was used.



FIRECRACKER STOCK ALERT!!!
MONDAY, JULY 3, 2006
Waypoint Biomedical, Inc
Symbol: WYPH
Price: $ 17

Sometimes these little stocks can make gains in days or weeks that take Blue chips months or years to accomplish, if at all! Know what we mean? You be the judge!

RECENT HEALINES:

1) WayPoint Biomedical's Clinical Study Results Extremely Positive for New Health Essist Hangover-Free Patch(TM) Product

2) WayPoint's Drink Detective(TM) Drink Spiking Test Kit and Awareness Program Gains Acceptance and Repeat Sales in the College Community

3) WayPoint Biomedical Launches New Range of Clinical Drugs of Abuse Rapid Tests

REMEMBER THE NAME, REMEMBER THE SYMBOL: WYPH.
BANG BANG!!!

Information within this report contains forward looking statements within the meaning of Section 27A of the Securities Act of 1933 and Section 21B of the SEC Act of 1934. Statements that involve discussions with respect to projections of future events are not statements of historical fact and may be forward looking statements. Don't rely on them to make a decision. Past performance is never indicate of future results. We have received four hundred thousand free trading shares from a third party, not an officer, director, or affiliate stakeholder. We intend to sell all four hundred thousand shares now, which could cause the stock to go down. This company has: nominal cash, nominal revenues in its most recent quarter, an accumulated deficit, and a negative net worth. These factors raise substantial doubt about its ability to continue as a going concern. A failure to finance could cause the company to go out of business. This is a penny stock and is a high risk security. Please read the company's financial reports before you invest. This report shall not be constructed as any kind of investment advice or solicitation.

Figure 5: Spam image with clean embedded text and several well-known spam keywords, e.g. 'stock', 'company', which are in the OCR plug-in keyword list. This image was used as source for generating images with different obfuscation techniques and levels.

4.1 Experiments with artificial spam images

The first experiment we carried out was aimed at assessing the ability of image spam with obfuscated text to evade detection by *SpamAssassin* equipped with an OCR plug-in (gocr was used as OCR tool).

The analysis was carried out using spam images with different degrees of text obfuscation. To this aim we used a dataset of 1,779 real spam emails with attached images received in the authors' personal mailboxes between July 2006 and February 2007.

To carry out a systematic analysis of the *SpamAssassin* performance as a function of the degree of obfuscation, we substituted the original images attached to each email with artificial images obtained as described in Section 3. More precisely, we first generated an image which was easily detectable as spam by the OCR plug-in: it was a clean image (degree of obfuscation equal to 0%) with embedded text containing several keywords known to be included in the OCR plug-in database, and using a font easily recognized by gocr. The image is shown in Figure 5. This image was attached to each of the 1,779 spam emails to create a dataset with the most favourable conditions for the OCR plug-in, as a baseline for the subsequent comparison with obfuscated images.

Then we modified the image using the three obfuscation techniques described in Section 3, and ten different degrees of obfuscation ranging from 10% to 100% in steps of 10%. For each obfuscation technique and each degree of obfuscation, we generated 1,779 spam images and attached them to the 1,779 spam emails (note that each image was different due to the random choice of the obfuscation parameters).

This led to three datasets, one for each obfuscation technique; each dataset was made up of ten subsets (one for each degree of obfuscation) containing the same 1,779 spam emails, and the emails of each subset contained spam images characterized by the same degree of obfuscation.

We remind the reader that the OCR plug-in has a binary scoring system: it outputs a default value of 3 if the input image is deemed to be spam, and 0 otherwise. Moreover, we point out that even if the OCR plug-in outputs a score value of 3, the input email is not necessarily labelled as spam by *SpamAssassin*, since the final decision also depends on the output of the other modules (which can be negative), and on the fact that the default *SpamAssassin* threshold is 5. Similarly, an email can be labelled as spam by *SpamAssassin* even if the output of the OCR plug-in is 0. For these reasons, in our experiments we evaluated both the performance of the whole *SpamAssassin* filter and the performance of the individual OCR plug-in. Performances were evaluated in terms of the fraction of spam emails correctly labelled as spam, which equals 1 minus the false negative (FN) rate, defined as the fraction of spam emails incorrectly labelled as ham.

Figure 6 shows the OCR plug-in detection rate (1–FN), for each obfuscation technique, as a function of the degree of obfuscation. Note first that at zero degrees of obfuscation, all images were correctly recognized as spam by the OCR plug-in, as desired. It can also be seen that, even if the behaviour of 1–FN depends on the specific obfuscation technique, it almost always decreases as the degree of obfuscation increases (with few exceptions). The decrease is very rapid for the first obfuscation technique, slower for the second one (note that even at 100% degree of obfuscation, about 66% of spam images are still recognized), and abrupt for the third one, once the degree of obfuscation exceeded 80%. Note that at 100% of the degree of obfuscation (which does not compromise human readability, as explained in Section 3.2) the OCR plug-in was not able to recognize any

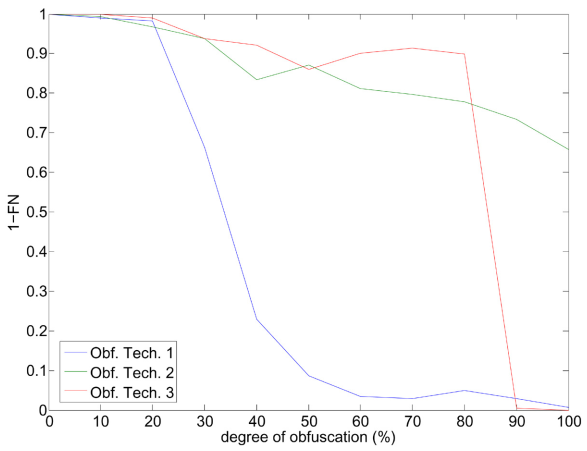


Figure 6: Spam detection rate of the OCR plug-in as a function of the degree of obfuscation for the three obfuscation techniques considered in the experiments.

spam email when the first and third obfuscation techniques were used. This provides clear evidence that some obfuscation techniques are effective against detection by the OCR plug-in, even with low degrees of obfuscation (this is the case in the first obfuscation technique).

Consider now the performance of the whole *SpamAssassin* filter. In our experiments we analysed its behaviour for each of the four configurations discussed in Section 2. These configurations are denoted in the following as 'bayes' (only the naïve Bayes text classification module was used), 'net' (only Internet-related rules were used), 'bayes 1–FN + net' (both kinds of rules were used) and 'local' (none of these rules was used).

The naïve Bayes text classifier was trained on 5,273 spam emails collected in the authors' mailboxes (from November 2005 to June 2006) and 3,515 ham emails taken from the *Enron* dataset [14, 15] (since the naïve Bayes is a statistical classifier, we chose a 2:3 proportion between ham and spam training emails, which, according to recent estimates, is similar to the proportion observed in real email traffic).

In Figure 7 we report the 1–FN values as a function of the degree of obfuscation for each of the four configurations and each of the three obfuscation techniques. First, as one might expect, it can be seen that the 'bayes + net' configuration is the most effective, while 'local' is the least effective. Interestingly, the use of Internet-related rules only ('net') significantly outperformed the use of the naïve Bayes module only ('bayes').

Consider now the performance of *SpamAssassin* with respect to the obfuscation techniques: it is easy to see that the following features are similar to the behaviour of the OCR plug-in. The rate of correctly recognized spam emails almost always decreases for increasing degrees of obfuscation. Such a decrease is smooth for the first and second obfuscation techniques, while it is abrupt for the third technique.

Finally, the second technique proved to be less effective in evading *SpamAssassin* (since it results in higher 1–FN values at 100% degree of obfuscation). However, a remarkable difference appears with respect to the OCR

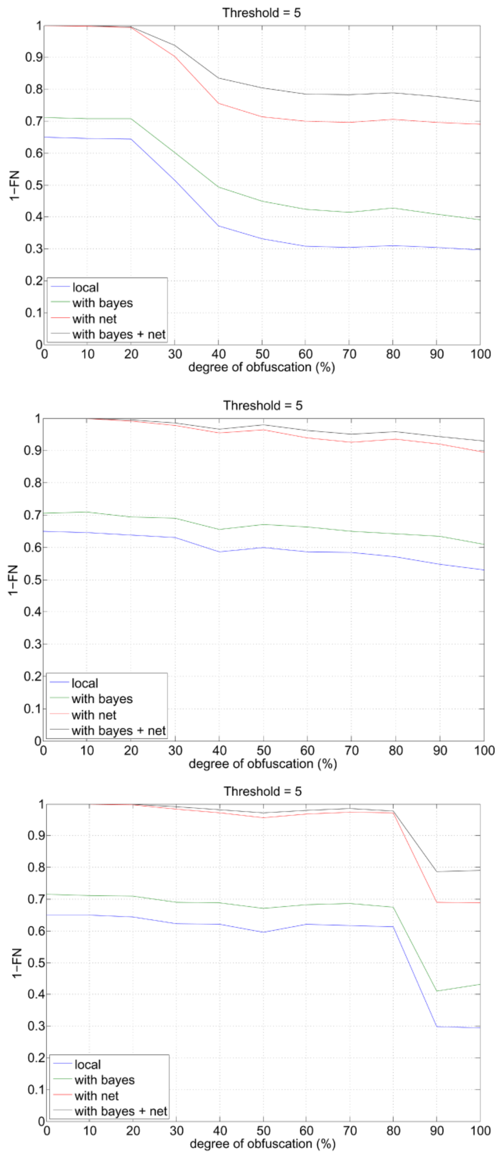


Figure 7: SpamAssassin detection rate versus the degree of obfuscation, for obfuscation techniques 1 (top), 2 (middle) and 3 (bottom), and the four different SpamAssassin configurations, using the default threshold value 5.

plug-in performance: even at 100% degree of obfuscation a significant fraction of spam emails is correctly recognized by *SpamAssassin* when the first and third obfuscation techniques are used, although in these conditions the OCR plug-in labelled all the images as ham (see Figure 6). This means that a large fraction of image spam (between 75% and 90% in our experiments across the different obfuscation techniques when the 'bayes + net' configuration was used), can be recognized even if the OCR plug-in is evaded, thanks to the other filtering rules. Instead, the remaining fraction of image spam emails (from 10% to 25%, depending on the obfuscation technique) can be detected by *SpamAssassin* only thanks to the OCR plug-in (the lower the degree of obfuscation, the higher the detection rate).

To sum up, our experimental results suggest that *SpamAssassin* is rather robust against image spam (perhaps more robust than one might think), even if its OCR-based filtering module can be evaded quite easily using obfuscated text. Using an OCR-based filtering module can improve *SpamAssassin*'s detection capability further if text embedded into images is clean or exhibits very low degrees of obfuscation.

4.2 Experiments with real spam images

In this section we provide an evaluation of *SpamAssassin* on a real dataset of image spam, in order to assess its performance in a realistic working environment. In this case we compare the performance of *SpamAssassin* with and without the OCR plug-in. We predicted that the improvement in spam detection rate due to the use of an OCR-based plug-in would be lower than the maximum one observed in the previous experiments (corresponding to clean spam images), given that many real spam images contain obfuscation techniques. For these experiments we used the same emails as in the previous experiments plus 253 emails with more than one attached image.

In Table 1, rows (a) and (b), we report the 1–FN values of *SpamAssassin* respectively with and without using the OCR plug-in for the four configurations explained above. Focusing on the most effective configuration ('bayes + net', fourth column), it can be seen that over 80% of image spam emails were recognized by *SpamAssassin* without the OCR plug-in. When the OCR plug-in was used, only 6% more image spam emails were detected (this percentage is reported in row (c)).

To further analyse these results, in row (d) we report the percentage of emails correctly labelled as spam by both the OCR plug-in and *SpamAssassin*, and in row (e) the percentage of spam emails correctly labelled as spam by the OCR plug-in and mislabelled as ham by *SpamAssassin*. It can be seen that the OCR plug-in detects only about 25% of image spam (and, as shown in row (c), only for 6% of spam emails was such detection useful for the overall *SpamAssassin* filter). As shown by row (e), the detection of a spam image by an OCR plug-in was always sufficient for it to be labelled correctly as spam by *SpamAssassin*.

	local	net	bayes	bayes+net
(a)	41.3%	51.4%	78.4%	86.8%
(b)	31.6%	46.2%	70.3%	80.8%
(c)	9.7%	5.2%	8.1%	6.0%
(d)	15.8%	15.9%	25.3%	25.3%
(e)	9.5%	9.4%	0%	0%

Table 1: Percentages of spam emails detected by SpamAssassin using the OCR plug-in (a); spam emails detected without using the OCR plug-in (b); spam emails correctly labelled by SpamAssassin only when the OCR plug-in was used (c); spam emails labelled as spam both by the OCR plug-in and SpamAssassin (d); spam emails labelled as spam only by the OCR plug-in (e).

The above results seem to confirm that most image spam can be detected by *SpamAssassin* even without using an OCR-based filtering module (that is, they are recognized for characteristics other than the text embedded into images). Nevertheless, there is also evidence that such kinds of module can be useful against those kinds of image spam in which text obfuscation techniques are less likely to be used (e.g. phishing emails, which, to be effective, must look as if they come from reputable senders, and thus should be as 'clean' as possible).

It should also be noted that a higher effectiveness than that exhibited by the OCR plug-in could be attained by using more effective OCR tools (although this could lead to an undesirable higher computational complexity), perhaps tailoring them to the characteristics of image spam (mainly low-resolution images), and exploiting different text analysis techniques from the simple keyword detection carried out by the OCR plug-in.

However, we believe that different techniques should be used to detect the percentage of image spam with obfuscated text which currently evades a filter like *SpamAssassin* (about 15% in our experiments on real spam emails). For instance, techniques based on pattern recognition and computer vision techniques can be used, like the ones in [16, 17], and the ones investigated by the authors in [18–20], which are aimed specifically at detecting the presence of obfuscated text. A combination of such kinds of approach and OCR-based approaches could also allow a tradeoff between effectiveness and efficiency. A hierarchical architecture can be devised for a spam filter, in which computationally demanding OCR tools are used only if the email has not been recognized as spam by other techniques.

5 CONCLUSIONS

We assessed the performance of *SpamAssassin* equipped with an OCR plug-in against image spam with obfuscated text. We used an artificial image spam generator to simulate three different obfuscation techniques and generate text images with different degrees of obfuscation. Those techniques were studied, and their effectiveness in evading *SpamAssassin* was assessed.

We showed how the first and third obfuscation techniques are more effective against *SpamAssassin* and the OCR plug-in (using gocr) than the second technique. We carried out an extended experimental analysis over a real image spam stream, which underlined the relatively low usefulness of an OCR-based plug-in in the *SpamAssassin* architecture, mainly due to the low effectiveness of gocr for this task. However, we still believe that an OCR-based plug-in should be present in anti-spam filters, because it can be helpful in identifying that kind of image spam which does not usually contain obfuscated text, e.g. phishing emails, unless other more efficient approaches are proposed. Moreover, more complex OCR-based plug-ins can achieve better performance.

Finally, some useful suggestions to improve spam filtering efficiency have been proposed: we suggest a hierarchical architecture to analyse image spam, using obfuscation content detectors and OCR plug-ins, which we are currently investigating.

REFERENCES

[1] Sahami, M.; Dumais, S.; Heckerman, D.; Horvitz, E. A Bayesian approach to filtering junk e-mail. AAAI Technical Report WS-98-05, Madison, Wisconsin, 1998.

[2] Drucker, H.; Wu, D.; Vapnik, V. N. Support vector machines for spam categorization. IEEE Transactions on Neural Networks, 10(5):1048–1054, 1999.

[3] Graham, P. A plan for spam, 2002. http://paulgraham.com/spam.html.

[4] Androutsopoulos, A.; Koutsias, J.; Cbandrinos, K. V.; Spyropoulos, C. D. An experimental comparison of naive bayesian and keyword-based anti-spam filtering with personal email messages. Proceedings of the ACM International Conference on Research and Developments in Information Retrieval, pp.160–167, 2000.

[5] http://spamassassin.apache.org/.

[6] http://wiki.apache.org/spamassassin/MassCheck.

[7] Available at http://jocr.sourceforge.net/.

[8] Available at http://code.google.com/p/tesseract-ocr/.

[9] Fumera, G.; Pillai, I.; Roli, F. Spam filtering based on the analysis of text information embedded into images. Journal of Machine Learning Research (special issue on Machine Learning in Computer Security), 7:2699–2720, 2006.

[10] Vinciarelli, A. Noisy text categorization. IEEE Transactions on Pattern Analysis and Machine Intelligence, 27(12):1882–1885, 2005.

[11] http://razor.sourceforge.net/.

[12] http://pyzor.sourceforge.net/.

[13] http://www.rhyolite.com/anti-spam/dcc/.

[14] Available at http://www.cs.cmu.edu/enron/.

[15] Klimt, B.; Yang, Y. The Enron corpus: a new dataset for e-mail classification research. Proceedings of the European Conference on Machine Learning, pp.217–226, 2004.

[16] Dredze, M.; Gevaryahu, R.; Elias-Bachrach, A. Learning fast classifiers for image spam. Proceedings of the International Conference on Email and Anti-Spam (CEAS 2007).

[17] Wang, Z.; Josephson, W.; Lv, Q.; Charikar, M.; Li, K. Filtering image spam with near-duplicate detection. Proceedings of the International Conference on Email and Anti-Spam (CEAS 2007).

[18] Biggio, B.; Fumera, G.; Pillai, I.; Roli, F. Image spam filtering by content obscuring detection. Proceedings of the International Conference on Email and Anti-Spam (CEAS 2007).

[19] Biggio, B.; Fumera, G.; Pillai, I.; Roli, F. Image spam filtering using visual information. Proceedings of the International Conference on Image Analysis and Processing (ICIAP 2007), IEEE Computer Society, pp.105–110.

[20] Biggio, B.; Fumera, G.; Pillai, I.; Roli, F. Image spam filtering using textual and visual information. Proceedings of the MIT Spam Conference 2007.

[21] McCallum, A.; Nigam, K. A comparison of event models for Naive Bayes text classification. Proceedings of the AAAI Workshop on learning for text categorization, 1998.