Breaking and Fixing Content-Based Filtering

Mayank Dhiman
Stealth Security, Inc.
mayank@stealthsec.com

Markus Jakobsson
ZapFraud, Inc.
markus@zapfraud-inc.com

Ting-Fang Yen
DataVisor, Inc.
tingfang.yen@datavisor.com

***Abstract*—We demonstrate a vulnerability in existing content-based message filtering methods, showing how an attacker can use a simple obfuscator to modify any message to a homograph version of the same message, thereby avoiding digest and signature based detection methods. We measure the success of this potential attack against Hotmail, Gmail and Yahoo mail. While the attack is bothersome both in terms of its simplicity and its success, it is also easily countered. We describe some computationally practical countermeasures.**

I. Introduction

The first commercial email spam message was sent in 1994 by Canter and Siegel, two Arizona lawyers looking for clients interested in obtaining a U.S. Green Card. Directed to several thousand newsgroups, the message immediately angered so many that the ISP of the two attorneys terminated the connection of the offenders within days.

By the end of the nineties, spam had become mainstream, and with that, it had become impossible to base the defense against it simply on the termination of online services. Instead, the service providers started to scan the content, first looking for digests associated with previous spam messages, and later (with the resulting introduction of spam poison), for signatures – such as mention of keywords associated with unwanted email. A good example of this comes from pharma spam, whose commercial success in the mid-2000s caused its rapid rise in volume to over three quarters of all spam. As service providers started blocking large quantities of messages containing the word "Viagra", scammers quickly countered by morphing the keyword into "V ! @ g.r A" and similar mutations. It did not matter to the spammers – whose goal was simply to sell products – that the recipients of their emails understood that the messages were spam. However, the same could not be said for *scammers*, whose goals involve tricking recipients into believing that the messages are authentic, and not sent in bulk. This made it clearly undesirable for scammers to use morphed keywords that may draw attention from the intended victims.

Today, detection of unwanted emails is based on a combination of tools, including *volume* (how many messages originated from a sender); *reputation* (what recipients commonly do to the messages – e.g., open the message, place it in the spam folder, etc.); and *content signatures* (sequence of words associated with unwanted messages, but unlikely to be found in legitimate messages). Among unwanted messages, maybe the most difficult type to detect and block is scam messages. This is because they are typically sent in small batches, rendering both volume and reputation techniques largely meaningless and causing an increased reliance on signatures. In a sense, this places message content at the center of the battle between scammers and service providers.

We believe that the next natural move by the scammers is to use homograph attacks, in which unwanted messages are represented using a collection of encodings that are visually identical in the eyes of the recipients. In fact, by analyzing the famous "Podesta" email[1], made available by WikiLeaks, we can determine that this was actually done in that email. More particularly, for every occurrence of the word "password", the "o" was replaced by a Cyrillic letter with the very same appearance as an "o"; the same was done to the "o"s in the word "Someone" used in the assertion "Someone has your password". The only rational reason for doing this is to evade word-based and signature-based detection methods.

It is not difficult for attackers to find suitable replacement letters. In fact, for each Latin letter, there is a *large* number of "confusable" characters – the exact number depending on the fonts used and the exactness of match desired by the sender. We demonstrate the success of this obfuscation technique by showing that 90 common email scam messages that were all blocked in their "raw form" were almost without exception delivered after being obfuscated – whether to Google, Yahoo or Hotmail accounts. This obfuscation, which is not detectable to the human eye, is straightforward to automate – we wrote a simple obfuscator that performs replacements in an input message, creating identical-looking obfuscated output messages – each of which is unique from the others and from the original.

It should be noted that the use of a randomized compilation of a scam message, performed on a per-transmission basis, allows an attacker to send very large volumes of *visually identical* messages, all while circumventing digest-based volume detection used in spam filters. This would be the first polymorphic spam/scam message.

However, we do not believe that this seemingly devastating attack will result in a necessary victory for those wishing to bypass content-based filters. In a previous study of homograph attacks [1], it was suggested to perform a "reverse mapping" from non-Latin characters looking like Latin characters and to the corresponding Latin characters. We suggest combining this with a few simple heuristics, detailed in Section IV.

978-1-5386-2701-3/17/$31.00

[1] See https://wikileaks.org/podesta-emails/emailid/34899

II. Related Work

The abuse of Unicode characters to create visually similar (yet distinctly encoded) text has been widely studied in the context of domain name spoofing [2], [3]. A malicious party can register a domain name that is identical to an existing domain name, except that a subset of the characters is replaced by their indistinguishable Unicode counterpart – for example, using the small Cyrillic letter "р" to replace the lower-case Latin "p" These "confusable" characters can be effectively used in domain phishing attacks, where the attacker spoofs legitimate domain names (such as a banking website) to trick users into logging in and giving out their credentials or personal information. In addition to domain names, non-Latin international characters are also allowed in email addresses [4]. This can allow a malicious party to masquerade as another user – such as in the increasingly popular Business Email Compromise (BEC) scams [5].

In this work, we study an attack that uses these Unicode "confusable" characters in scam messages, rather than domain names or email addresses. As one of the most difficult type of unwanted messages to detect and block, scam is different from spam in that they are typically sent out in small batches, often crafted specifically for the recipients. This requires not only the content of the messages to be believable, but also the appearance and formatting of the message to look "correct." Scammers hence do not have as much freedom as spammers in manipulating their messages to evade spam filters, e.g., replacing the word "Viagra" with "V ! @ g.r A". However, as we will show, Unicode confusable characters can be used greatly to a scammer's advantage to achieve the same goals.

Most closely related to our work is a study on Unicode-obfuscated spam by Liu and Stamm [1]. The authors replaced English characters in spam message at random with similar Unicode characters, and measured the effectiveness of the SpamAssassin spam filter at detecting them. However, the characters that are replaced this way can be easily identifiable by an end user, mainly due to differences in stroke size and the height and length of the characters (and especially if they are placed side-to-side with normal English characters). While useful for spam, such discrepancies would greatly reduce the efficacy of scam messages in which the goal is to appear authentic. As we will describe in Section III-A, one of the contributions of our work is in identifying a set of Unicode characters that are visually indistinguishable from their Latin counterparts. This is important both in the context of scams (such as 419 scams and Business Email Compromise scams) and for phishing attacks – in other words, where an attacker wants to masquerade a trusted party.

In light of security problems that arise from mixing homoglyphs from different language encodings, the Unicode Consortium has published a technical standard to describe methods for detecting Unicode abuse [6]. The standard defines "restriction-levels," which limit the use of arbitrary Unicode characters in texts. The restrictions range from ASCII-only, allowing only characters from the same script, to allowing only a specific combinations of scripts to be used in the same text. Restricting the usage of Unicode characters only serves to minimize the problem, but does not completely solve it. Other works [1] have proposed performing a "reverse-mapping" of Unicode characters to ASCII characters that are visually similar, after which traditional signature or content-based methods can be applied to detect unwanted text. We describe ways to augment this approach in Section IV.

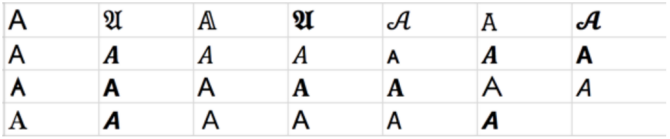

Figure 1. Confusable characters for the Latin upper-case A.

III. Methodology

We demonstrate how Unicode confusable characters can be used to obfuscate scam messages to bypass existing email filters, while preserving the readability of the messages. More specifically, we show how an attacker can produce two identical-looking messages M_1 and M_2, such that M_1 is blocked by email filters and M_2 is let through. (In fact, we show that a *tremendous number* of unique but identical-looking messages $M_2 \ldots M_n$ can be produced from M_1.)

In this section, we first describe our algorithm for generating high-fidelity scam messages using Unicode confusable characters, and then detail our experiment to measure the effectiveness of the obfuscated messages against state-of-the art email filters deployed at popular email providers.

A. Scam Obfuscation using Confusables

The Unicode Consortium made available a list of visually similar confusable characters [6]. However, characters that are similar may still be easily distinguishable, due to differences in stroke size and the height and length of the characters. Figure 1 shows an example of the confusable characters for the Latin upper-case A.

Leveraging the list of confusable characters provided by the Unicode Consortium, we further performed a multi-step manual vetting process to derive a set of high-fidelity confusable characters suitable for use in scam messages. We focus on English letters (both lower and upper case) for this study.

1) **Examining confusable characters:** For each character, we first removed from consideration all its associated confusable characters that appear widely different and easily distinguishable to the human eye. For example, most of the characters shown in the first row in Figure 1 will be removed in this step.
2) **Examining confusable characters in words:** Our second vetting stage consisted of viewing the remaining confusable characters with other words. This can reveal subtle spacing and stroke differences in the character that is not visible while viewing it in isolation. Figure 2

AFFIRM	AFFIRM	AFF I RM

Figure 2. Confusable characters for the Latin upper-case I inside a word context. The right-most confusable character is (by itself) identical to the one on the left, but put in context with other characters, it clearly stands out.

shows an example for the Latin upper-case I. The right-most confusable character is (by itself) identical to the left-most one, but put in context with other characters, it clearly stands out and should be filtered.

3) **Examining confusable characters in text:** Differences in the fonts used on mail readers and devices can render characters in slightly different ways. In a final vetting stage, we placed the remaining confusable characters inside a sentence, and evaluated this on different platforms. We asked friends and family to give us a sentence, and returned to them the same sentence but with all characters replaced with their corresponding confusables. Then we asked if something was off. This was used to discard yet some more confusable characters.

This process left us with a small set of 67 high-fidelity confusable characters, shown in Figure 3, that are visually indistinguishable to their Latin counterparts.

With these high-fidelity confusable characters, our scam message obfuscator takes in as input a block of text and outputs an obfuscated version where the characters are replaced with one of their corresponding high-fidelity confusable characters. If no suitable confusable is available for that character, the original character is preserved. The resulting message appears identical to the original to the human eye, but is in fact encoded very differently.

Based on the distribution of characters in the English language, one can compute the per-character entropy increase resulting from performing a compilation in which a random character (including the correct character) is selected, uniformly at random, for each letter in a word. For example, Figure 3 shows that "P" has two confusables and "z" has one; therefore, each time an "P" is detected, there are three candidate encodings while every time a z is detected, there are two. In the former case, the entropy increase would be $log_2(3) \approx 1.58$, whereas in the latter case, the entropy increase is exactly 1. Weighing these by the commonality of the character (while, for simplicity) assuming that all characters are used in their lower-case forms only, we get an average entropy increase of approximately 0.73 bits per character, or more than 2000 bits for a typical scam email (whose length, on average, is close to 3,000 characters). This shows that while it may be possible to enumerate all homograph versions of unique keywords, it is not practically meaningful to do this for snippets of text, and certainly not for entire emails.

B. Targeting Email Spam Filters

Commercial spam filters use a variety of ways to identify unwanted messages, such as based on the message body, the sender reputation, URLs in the message, attachments,

A	B	C	E	F	G	H	K	L	M	N	O	P	R	S	T	U	V	X	Z
A	B	C	E	F	G	H	K	L	M	N	O	P	R	S	T	U	V	X	Z
											O	P			T			X	
															T			X	

a	c	e	i	j	o	p	s	v	w	x	z
a	c	e	i	j	o	p	s	v	w	x	z
	c			j				v			
	c										

Figure 3. Our final list of high-fidelity confusable characters for the upper and lower case letters in the English alphabet. On the top row of each column is the Latin character, while the rows below are its confusable Unicode characters.

etc. For example, Gmail's spam filter uses a combination of methods including linear classifiers and artificial neural networks, and claims to block 99.9% of all spam emails [7]. In this experiment, we measure the effectiveness of email filters at detecting scam messages obfuscated with confusable characters. We focus on email filters deployed at three popular web email service providers: Yahoo, Hotmail, and Gmail. These service providers, dominating in the webmail market, are likely to have deployed the state-of-the-art spam filtering technology, and hence allow us to measure the "best case" scenario in scam blocking.

We obtained 90 scam messages that are detected by these three major email providers. More specifically, these messages correspond to 419 scams, a form of advanced-fee fraud in which the fraudster attempts to extract money from the victim by promising a large sum of money in the future. We sent the scam messages from newly registered accounts at each of the three popular email providers, destined to equally fresh recipient accounts. All 90 messages ended up in the recipients' spam folder.

To test whether an obfuscated version of these messages would be detected, we compiled these 90 scam messages using our obfuscator described in Section III-A and sent them from six email accounts (two from each email provider) to three recipients (one from each email provider), all newly registered for the purposes of this study. Each obfuscated scam message was sent by all of the senders to the same three recipients over the course of five days. We deliberately paced the sending of the messages so as to simulate the low-and-slow behavior of scammers and to avoid triggering rate-limits enforced by the email providers.

Table I lists the fraction of messages sent to each recipient that failed to reach the recipients' inbox from each sender account, i.e., blocked by the email filters. All 90 obfuscated scam messages was successfully delivered to all three recipients, and the vast majority of messages made it into the recipients' inbox — 96% of the sent scam messages was successfully delivered.

However, the same message from the same sender may not reach all recipients, e.g., Yahoo sender accounts have a much lower success rate to the Gmail recipient than to Hotmail

Table I
THE FRACTION OF OBFUSCATED SCAM MESSAGES SENT TO EACH RECIPIENT THAT FAILED TO REACH THE RECIPIENT'S INBOX FROM EACH SENDER ACCOUNT.

recipient →	Gmail	Hotmail	Yahoo
Gmail (sender 1)	1/90	0/89	0/90
Gmail (sender 2)	1/90	1/89	2/90
Hotmail (sender 3)	1/90	0/89	0/90
Hotmail (sender 4)	0/90	0/89	0/90
Yahoo (sender 5)	20/90	0/89	0/90
Yahoo (sender 6)	19/90	0/89	0/90

or Yahoo recipients. This shows that the obfuscated scam messages were filtered at inbound time on the recipient side, rather than at outbound time on the sender side. Overall, the block rate of obfuscated scam messages is 7.8% for the Gmail recipient, 0.4% for the Hotmail recipient, and 0.2% for the Yahoo recipient.

The low block rate at these three popular email providers suggests that there is little protection against scam messages obfuscated with confusable characters.

IV. DETECTING OBFUSCATED SCAM

Liu and Stamm [1] suggested using a "reverse mapping" to de-obfuscate homographic spam. We suggest that this can be augmented in several ways to improve the detection.

First, it should be recognized that whereas English-language spam and scam messages dominate the Internet email traffic, there is a market for other languages (and character sets) as well – and commonly, filters blocking spam and scam for other character sets are not getting the same attention as Latin-character abuse does. It is straightforward to scan messages to identify the use of multiple charsets, and to create one output stream for each potential charset that the message could be mapped to. For example, if the first few characters of a message are Latin, and then there are Cyrillic characters, then two obvious candidate mappings would be from Cyrillic to Latin – and from Latin to Cyrillic. For most legitimate messages, there would only be one charset, and therefore, no candidate mappings at all; for the small portion of legitimate messages containing multiple charsets, there would be a very low likelihood that the parsed characters would happen to be confusables, and so, the creation of the mappings could be terminated early. Once a collection of mapped results have been produced, these can be individually scanned for undesirable content.

Second, whereas content-based scanning is beneficial for many types of spam and scam, it is important to recognize that any content-based method is vulnerable to changes in the message contents. This has been partially addressed by the introduction of story line detection [8]. However, we suggest another, more general approach: By identifying and counting the transitions from one character set to another, it is possible to quantify the degree of likely obfuscation in a message. Therefore, even if the mapped content does not trigger a content-based filter, a risk score can be determined based on the number of transitions. Here, different weights can be given to transitions between sentences and words (both having low weight), and to transitions inside words (high weight), reflecting the likely nature of the homograph abuse. To be precise, only transitions involving characters that are identified as confusables could contribute to the score.

V. DISCUSSION

Modern spam filters incorporate a combination of methods to detect unwanted messages, including IP and domain blacklists, user behavior analysis, anomaly detection, and, more recently, machine learning and deep learning models. As a result, it can be difficult to isolate the cause for a message being classified as spam (or not). In our experiments described in Section III-B, the obfuscated scam messages were able to bypass spam filters at popular email services, while the original (unobfuscated) messages were blocked. We acknowledge that obfuscation may not be the sole reason for the different outcomes, though we made attempts to minimize discrepancies between the two cases so as to reduce the impact from other factors not related to the message content.

Some homographic characters are more visually similar to their Latin counterparts than others (see Section III-A for our method for selecting "high-fidelity" confusable characters). However, this may also be affected by the choice of fonts in which the characters are displayed. An email scammer can use web fonts (e.g., Google Web Fonts [2]) to ensure that the message will be displayed as expected to the victim, even if that font is not installed locally on his or her machine. In a more extreme attack, a malicious font can be crafted to map one character to another (similar to a substitution cipher), which will allow the scam message to bypass signature-based detection. A future direction would be to demonstrate and evaluate the effectiveness of this atack.

VI. CONCLUSION

We show in this work that homograph attacks using "high-fidelity" confusable characters is a desirable tool for scammers wishing to generate polymorphic messages that are visually identical to the "plaintext messages" they are derived from. This type of attack, applied to low-volume *scam* messages, would drastically limit their blocking efficacy. We propose countermeasures to identify possible homograph messages and (where applicable) perform a mapping to determine how a human recipient would interpret them. The mapped result could then be screened by content-based filters. We argue that these protective techniques should preferably be implemented and deployed before they are needed, as not doing this would correspond to exposing end users to an unnecessary risk. From our experiments, it is clear to us that countermeasures are not currently in use.

ACKNOWLEDGMENT

The first author would like to thank Mengxi Tian for helping to weed out the confusables and test the obfuscator.

[2]https://fonts.google.com/

REFERENCES

[1] C. Liu and S. Stamm, "Fighting Unicode-Obfuscated Spam," in *APWG eCrime Researchers Summit*, 2007.

[2] E. Gabrilovich and A. Gontmakher, "The Homograph Attack," *Communications of the ACM*, vol. 45, no. 2, 2002.

[3] T. Holgers, D. E. Watson, and S. D. Gribble, "Cutting through the Confusion: A Measurement Study of Homograph Attacks," in *USENIX Security Symposium*, 2006.

[4] J. Klensin and Y. Ko, "Overview and Framework for Internationalized Email," http://www.rfc-editor.org/info/rfc6530, 2012.

[5] "Business Email Compromise," https://www.ic3.gov/media/2015/150827-1.aspx, 2015.

[6] "Unicode Technical Standard 39: Unicode Security Mechanisms," http://www.unicode.org/reports/tr39/.

[7] S. H. Somanchi, "The mail you want, not the spam you don't," https://gmail.googleblog.com/2015/07/the-mail-you-want-not-spam-you-dont.html, 2015.

[8] M. Jakobsson and W. Leddy, "AI vs. the Phishers," in *IEEE Spectrum Magazine*, 2016.