



Deliverable 5.3

Third Feature Extraction Report

K.U.Leuven
Version 07
July 18, 2008



IST 2006 027600

AntiPhish

Anticipatory Learning for Reliable Phishing Prevention

Specific Targeted Research or Innovation Project

2.4.3 Towards a global dependability and security framework

D 5.3 Third Feature Extraction Report

Due date of deliverable: M30 (June 30, 2008)

Actual submission date: July 21, 2008

Start date of project: 01. Jan. 2006

Duration: 36 months

Lead Contractor for this Deliverable: K.U.Leuven

Revision: 07

Project co-funded by the European Commission within the Sixth Framework Programme (2002-2006)		
Dissemination Level		
PU	Public	X
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Revision history

Deliverable administration and summary		
Project acronym: AntiPhish		ID: IST-2006-027600
Document identifier:	AntiPhish-del-D53-ThirdFeatureExtractionReport	
Leading partner: K.U.Leuven		
Report version: v07		
Report preparation date: July 18, 2008		
Classification: Public		
Nature: Report		
Author(s) and contributors: Marie-Francine Moens, Christina Lioma, Juan Carlos Gomez, Gerhard Paaß, Andre Bergholz in collaboration with all partners		
Status:		Plan
		Draft
		Working
	x	Final
		Submitted
		Approved

The AntiPhish © Consortium has addressed all comments received, making changes as necessary. Changes to this document are detailed in the change log table below.

Date	Edited by	Status	Changes made
01.07.2008	Marie-Francine Moens	Draft	Draft
01.07.2008	Christina Lioma	Draft	Draft
01.07.2008	Juan Carlos Gomez	Draft	Draft
07.07.2008	Marie-Francine Moens	Working	Integration of additional parts
14.07.2008	Andre Bergholz	Working	Image features, Evasion detection
16.07.2008	Marie-Francine Moens	Working	Small corrections
18.07.2008	Marie-Francine Moens	Final	

Notice that other documents may supersede this document. A list of latest public AntiPhish deliverables can be found at the AntiPhish webpage at www.AntiPhishResearch.org/publications.

Copyright

This report is © AntiPhish Consortium 2008. Its duplication is allowed only in the integral form for anyone's personal use for the purposes of research or education.

Citation

Marie-Francine Moens, Christina Lioma, Juan Carlos Gomez, Gerhard Paass, André Bergholz (2008). Deliverable D5.3 Third Feature Extraction Report. K.U.Leuven, AntiPhish Consortium, www.antiphishresearch.org.

Acknowledgements

The work presented in this document has been conducted in the context of the EU Framework Programme project IST 2006 027600 AntiPhish. AntiPhish is a 36-month project that started on January 1st, 2006 and is funded by the European Commission as well as by the industrial partners. Their support is appreciated.

The partners in the project are Fraunhofer Institute for Intelligent Analysis and Information Systems (FHG), Symantec LIRIC Limited (LIRIC), Symantec Ltd. (Symantec Ireland), TISCALI Services S.r.l. (Tiscali) and K.U.Leuven / CS-LIIR (K.U.Leuven). The content of this document is the result of extensive discussions within the AntiPhish© Consortium as a whole.

More information

Public AntiPhish reports and other information pertaining to the project are available through AntiPhish public web site under www.antiphishresearch.org.

Table of contents

1	Introduction	7
2	Debugging of the code	8
3	Experiments with the hidden text salting features	9
4	Topic and type features	10
5	Graphical information	11
6	Preliminary tests for tasks 5.7 and 5.8.....	12
7	Task 5.7 Multi-message features.....	13
8	Task 5.8 Extraction adaptation	14
9	Detection of new hidden salting	15
10	Conclusions.....	16
11	References	17

Executive summary

This document describes the work done with regard to the feature extraction in WP 5, which was realised at the end of the second project year and the first 6 months of the third project year. With regard to the work of K.U.Leuven, due to the leaving of the researcher, Jan De Beer, in the beginning of November 2007, and the start of Juan Carlos Gomez on March 15, 2008, some delays have occurred. Most of the work until now has been focused on studying and debugging the existing code of WP 5. The design for task 5.7 and 5.8 has been finished and the implementation has begun. With regard to IAIS work has focused around task 5.6, graphical information. Features for the detection of image distortion and the detection of logos in emails have been implemented.

Further results will be reported in a final report of WP 5.

1 Introduction

WP 5 of the AntiPhish project regards the extraction of features that are highly discriminative to distinguish phishing emails from other types of emails. The extracted features will be used in highly accurate phishing email filters.

Feature extraction reports 5.1 and 5.2 described the extraction of the more traditional email features, such as unigrams, syntactic features, structure and layout features, semantic features based on probabilistic latent semantic analysis, and salting features. An important aspect of the research is to anticipate novel features. Because phishing emails have to look very trustworthy, we expect phishing to hide content in order to mislead existing filters. In this category of hidden salting we were able to detect when a new hidden salting feature emerges in a stream of emails, by comparing an email as perceived by the user (computed by considering all known hidden salting tricks) with the OCR scan of an email. Upon divergence, we have a strong indication of an unknown hidden salting trick.

Our current work on feature extraction regards the building and exploiting of profiles of groups of emails, and the extension of the work on anticipating new features beyond hidden salting.

2 Debugging of the code

The antisalting code designed and developed by K.U.Leuven for task 5.1 (salting feature extraction) in WP 5 was tested by Fraunhofer IAIS and Symantec over a set of more than 80,000 emails from a corpus of Symantec and some bugs were found inside the code when the output results were analyzed. The most important difficulty with the code was related to non-deterministic outputs; which means that when a particular message was provided to the antisalting code several times (i.e. the same input), the results were changing from time to time (while we are expecting the same output in all runs). Even if this abnormal performance was not present in all the messages, with a percentage of 11% of non-determinism, a solution was clearly necessary.

We spent some time analyzing the code in a very detailed way, in order to review and test all the possibilities for this abnormal behaviour, which resulted in a number of modifications.

Our principal modification to the code was done by changing the HashMap data structure by the LinkedHashMap data structure in all the classes and methods of the code. The reason to implement such change in the structure was that the HashMap class does not guarantee the order of the map; in particular it does not guarantee that the order of the elements inserted in the map will remain constant over time. At the end, the way in which HashMap stores the elements was a source of non-determinism. On the other hand LinkedHashMap is an implementation of the Map interface with predictable iteration order. It means that this class maintains a doubly linked list running through all of its entries, and then this linked list defines the iteration ordering, which is normally the order in which keys were inserted into the map (insertion-order).

Preserving the insertion order in the list is important since the glyphs taken from the renderer and used to construct a glyph model by the code need to be inserted and extracted in the same sequence; otherwise when glyphs are extracted without a fixed order it is possible to obtain words or phrases that make no sense, leading to a mismatch in predicting the reading order, and since the sequence of words and phrases changed with time (according with the HashMap behaviour), reproducing the same results twice was not possible. With this solution it was possible to reduce the 11% of non-determinism to less than 1% in total (and only 0.02% in emails that contained hidden text salting).

The remaining non-determinism was due to a lack of synchronization between two threads: the one from the Swing library, which deals with the rendering of the HTML page over a graphical component, and the one from the antisalting code in charge of collecting the glyphs in the order they are painted by the renderer. Since with complex mails, i.e. ones with a lot of components to be painted, the collector spent much time to determine the order, sometimes the renderer was faster than the collector, and the collector did not have enough time to perform its task. We solved this problem by inserting a small pause in the process of the thread from the Swing library, giving enough time to the collector to perform its task.

3 Experiments with the hidden text salting features

3.1 Salting features

We have identified four main hidden text salting features. These features are defined on an individual character (or *glyph*) basis:

- Clipping: given an initial spatial confinement within which a glyph can be drawn, we test the extent to which a glyph is positioned outside the bounds of this confinement.
- Concealment: we test whether a glyph is concealed by other glyphs or shapes.
- Glyph colour: we test whether the colour of the glyph contrasts with the background colour enough for it to be visible.
- Glyph size: we test whether the glyph is large enough in size and shape so that it is visible.

We also compute the segments of an email that have a consistent reading order as perceived by the reader of an email based on document segmentation techniques. The reading order itself is defined based on language models considering layout features, word length, n-grams of characters and common words obtained from a reference corpus in a given language. If the reading order of the source text differs from the reading order perceived by the user, we have a strong indication of salting. Details on the detection of all salting tricks are described in [De Beer Moens 2007].

3.2 Evaluation of hidden text salting features for phishing email classification

We use the above salting tricks as features for classifying emails as phishing or non-phishing in a real-life corpus provided by Symantec. The corpus consists of 20,000 non-phishing and phishing emails, compiled over a period of almost seven months, from April 2007 to mid-November 2007. The corpus contains 16,364 non-phishing and 3,636 phishing emails, which is a ratio of four to five non-phishing emails for one phishing email. We use a standard Support Vector Machine (SVM) classifier with 10-fold cross validation to classify the emails. We obtain 96.46% precision, 86.26% recall, and 91.07% F-measure. The best classification feature, found in 86% of all phishing emails, is font colour. State-of-the-art phishing classification reaches F-measures of 97.6% (with random forests [Fette Sadeh Tomasic 2007]) up to 99.4% (with SVMs [Bergholz Paass Reichartz Strobel Chang 2008]) when using known discriminative features, such as URL length & longevity, HTML & Javascript information on the 2002-2003 spamassassin corpus & a public phishing corpus (these corpora are described in [Fette Sadeh Tomasic 2007]). We use only salting features. If we also combine these salting features with known discriminative features, performance may improve.

4 Topic and type features

A master thesis student, Cao Jing (K.U.Leuven, Master of Artificial Intelligence), performed a number of experiments on a set of emails with regard to: 1) the extraction of sentences in emails that contain an action verb; 2) the classification of the action verb into “potentially dangerous” and “potentially safe”; 3) the extraction of a temporal expression of a time frame (e.g., within 24 hours) in a sentence; 4) the classification of this temporal expression as “urgent” or “not “urgent” (respectively referring to short and longer time frames); 5) the identification of salutations; and 6) the classification of the salutation as “personal” and “impersonal”. Classification was done with a simple naïve Bayes classifier using annotated examples. We obtained accuracy levels of 90% and more for the above classifications by using simple unigram features.

5 Graphical information

To defeat usual text-based filtering techniques spammers and phishers started to embed the message into attached images, a trick known as image spam or image phishing. In addition, invisible text is often inserted into the body of an email. According to Commtouch image spam accounted for 30-50% of all spam in 2006.

5.1 Image distortion

Spammers started to obscure image text to defeat OCR tools by introducing small random distortion and stains, which easily can be read by humans but drastically reduce the accuracy of OCR. To detect such distortions, different features for images were developed based on low-level image processing techniques, e.g., based on color moment and color heterogeneity. For image phishing, these techniques are less relevant as phishing emails should look clean and professional as if they come from reputable senders. Therefore OCR techniques are potentially valuable for phishing emails, if a high-quality OCR solution is used.

We implemented algorithms based on CAPTCHA breaking techniques. We compute two features that are able to detect random background noise and broken or merged characters. Emails that contain these obscuring techniques have a high potential to be spam.

5.2 Logo detection

A phishing email looks much more authentic, if it contains a proper logo of the affected institution, e.g., a bank. Note that only relatively few brands are attacked, according to the Anti-Phishing Working Group 144 different brands in December 2007. The ability to robustly detect logos in an embedded image and identify the associated brand or institution is a valuable feature for phishing classification.

We implemented a novel technique for logo detection [Konya Seibert Glahn Eickeler 2008]. Compared to the standard object detection scenario, detecting and comparing logos is a relatively easy task as the logo will not be subject to major transformations, such as occlusion or change of perspective. This is due to the fact that it is the phisher's goal to use the logo in exactly the same way the bank or company would do. To detect a logo we first perform a color segmentation on the image. Then we identify all connected components in this image. The next step is to form a graph from these components (one component per node) and match this graph to the trained graphs of the original logos. This algorithm is very fast because in general a logo-graph is only composed of very few nodes.

As logo detection features we record for each email the total count of detected logos, the count of distinct logos, and for each known logo a Boolean feature indicating whether or not the logo appears in the email.

6 Preliminary tests for tasks 5.7 and 5.8

6.1 Multi-message features (task 5.7)

The goal of task 5.7 is to extract features common to groups of emails, as opposed to individual emails, in order to facilitate the association of email groups with phishing or non-phishing behaviour. This 'email group behaviour' can reveal phishing features at a macroscopic level that could have gone undetected when looking at individual emails at a microscopic level.

6.1.1 Email dissemination behaviour

The number of emails sent per month, day, and hour is recorded over a period of 12 months. It is found that, overall, non-phishing emails tend to be disseminated randomly, whereas phishing emails tend to be disseminated in bursts for certain months. For most months, there are usually two 'peaks', i.e. two 2-3 day periods during which batches of phishing emails are massively disseminated.

6.1.2 Distribution of URL links in emails

The number of URL links included in the text of an email is recorded per month, day, and hour. It is found that overall the distribution is very different for phishing and non-phishing emails. The difference is particularly pronounced for certain months, during which the dissemination of phishing emails (described in 6.1.1) is also on the rise, such as January and September.

6.1.3 Distribution of terms in emails

The distribution of terms included in the text of an email is recorded per month, day, and hour. Overall, notable differences are found in the distribution of words for phishing and non-phishing emails. Again, this difference is also correlated to the email dissemination statistics described in 6.1.1. Overall, it seems that the more verbose phishing emails are posted in the weekends.

Overall, the variation of the above features with respect to the temporal distribution of emails tends to be much lower in the group of non-phishing emails, and higher in the group of phishing emails.

6.2 Extraction adaptation features (task 5.8)

The goal is to adapt the classification of emails into phishing and non-phishing to new, dynamic features. This can help generalise our technique from known to unknown features, and also to make predictions about the evolution of phishing in general.

This work is in progress. So far, frequent item sets algorithms are being tested on our email data.

7 Task 5.7 Multi-message features

Extraction of multi-message features, a task of WP 5 of the AntiPhish project is work in progress intended to be completed soon. At this point, K.U.Leuven has already designed the general schema, defining tools to be used and methods and structures to be implemented, all of this into a framework planned in order to carry out with this task.

Analysis of individual mails is the most common way to extract features (semantic, syntactic, graphical, etc.) used for detecting and classifying malicious messages. Nevertheless, the number of messages in a database inside a mail server could be enormous, and all of this information can be exploited by algorithms with the purpose of extracting inherent knowledge contained in a collection of messages. Similar messages share some common features, and trying to find these collective elements is very useful to identify groups or categories of mails. Phishing mails share some known elements like the urgency for a click on a link, the request for information or the fact that the phishers do not know the actual names of the users. However, there exist sets of features that are still not known or identified, that are not directly observable from an individual mail and that can be potentially useful to distinguish phishing messages. Following this idea, the particular objective in this task consists of analyzing a set of mails in order to obtain groups of features that are good enough to perform a classification or discrimination between ham and phishing mails.

The general plan to proceed in solving this task is based on two strategies: the first one consists in the proper extraction of frequent patterns from a set of mails using the FP-growth algorithm, thus we can obtain an abstract representation of the whole set of data by mean of these frequent features, which in fact reflect strong associations between messages and carry underlying semantics of the data.

The second strategy consists in the reduction of the data-space dimensionality, where the features from the original messages are transformed into a less dimensional space using Principal Component Analysis (PCA), which can produce an abstract representation of the whole set by mean of a set of principal components, which represent a new space that is expressed in terms of statistic similarities and differences between messages.

The use of these approaches presents some advantages: reduction of the overall size of the input dataset, and with this less space is required for the processing, reduction of the processing time by only handling with the most representative (semantic or statistical) features from the whole set of messages and extraction of features that are highly representative of the set of mails. At the end, the resulting features of applying these methods can then be used by any classifier by themselves or in conjunction with other kind of features.

8 Task 5.8 Extraction adaptation

This task is related to task 5.1 – task 5.7 as it investigates how the feature extraction needs to be adapted according to the changes in time of new phishing techniques. At this point, K.U.Leuven has already designed the general schema, defined tools to be used, methods and structures to be implemented, and planned the framework for carrying out this task.

Phishers are continuously developing new ideas, methods and techniques that each time are more sophisticated and complex, all with the purpose of evading the automatic filters installed in mail servers and trying to reach the inbox of the users. Inclusion of new features or modifications of previous ones is an expected treat, since once a feature is identified by a filter, it could be inserted into a black list. Detecting or anticipating this new feature is of primordial importance for any filter, since they are not obvious or directly related with a semantic interpretation, rather they can be associated with other rules like probability distributions or statistical significance.

Following the previous argument, for T5.8 the objective is to detect automatically new (previously unknown) features than can help to discriminate between phishing and other emails. A completely a priori knowledge of new features is not possible, because such task implies a prediction of the future imagination of the phishers. Nevertheless, mechanisms to detect new phishing features inserted in messages can be implemented.

The general process to follow in order to deal with this task consists in periodically updating the general set of features previously detected. A posterior evidence of phishing will provide the new features to be inserted into the set. We are planning to do this by applying the FP-Growth algorithm and PCA over sets of periodically defined phishing mails. These two methods provide an abstract representation of a set of mails under different approaches, the FP-Growth algorithm provides a representation that underlies a semantic significance of the set of mails; while the PCA provides a representation expressed in terms of statistical similarities and differences between messages. Since with time new features are inserted into mails, FP-Growth and PCA can help to detect which of these features are highly representative of the defined set. Then it is possible to perform a comparison between previous known features and newly detected features, in order to update the set of features that are discriminative for the set of messages.

9 Detection of new hidden salting

We have developed a hidden salting simulation model based on cognitive theory [Bergholz Paass Reichartz, Strobel, Moens Witten 2008]. For an input email the model simulates which text will actually be seen by the user based on a fixed set of known salting tricks. In addition, the model indicates how many times each of the known salting tricks was used in the email.

Because spammers are likely to continuously develop new tricks to evade detection, the hidden salting simulation system needs continuous maintenance. We attempt to close the loop by identifying email messages that are likely to make the hidden salting simulation system fail. To this end, we try to detect differences between the simulated perceived text as generated by our hidden salting simulation system and the message text as obtained by applying OCR to the email message rendered by some rendering engine.

We propose robust text comparison metrics that allow us to compare the simulated perceived message text to the true message text obtained by applying OCR to the rendered email. Afterwards we use the distances computed with these metrics' as features to train a classifier. An outlier as detected by this classifier would be a candidate email for containing a new, unseen salting trick. We automatically determine the threshold of when to consider an email as outlier.

Our work only simulates the real-world situation and it remains open at this point how many new salting tricks it would discover in reality. However, each of the now known salting techniques was once truly unknown, and our simulation tests the system in a manner reflecting this. Our system generalizes effectively, without modification, to detect future salting tricks that are currently truly unknown. On the other hand one could in argue that we do not discover emails containing new salting tricks but emails that are not processed well by our hidden salting simulation system. However, that is of course beneficial as well. Ultimately, our work identifies emails that are in one way or another difficult to process and that help human developers to improve their email filtering systems. Given the multi-billion dollar damages resulting from unwanted emails any such improvement may result in significant economic benefits, and our work presents a systematic process for discovering possible improvements.

10 Conclusions

We reported on the finished and ongoing research on feature extraction. We would like to stress that K.U.Leuven lost some time in finding and hiring a competent researcher for the replacement of Jan De Beer and in solving bugs in the complex software for hidden salting detection. At this moment we are confident that these problems are behind us and we are progressing towards the final goals of the AntiPhish project with regard to the feature extraction tasks. We have implemented successful ways of identifying new hidden salting tricks and are proceeding steadily towards our next goals of extracting multi-message features and implementing generic ways of anticipating new phishing features.

11References

[Bergholz Paass Reichartz Strobel Chang 2008] A. Bergholz G. Paass F. Reichartz S. Strobel J.-H. Chang (2008). Improved phishing detection using model-based features. Conference on Email and Anti-Spam (CEAS), 2008.

[Bergholz Paass Reichartz, Strobel, Moens Witten 2008] A. Bergholz, G. Paass, F. Reichartz, S. Strobel, M.-F. Moens, B. Witten. Detecting Known and New Salting Tricks in Unwanted Emails. In Proceedings of the International Conference on Email and Anti-Spam (CEAS), 2008.

[De Beer Moens 2007] J. De Beer, M.-F. Moens. Detecting and Resolving Text Salting through Rendering Analysis and Cognitive Processing. Technical Report (46 p.), 2007.

[Fette Sadeh Tomasic 2007] I. Fette, N. Sadeh, A. Tomasic. Learning to detect phishing emails. International World Wide Web (WWW) Conference, 647–656, 2007.

[Konya Seibert Glahn Eickeler 2008] I. Konya, C. Seibert, S. Glahn, S. Eickeler (2008). A Robust Front Page Detection Algorithm for Large Periodical Collection. Submitted for publication.

[Lioma Moens De Beer Gomez 08] C. Lioma, M.-F. Moens, J. De Beer, J.-C.-Gomez (2008). Challenging hidden text salting in digital media. In preparation.

[Lioma Moens Gomez De Beer Bergholz Paass Horkan 08] C. Lioma, M.-F. Moens, J.-C.-Gomez, J. De Beer, A. Bergholz, G. Paass, P. Horkan (2008). Anticipating hidden text salting in emails. Recent Advances in Intrusion Detection (RAID), 2008.