Mitigation of Spear Phishing Attacks: A Content-Based Authorship Identification Framework

Mahmoud Khonji
Computer Engineering
Khalifa University
P.O. Box 573
Sharjah, UAE
Email: mkhonji@ku.ac.ae

Youssef Iraqi
Computer Engineering
Khalifa University
P.O. Box 573
Sharjah, UAE
Email: youssef.iraqi@ku.ac.ae

Andrew Jones
Information Security
Khalifa University
P.O. Box 573
Sharjah, UAE
Edith Cowan University
Email: andrew.jones@ku.ac.ae

***Abstract*—Phishing is a semantic attack that takes advantage of the naivety of the human behind electronic systems (e.g. e-banking). Educating end-users can minimize the impact of phishing attacks, however it remains relatively expensive and time consuming. Thus, many software-based solutions, such as classifiers, are being proposed by researchers. However, no software solutions have been proposed to minimize the impact of *spear* phishing attacks, which are the targeted form of phishing, and have a higher success rate than generic bulk phishing attacks. In this paper, we describe a novel framework to mitigate *spear* phishing attacks via the use of document authorship techniques — Anti-Spear phishing Content-based Authorship Identification (ASCAI). ASCAI informs the user of possible mismatches between the writing styles of a received email body and of trusted authors by studying the email body itself (i.e. the *writeprint*), as opposed to traditional user ID-based authentication techniques which can be spoofed or abused. As a proof of concept, we implemented the proposed framework using Source Code Author Profiles (SCAP), and the evaluation results are presented.**

***Index Terms*—spear phishing; usable security; e-mail mining; stylometrics;**

I. Introduction and Motivation

Phishing is defined as a form of crime that unlawfully, and through *Social Engineering*, obtains data from victims for the attacker's benefit, over an electronic communication channel. Through social engineering, the victim is persuaded to perform certain actions, such as submitting personal information directly to the phisher, or executing malware (e.g. Zeus and SpyEye) that would in turn perform actions for the attacker's benefit without the victims knowledge.

At its core, phishing is a social engineering attack that exploits the human using a given system (e.g. banking, social networking. . . etc), and therefore user awareness training programs would aid the fight against phishing attacks. However, user awareness training programs, alone, are not sufficient due to a number of reasons, such as: expense, user cognition limits [1], knowledge retention [2].

Phishing attacks can be grouped into multiple categories based on the media used to deliver the phishing content, such as: SMSishing for SMS, Vishing for phishing over VoIP . . . etc. However, since the attack targets the human via social engineering, phishing is also categorized based on the approach used to persuade the victim — Bulk phishing, and spear phishing attacks.

Although many fall victims to bulk phishing attacks, detecting the bulk phishing attacks is far easier than the spear phishing attacks. This is due to the generic nature of bulk phishing attacks as the targeted base is massive, and thus generic emails are sent. Many users receive bulk phishing attacks for services that they do not use (e.g. a PayPal phishing email when the user does not own a PayPal account), which makes its detection easier. Moreover, several software classifiers have been proposed to automatically detect bulk phishing attacks.

On the other hand, *spear* phishing attacks are targeted (i.e not generic as in the bulk phish), and thus have higher success rates. Spear phishing attacks do not *only* succeed in fooling naive end-users, but also technically-aware end-users. Recently, a number of *spear* phishing attacks against Email Service Providers (ESPs) were reported, where the targeted person was a known employee of an ESP. The content of the spear phishing message did not ask for passwords directly, but rather followed a personal approach where the phisher claimed to be an old colleague with personal URL links, however the provided links contained malware to infect the machine of the targeted ESP employees, which did infect a number of ESP employee machines [3].

User ID-based authentication is unhelpful in this scenario for a number of reasons:

- Humans do not read text sequentially, but in parallel; for example, many people find it difficult to identify the difference between "Mi**cr**osoft" and "Mi**rc**osoft" (note that *r* and *c* characters are swapped), so support@microsoft.com and support@mircosoft.com are technically two different IDs while the human mind tends to recognize them as the same. Typo-squatting and cousin-naming are two types of attacks that take advantage of this weakness, which eventually results into limiting the usability of end-to-end (e.g. S/MIME, OpenPGP) and inter-domain (.e.g DKIM, SPF) authentication mechanisms.
- Many users still choose weak passwords. As studied in [4], 23% of the passwords were brute-forced within only

978-1-908320-00-1/11/$26.00 ©2011 IEEE

30 minutes.

- Passwords can be stolen by tools, such as keyloggers. Once stolen, the identity of the person can be claimed.

The primary objective of this paper is to describe a framework, by which the identities of message senders are evaluated by analyzing the message's body, without relying on senders user IDs. This can minimize the gap that is left by traditional user ID-based authentication mechanisms. The proposed framework does not conflict with user ID-based authentication techniques but rather complements them to further enhance security in a usable manner.

This paper is structured as follows: related work in Section II, framework design in Section III, implementation challenges in Section IV, implementation details in Section V, performance evaluation approach in Section VI, performance evaluation results in Section VII, implementation issues in Section VIII, and then the conclusion is drawn in Section IX

II. Related Work

Use of data mining techniques to correlate authors with their writings, such as e-mail messages and software source codes, is not new. Use of data-mining forensic techniques in e-mail and social networking services is broad, which incorporates the identification of the true origins of Spam, phishing attacks or abusers, such as [5], [6]. However, their use was targeted for forensics purposes rather than actively protecting identities of trusted senders.

Anti-phishing software classifiers in the literature primarily target bulk phishing messages, such as [7], [8], and the only spear-phishing mitigation techniques in the literature are end-user training or education approaches, such as [9], [10]. To the best of our knowledge, no software technique was proposed to mitigate spear phishing attacks.

In this paper, we continue from there by providing a software framework for e-mail mining techniques to construct a white-list of trusted e-mails sender stylometric profiles, which are then used to detect identity impersonation attempts for daily email use. In other words, our framework could be looked at as a white-list approach to keep the good guys safe, while previous email authorship contributions were more of a black-list approach to identify the bad guys (e.g. spam/phishing profiling).

III. Framework Design

To simplify the process, the following steps describe the proposed framework from a high-level perspective:

- Alice is a client for organization B, which is owned by Bob, and they do communicate via an electronic communication channel, such as e-mail.
- This communication of messages can be fed into ASCAI, where it learns and constructs a profile for Bob as well as other regular senders. The purpose of this profile is to let Alice know when a future message from Bob's user ID (or any visually similar user ID due to typo-squatting or cousin-naming attacks) is really written by Bob.
- Eve is an adversary that aims to gain access to confidential data owned by Alice, and since Eve knows that Alice is a client of Bob, she launches a *spear* phishing attack against Alice. Eve might use an e-mail sender ID that is *technically* different than Bob's, but visually appearing similar to Alice's eyes (e.g. support@mi**rc**osoft.com looks very similar to support@mi**cr**osoft.com). Eve might also use an e-mail sender ID that that is identical to that of Bob's in case no end-to-end (e.g. OpenPGP) or inter-domain (e.g. DKIM) authentication techniques were deployed.
- ASCAI calculates a writeprint for the newly arrived message (i.e. Eve's) and then guesses the actual author by utilizing the pre-computed authors writeprint profiles (which includes Bob's writeprint profile), and the writeprint profile of Eve's message.
- Because the message was not written by Bob, the writeprint profiles would have a high chance to conflict against each other. ASCAI can then be able to yield a warning that the message is not authored by the claimed identity (i.e. Bob's), or suggest that the message was written by Eve (depending on the mode of operation of ASCAI which are detailed in following paragraphs).
- If Alice and Eve did communicate previously, ASCAI might be able to inform Alice that the message was actually written by Eve. Thus, Alice not only knows that the message is not from Bob, but she also knows the real sender (Eve in this case).

To facilitate the above scenario, the required components of the ASCAI framework are (depicted in Figure 1):

- A mechanism to extract the *claimed identities* in unclassified messages. This can be extracted from the *From:* RFC822 header field or the signature, which might involve use of heuristics, Natural Language Processing (NLP) techniques such as Named Entity Recognition (NER), or metaphone algorithms to detect similarity between words.
- A mechanism to construct writeprint profiles of sending authors, which implies mapping author writeprints and their actual *author identities*.
- A mechanism to compare the similarity between *claimed identities* and *predicted identities* of authored messages.
- User interface (UI) API which facilitates presenting the gathered information or warnings to the end user.

Two modes to calculate the similarity between the *claimed identities* and the *predicted author identities* are proposed in this paper:

- Passive — as shown in Figure 2, this mode passively presents both of the identities (the *claimed identities* and the *predicted identities*) of authored messages, and assumes that the user gives attention to the presented information. This mode can take advantage of end-user's ability in understanding the semantics of natural languages, which ultimately can enhance the accuracy of the extraction of the semantically *claimed identity*.

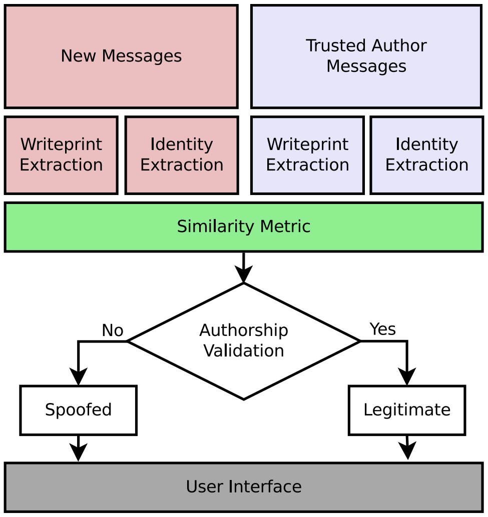


Fig. 1. ASCAI Framework.

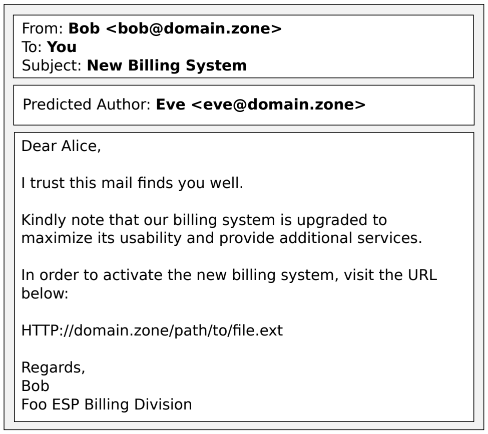
However, this mode requires educated or technically aware end users that pay attention to various UI elements. Although this mode does not warn the end-user when a spoofing attempt is detected, it provides information that makes the detection of typo-squatting, cousin-naming or stolen identity attacks detectable in an easier manner. Historically, technical employees at service providers did fall victims of spear phishing attacks that involved typo-squatting, cousin-naming or stolen identities [3], and thus this mode can be helpful.

- Active — as shown in Figure 3, this mode actively blocks the email message's content in case a mismatch is found between the *claimed identity* and the *predicted identity* of an authored message. It is technically possible to offload the similarity comparison between both of the identities to the software by using similarity algorithms that utilize heuristics, Named Entity Recognition (NER) or metaphone algorithms. This mode can be advantageous as it does not require the end-user to give attention to various informative UI elements. However, it does increase the implementation complexity and the software might not be as accurate as an aware end-user.

IV. Implementation Challenges

A number of implementation challenges exist:

- Extracting writeprints from written messages — writeprints are subject to modifications due to the nature of electronic communication channels. Finding an effective method to extract writeprints is a challenge.
- Predicting the proclaimed sender identity of a message — the proclaimed identity in this case is not limited to the user ID, but includes any *semantically* claimed identity. This imposes a challenge as computers find it difficult to extract semantics of messages. However, since most email messages present the sender identities in similar positions, such as *From:* field, or signature (which is often positioned at the bottom of most emails), then using heuristics to extract the proclaimed sender identities is possible. Heuristics can narrow down search spaces to extract certain portions of the message, which can then be processed by a number of techniques, such as Named Entity Recognition (NER) techniques or metaphone algorithms.
- E-mail delegations — in many cases, it can be desirable to delegate email sending permissions to multiple employees. In such scenario, an email address can be used to send messages that are written by multiple authors at different times, which can lead to false positives. Possible solutions can be the ability to map multiple *author writeprint profiles* to a single email identity, or the ability to activate ASCAI on specific email IDs that are likely to be authored by a single user.



From: **Bob <bob@domain.zone>**
To: **You**
Subject: **New Billing System**

Predicted Author: **Eve <eve@domain.zone>**

Dear Alice,

I trust this mail finds you well.

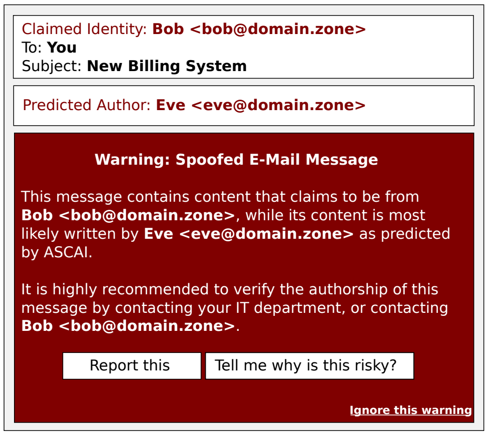
Kindly note that our billing system is upgraded to
maximize its usability and provide additional services.

In order to activate the new billing system, visit the URL
below:

HTTP://domain.zone/path/to/file.ext

Regards,
Bob
Foo ESP Billing Division

Fig. 2. ASCAI's passive mode presenting predicted author's name through the UI.



Claimed Identity: **Bob <bob@domain.zone>**
To: **You**
Subject: **New Billing System**

Predicted Author: **Eve <eve@domain.zone>**

Warning: Spoofed E-Mail Message

This message contains content that claims to be from
Bob <bob@domain.zone>, while its content is most
likely written by **Eve <eve@domain.zone>** as predicted
by ASCAI.

It is highly recommended to verify the authorship of this
message by contacting your IT department, or contacting
Bob <bob@domain.zone>.

Fig. 3. ASCAI's active mode blocking content-data due to a mismatch that is heuristically found between the *claimed identities* and the *predicted author identities*, and presenting an active warning through the UI.

V. Implementation Details

This section describes a preliminary implementation of the ASCAI framework which is then evaluated in Section VI.

This implementation follows the *passive* mode as depicted in Figure 2, and uses the SCAP [11] methodology to construct writeprint profiles for trusted authors, and for unclassified messages.

It is important to note that the ASCAI framework does not mandate the use of any specific writeprint extraction method, and an alternative writeprint extraction method can be used. Our use of SCAP is due to its high classification accuracy in application source code authorship while being agnostic to the processed language. SCAP achieved 100% of classification accuracy on most datasets [11]. While it is true that SCAP is originally proposed for source code authorship, its use of byte-level n-grams makes it language agnostic and thus can make it a promising tool for natural languages too, which is the motive behind the evaluation in Section VI.

The SCAP methodology constructs author profiles by concatenating all of their emails into a single file, and then extracting a table of L most frequent n-grams.

n-grams are essentially series of n tokens, which could be bytes, characters, words and so on. According to SCAP methodology, byte-level n-grams were found to be effective to construct profiles of programme source code authors by studying their source codes. Similar to [11], we used Perl's *Text::Ngrams* module to extract L most frequent n-grams.

The similarity metric that is followed by SCAP to map unclassified document's writeprint profiles with author writeprint profiles is the size of intersection between email writeprint profiles and author writeprint profiles, which is also what we have followed in this study. A message is assumed to be written by the author with largest amount of profile intersection. See Equation (1).

$$Similarity(P_e, P_a) = |P_e \cap P_a| \tag{1}$$

where P_e and P_a are email and author profiles respectively, and $|P_e \cap P_a|$ is the size of the intersection between P_e and P_a.

For evaluation purposes, another similarity metric is also implemented, namely Jaccard's [12] similarity index, as calculated by Equation (2).

$$Jaccard(P_e, P_a) = \frac{|P_e \cap P_a|}{|P_e \cup P_a|} \tag{2}$$

where $|P_e \cup P_a|$ is the size of the union between P_e and P_a.

Another element that defines the *Writeprint Extraction* module's performance, is the n-grams frequency ranking. In reality, there are often many n-grams that share similar frequencies. We have used three n-grams frequency ranking methods, which we will refer to as *greedy*, *non-greedy* and *strict* frequency ranking methods:

- Non-greedy ranking — returns a list of the L most frequent n-grams. However, in case of a tie where other n-grams that share the same frequency as the last n-gram in the returned list, all n-grams with the same or smaller frequency will be excluded. This often results in a profile size that is smaller than L.
- Greedy ranking — returns a list of the L most frequent n-grams. However, in case of a tie where other n-grams that share the same frequency as the last n-gram in the returned list, all n-grams with the same frequency will be also included. This often results in a profile size that is larger than L.
- Strict ranking — returns a list of the L most frequent n-grams. However, in case of a tie where other n-grams that share the same frequency as the last n-gram in the returned list, a number of the other n-grams will be included *randomly* to make up exactly a profile with an equal size to L. However, we have excluded this from the evaluation in Section VI as it did perform very poorly during our preliminary tests.

This experimental implementation does not implement the *Identity Extraction* module as depicted in Figure 1 since the implementation follows the *passive* mode, which expects the end-user to extract and compare the identities.

VI. Performance Evaluation Approach

A. Dataset

The evaluation dataset is composed of email messages extracted from authors' email Inbox between Feb 2010 and May 2011, which is composed of:

- 289 emails.
- 12 authors.

The extracted text in the dataset is noisy as it includes the full e-mail body, including forwarded text written by other authors and their signatures. The only performed preprocessing is done to remove HTML tags, JavaScript and CSS from the text.

All of the emails in the dataset are legitimate emails, which is sufficient for evaluating the writeprint extraction module since it classifies based on author writing styles (writeprints).

B. Evaluation Metrics

Similar to [11], the Accuracy rate is used to measure the performance of the system. See Equation (3).

$$ACC = \frac{N_c}{N} \tag{3}$$

Where N_c is total number of correctly classified testing emails (i.e. emails that their authorship is correctly identified), and N is total number of testing emails.

The SCAP methodology is evaluated against the dataset using 10-fold cross-validation. The 10-fold cross-validation essentially splits the dataset equally into 10 folds, and then trains the classifier with 9 folds while using the remaining 1 fold for testing. This process is repeated for 10 rounds where a different fold is chosen for testing in each round. The evaluated accuracy in each round is then measured, and the average accuracy of all of the rounds is considered as the overall classification accuracy of the *Writeprint Extraction* module. This approach makes sure that every email in the dataset is being trained with and tested against.

TABLE I

SCAP EVALUATION RESULTS ON E-MAILS WITH VARYING L AND n VALUES, USING NON-GREEDY n-GRAM RANKING, AND PROFILE INTERSECTION SIMILARITY METRIC.

L	**n-gram**						
	5	6	7	8	9	10	11
125	56%	52%	53%	47%	46%	48%	44%
250	61%	54%	52%	51%	48%	49%	50%
500	78%	76%	74%	73%	72%	68%	67%
1000	79%	78%	78%	80%	80%	79%	79%
1500	82%	82%	**83%**	**83%**	82%	82%	82%
2000	81%	82%	82%	81%	80%	80%	79%
2500	78%	81%	80%	81%	81%	80%	80%
3000	77%	80%	79%	80%	80%	81%	81%

TABLE II

SCAP EVALUATION RESULTS ON E-MAILS WITH VARYING L AND n VALUES, USING GREEDY n-GRAM RANKING, AND PROFILE INTERSECTION SIMILARITY METRIC.

L	**n-gram**						
	5	6	7	8	9	10	11
125	44%	46%	46%	55%	53%	51%	54%
250	75%	78%	75%	78%	79%	81%	84%
500	79%	80%	82%	82%	81%	79%	80%
1000	79%	80%	83%	86%	**87%**	**87%**	86%
1500	82%	83%	82%	83%	83%	83%	82%
2000	81%	81%	82%	81%	79%	80%	80%
2500	76%	75%	80%	81%	81%	81%	81%
3000	75%	79%	79%	81%	80%	80%	79%

Since the implementation follows the *passive* mode of ASCAI, the evaluation assumes that an ideal *Identity Extraction* module is in place, which is achieved by an aware end-user.

VII. Performance Evaluation Results

This evaluation is repeated for a number of times to evaluate the *Writeprint Extraction* module with:

- Two n-gram ranking methods, namely: *non-greedy* and *greedy* ranking methods. The *strict* ranking method was excluded as it performed very poorly during our preliminary tests.
- Two similarity metrics, namely: profile intersection size (Equation (1)) and Jaccard's similarity metric (Equation (2)).

A. n-gram Ranking Methods

To evaluate the ranking methods, we fixed the similarity metric to the one proposed in SCAP [11], which is profile intersection size (Equation (1)). Tables I and II present the accuracy of the *Writeprint Extraction* module as measured by Equation (3).

As presented in Table I, the non-greedy n-gram ranking method achieved a maximum of 83% of accuracy when $L = 1500$ and both $n = 7$ and 8. However, as presented in Table II, the greedy n-gram ranking method achieved a noticeably higher accuracy rate of 87% when $L = 1000$ and $n = 9$ or 10.

TABLE III

SCAP EVALUATION RESULTS ON E-MAILS WITH VARYING L AND n VALUES, USING GREEDY n-GRAM RANKING, AND JACCARD'S [12] SIMILARITY METRIC.

L	**n-gram**						
	5	6	7	8	9	10	11
125	60%	57%	53%	63%	65%	65%	65%
250	75%	78%	78%	82%	**84%**	**84%**	83%
500	79%	78%	81%	80%	80%	82%	82%
1000	76%	77%	80%	81%	79%	78%	77%
1500	69%	70%	67%	65%	66%	63%	62%
2000	62%	60%	61%	63%	64%	63%	64%
2500	50%	52%	54%	56%	56%	57%	57%
3000	49%	51%	52%	54%	52%	48%	47%

The numbers $L = 1000$ and $n = 9$ or 10 mean that the 1000 most frequent 9 or 10-byte sequences in emails serve as the most distinctive writeprints to correlate authors with their writings. The reasons behind achieving best results with $L = 1000$ and $n = 9$ or 10 is highly dependent on the dataset we evaluated against. The dataset itself is also affected by the syntax and grammar rules of the English language, as these rules affect the distribution of characters which in turn affects the distribution of the n-grams. The values $L = 1000$ and $n = 9$ or 10 are not universal, but depend on a number of other factors such as the natural language used to write the messages. In other words, different L and n values might be needed to process messages written in other languages as the language influences the distribution of the n-grams accordingly.

B. Similarity Metrics

This subsection evaluates the performance of the *Writeprint Extraction* module when a different similarity metric is used, namely: Jaccard's [12] similarity (Equation (2)) instead of the profile intersection size (Equation (1)). In this section, we also use the greedy n-gram ranking method as it achieved a higher accuracy rate than the non-greedy ranking method in Section VII-A.

As presented in Table III, the *Writeprint Extraction* module, with the Jaccard similarity metric achieved 84% of prediction accuracy for $L = 250$ and $n = 9$ and 10, which is noticeably less accurate than SCAP's native similarity metric (profile intersection).

Although the performance is not as high as 87%, the profile size is reduced. This reduces the space requirements to store the profiles, and in our implementation resulted in higher classification speed as well.

For example, and according to our experimental implementation, the total time taken for testing instances in a 10-fold cross-validation setup is 61.935 seconds when $n = 11$ and 51.935 seconds when $n = 9$. That is 16% reduction in testing time, for 3% reduction in classification accuracy.

The reduction in overall time delay might not be significant if ASCAI is implemented in an email server, however it can be beneficial if ASCAI is implemented on an email client as the delay may affect end-user's experience.

VIII. Implementation Issues

The implementation in this study has the following known limitations:

- Although a classification accuracy of 87% can be considered acceptable compared to the simplicity of the implementation and the noisy dataset, a higher classification accuracy is desirable to reduce false positives.
- The current implementation used SCAP for writeprint extraction and mapping to author IDs. If no message was previously communicated with the actual author, SCAP will map it to a closest author profile match (which is due to the different objectives of SCAP than that of this paper). To avoid such scenario, modified SCAP implementations or different mapping mechanisms could be used instead, which ASCAI is not specific about. Alternatively, the predicted identity could be disabled and to be merely used internally to detect mismatches in the *active* mode (i.e. to let the software warn the user if the identities do not match, without giving details of the actual author).
- If previous messages were communicated with an attacker (e.g. Eve), then (as evaluated) SCAP will have a 87% chance to correctly detect the true identity of the attacker. However, if no previous message is communicated previously with the attacker, then SCAP will map to any closest match which includes the semantically claimed identity (resulting in a False Negative). The random chance for such event reduces when more trusted authors profiles exist. For example, for 12 authors, there is only 8% random chance that an unknown attacker's profile might get mapped to the claimed identity's profile.

IX. Conclusion

In this study, the design of Content-Based Authorship Identification (ASCAI) framework is presented, which (to the best of our knowledge) is the first framework to enable content-based authorship techniques for daily email use to mitigate spear phishing attacks.

ASCAI aims at enhancing security usability by protecting trusted authors' identities from being proclaimed by other senders via typo-squatting, cousin-naming or identity theft attacks, which are common problems with spear phishing attacks. The approach that ASCAI follows to protect trusted authors is by studying the email body itself, as opposed to conventional user ID-based methods which have many weaknesses.

As a preliminary implementation, we evaluated an experimental implementation of the *Writeprint Extraction* module via the use of an effective source-code authorship technique, namely SCAP, which resulted in a maximum accuracy 87% of authorship prediction accuracy of email messages by using SCAP's native profile similarity metric (profile intersection size). The SCAP methodology proved to be simple and effective for natural languages despite the fact that it is originally designed for software source codes authorship.

As a future work:

- Explore improvements to the writeprint extraction method to further enhance classification accuracy. Evaluating other authorship techniques, aside SCAP, may prove promising to address the implementation issues and increase the effectiveness of ASCAI.
- Use heuristics, Named Entity Recognition (NER) or metaphone algorithms to implement and evaluate ASCAI's *active* mode.
- Enhance the pre-processing stage to reduce dataset noise by removing forwarded text and signatures. Removing noise from the dataset can increase the prediction accuracy of the authorship technique.
- Study the affects of possible adversarial responses against authorship techniques, such as *imitation* and *obfuscation* attacks.
- Explore the effectiveness of the addition of a model to facilitate a collaborative construction of trusted author profiles.

Acknowledgment

Thanks to Buhooth[1] for funding this work. The authors would also like to thank Layton, R.[2] and Frantzeskou, G.[3] for sharing valuable details of their n-gram implementations.

References

[1] S. Gorling, "The Myth of User Education," *Proceedings of the 16th Virus Bulletin International Conference, 2006.*

[2] A. Alnajim and M. Munro, "An evaluation of users' anti-phishing knowledge retention," in *Information Management and Engineering, 2009. ICIME '09. International Conference on*, April 2009, pp. 210 –214.

[3] B. Krebs, "Spear phishing attacks snag e-mail marketers," http://krebsonsecurity.com/2010/11/spear-phishing-attacks-snag-e-mail-marketers/, accessed May 2011.

[4] B. Schneier, "Real-world passwords," http://www.schneier.com/blog/archives/2006/12/realworld_passw.html, accessed May 2011.

[5] O. de Vel, A. Anderson, M. Corney, and G. Mohay, "Mining e-mail content for author identification forensics," *SIGMOD RECORD*, vol. 30, pp. 55–64, 2001.

[6] R. Layton, P. Watters, and R. Dazeley, "Automatically determining phishing campaigns using the uscap methodology," in *eCrime Researchers Summit (eCrime), 2010*, 2010, pp. 1 –8.

[7] I. Fette, N. Sadeh, and A. Tomasic, "Learning to detect phishing emails," in *Proceedings of the 16th international conference on World Wide Web*, ser. WWW '07. New York, NY, USA: ACM, 2007, pp. 649–656.

[8] A. Bergholz, J. De Beer, S. Glahn, M.-F. Moens, G. Paaß, and S. Strobel, "New filtering approaches for phishing email," *J. Comput. Secur.*, vol. 18, pp. 7–35, January 2010.

[9] S. Sheng, B. Magnien, P. Kumaraguru, A. Acquisti, L. F. Cranor, J. Hong, and E. Nunge, "Anti-phishing phil: the design and evaluation of a game that teaches people not to fall for phish," in *Proceedings of the 3rd symposium on Usable privacy and security*, ser. SOUPS '07. New York, NY, USA: ACM, 2007, pp. 88–99.

[10] P. Kumaraguru, Y. Rhee, A. Acquisti, L. F. Cranor, J. Hong, and E. Nunge, "Protecting people from phishing: the design and evaluation of an embedded training email system," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, ser. CHI '07. New York, NY, USA: ACM, 2007, pp. 905–914.

[11] G. Frantzeskou, E. Stamatatos, S. Gritzalis, and S. Katsikas, "Effective identification of source code authors using byte-level information," in *Proceedings of the 28th international conference on Software engineering*, ser. ICSE '06. New York, NY, USA: ACM, 2006, pp. 893–896.

[12] P. Jaccard, "Étude comparative de la distribution florale dans une portion des alpes et des jura," *Bulletin de la Socit Vaudoise des Sciences Naturelles*, vol. 37, pp. 547–579, 1901.

[1] http://www.buhooth.ae/

[2] r.layton@icsl.com.au

[3] gfran@aegean.gr