Distributed Phishing Attacks

Markus Jakobsson* Adam Young†

Abstract

We identify and describe a new type of phishing attack that circumvents what is probably today's most efficient defense mechanism in the war against phishing, namely the shutting down of sites run by the phisher. This attack is carried out using what we call a *distributed phishing attack* (DPA). The attack works by a *per-victim* personalization of the location of sites collecting credentials and a covert transmission of credentials to a hidden coordination center run by the phisher. We show how our attack can be simply and efficiently implemented and how it can increase the success rate of attacks while at the same time concealing the tracks of the phisher. We briefly describe a technique that may be helpful to combat DPAs.

Keywords: Covert channels, distributed attacks, phishing, social engineering, security.

1 Introduction

Phishing can be thought of as the marriage of social engineering and technology. The goal of a phisher is typically to learn information that allows him to access resources belonging to his victims. The most common type of phishing attack aims to obtain account numbers and passwords used for online banking, in order to either steal money from these accounts or use them as "stepping stones" in money laundry schemes. In the latter type of situation, the phisher, who may belong to a criminal organization or a terrorist organization, will transfer money between accounts that he controls (without stealing money from either of them) in order to obscure the actual flow of funds from some payer to some payee. Phishing is therefore not only of concern for potential victims and their financial institutions, but also to society at large.

While most phishing attacks are relatively unsophisticated, there is a very clear trend towards them becoming more and more clever, both in terms of the psychological aspects and the technology deployed. As this is occuring, the organizations concerned with preventing phishing attempts are also developing improved countermeasures. Without any *definitive* attack or countermeasure in sight, this is likely to remain a cat-and-mouse race where each party keeps trying to anticipate the other's next move.

The typical phishing e-mail of today looks like a legitimate e-mail from some organization, such as a bank, and contains a link to a webpage that looks identical to the *real* webpage, but which is controlled by the attacker. On this page, the user is prompted to log in; any captured user names and login credentials are sent to the phisher.

*School of Informatics, Indiana University at Bloomington, Bloomington, IN 47406. `www-markus.jakobsson.com`
†LECG LLC, Washington DC, adamy@acm.org.

Among the greatest threats to the success of such an attack is currently *detection*. That is, as soon as an attack is detected, the organization that the phisher is trying to represent will do its best to have the offending site taken out of commission. The (often innocent) host to the phishing page is likely to comply with a request to deactivate a phishing site, especially in light of potential legal actions were they to refuse. Detection mechanisms are likely to improve by the widespread use of deployment of honey pot techniques, collaborative detection [12], and by incentives to users for forwarding suspect phishing emails. (The latter may be more effective for highly targeted attacks, which are predicted [8] to become more common.)

While the phisher faces a very small risk of being personally tracked down and caught when his attack is detected and his password "collection center" is shut down, he is still affected by it, as it stops any further credentials from being harvested, and thereby limits the success rate of his attack. This, in turn, impacts the economy of phishing and prompts phishers to attempt more aggressive attacks.

In this paper, we describe a novel type of phishing attack that is immune to the effects of detection. We refer to this attack as a *distributed phishing attack* – or DPA in short – as the phisher will not rely on *one* collection center, but on a vast multitude of these. In the extreme case – when each victim is referred to a unique webpage – the benefits of detection vanish, assuming the different webpages used as collection centers are not clustered in a way that allows service providers and law enforcement to find many of these given only knowledge of some. In other words, if each potential victim of an attack is pointed to a webpage of a unique owner and location, neither of which can be predicted without access to the phishing e-mail, then the impersonated organization stands helpless in trying to shut down the collection sites.

To remain profitable to an attacker, a distributed phishing attack must limit the number of paid accounts that the attacker needs in order to perpetrate the attack. The easiest way of ensuring this is to have the collection centers hosted by unsuspecting users with reliable network connections. This can be achieved by means of malware, but also using software of a more symbiotic nature, which may be *intentionally* installed due to their known and beneficial features. A prime example of such software is a popular game that can be obtained for free. Such software would remain benevolent until triggered, at which time it would collect the desired credentials, and somehow transmit these to a *main* ion center, directly controlled by the phisher. At that time, it is too late to close down the site from which the software was obtained in the first place, which would – of course – be hosted at a site that cannot be traced to the phisher.

We note that the phisher would not want the code at the collection site to betray the location of the main collection center, or the phisher's benefits of distribution would be undone. This is so given that it would otherwise allow the impersonated organization or law enforcement to shut down that main site upon discovery of one attack instance (and reverse-engineering of the corresponding code.) It is also of importance to the phisher that law enforcement be unable to identify communications that correspond to delivery of captured credentials, or worse, detect whose credentials they are, or what the credentials are. To avoid that law enforcement closes down or otherwise isolates the site to which credentials are sent, the phisher may have this information posted on a large number of public bulletin boards using steganographic techniques. To avoid meaningful analysis of all posted messages in order to extract information about credentials, the phisher will public-key encrypt all credentials before the steganographic encoding is performed.

Apart from describing how a well-engineered distributed phishing attack would work, we also consider what can be done to protect against such attacks. In particular, we describe a general technique by which DPAs can be detected and apprehended. This relies on (potentially off-line) analysis of transmitted data, with the aim of performing central identification of likely DPAs, after which automated alerts can be generated to the ISPs of the host pages used in the DPA.

2 Related Work

Almost without exception, today's phishing attacks rely on a small set of tools and tricks to attack victims and cloak the attacks as legitimate requests for information.

Phishers often send their victims obfuscated messages, where the obfuscation may serve to bypass spam filters or to disguise and hide the true content of a message from a human recipient. For example, spammers often insert random words into email, and these words are typed in text that has the same color as the background in order to make emails unique; this constitutes a form of obfuscation that fools many common commercial spam filters. Similarly, an actual hyperlink in a phishing email is commonly not the same as the apparent hyperlink that is displayed to users by most email programs.

The detection of common obfuscation methods will help improve many spam filters and can also be used to provide visual feedback to users, which in turn will allow ordinary computer users to detect and avoid phishing attacks. Techniques to detect possible obfuscation attempts are useful in order to make users aware of phishing attacks. An example of such a tool is [4]. The importance of such tools will increase if phishers start mounting distributed phishing attacks.

Another way of combatting spam was recently proposed by Adida, Hohenberger and Rivest [1, 2]. They propose an identity-based authentication mechanism that retains the repudiability offered by emails of today. By virtue of the identity based construction, their proposal does not rely on a PKI infrastructure, which is an advantage in terms of deployability.

We note that whitelists are not likely to be useful techniques in the fight against DPAs, given their distributed nature. Similarly, blacklists are not meaningful either, in the absence of authentication of email, as is currently the case.

Most phishing attacks aim to capture some personal identifying information (PII) of a victim. This information is captured by having the phisher deploy a Web site that poses as a legitimate service provider of the victim. These fraudulent sites are nearly indistinguishable from the legitimate ones that they impersonate.

Examples of common PIIs are PINs, passwords, mothers maiden names, social security numbers, and the outputs from devices deployed for reasons of authentication, such as the SecurID [13] token of RSA Security. All but the last of these example PIIs are static, and it is clear that they have to be protected so that an attacker does not learn them. While the last example is a dynamically changing PII, it still needs to be protected, as there is still a short window of time during which it can be used by an attacker in order to gain access to a resource associated with the token. In

many cases, it suffices for an attacker to get access to a resource once for the damage to be done.

A good second line of defense against phishing is therefore the use of *mutual authentication* techniques, such as [7]. This would alert the user of any attempt made to masquerade as a service provider that the user is doing business with, and allow the user to abort the authentication before his PII is leaked. The importance of this type of defense would also increase in the light of DPAs.

Another threat is the recently described "doppelganger window" attack, proposed by Jakobsson and Myers [9]. This is an attack in which a victim is presented with a window that *appears* to be a valid login window using some secure method (whether SSL or some mutual authentication method), but that is not. The window would instead simply send the captured password to the attacker (who controls the site corresponding to the doppelganger window). This attack therefore approximates the effects of a keyboard logger, but without having to corrupt the victim machine. In [9], a visual feedback mechanism is proposed to defeat the doppelganger window attack.

Another known attack causes leaks of PIIs by side channels; here, a side channel might be a rogue site that asks users to register to get service, with the hope that the user would reuse PIIs used with other service providers. The password randomizing plugin proposed by Boneh et al. [5] would serve to prevent such leaks.

A related weakness corresponds to mechanisms to help users that have forgotten their passwords, which may be done either by sending the stored password to the user, or to allow the user to reset it – both after the user has authenticated itself properly to the site. The authentication can either be done by security questions (such as done in [10]) or by proving access to an account associated with the user. In the former case, reuse of security questions remains a problem unless a plugin is used for randomization of these (the user needs to make sure that such a plugin is *always* acting as a filter for information that needs to be protected). In the latter case, one relies on the secure and confidential delivery of emails, which is a reasonable assumption in most cases.

3 Background

The phishing cryptotrojan that we present obtains the login password pairs of phishing victims, encrypts these pairs using public key steganography, and then broadcasts the resulting files to a public bulletin board. This can be done using image files that are posted to Usenet. We refer the reader to [11] for information on modern steganographic techniques.

The benefit of having malware covertly broadcast asymmetric ciphertexts was shown in [18]. The fact that it is *broadcast* means that the phisher cannot be singled out when he reads the broadcast, since everyone obtains the broadcast. *Public key cryptography* is needed since the phisher does not want anyone else to be able to decrypt the ciphertexts. Finally, *public key steganography* is needed because it ensures that bulletin board hosts cannot identify the presence of the ciphertexts in bulletin board posts. For, if an embedded asymmetric ciphertext can be identified as such, then news servers have the option of rejecting image files that contain it. We refer the reader to [11] for information on modern steganographic techniques.

Public key steganography ensures that asymmetric ciphertexts are indistinguishable from the noise that one would typically find in a multimedia file, for example. Public key steganography can, for instance, produce asymmetric ciphertexts that are polynomially indistinguishable from bit strings that are chosen uniformly at random. The foundation of public key steganography has recently been placed on theoretical grounds [3] (and among other things, it utilizes probabilistic bias removal [17]).

A straightforward steganographic embedding of an asymmetric ciphertext in an image file does not always satisfy the definition of a public key stegosystem since the embedded data may be readily identified as an asymmetric ciphertext. For example, consider the use of ElGamal with a prime modulus that is public. Multiple embeddings of ElGamal ciphertexts could potentially be detected since both values in each ciphertext will be less than the prime modulus.

The use of public bulletin boards to enable viruses to conduct remote communication was presented in [16]. A public bulletin board, when implemented without a central point of control, has the critical property that it cannot be taken down by law enforcement. A worm called *Hybris* utilizing Usenet [15] was discovered in the year 2000. This worm contains the RSA public key of its author; it receives signed and encrypted patches from its author at runtime by reading alt.comp.virus posts, where a post consists of a single ciphertext. The worm only installs patches when they are properly signed, thereby giving the author the exclusive ability to update the worm after deployment. Prior malware had used a central website to provide updates to deployed malware and the central site was immediately taken down. This and the use of alt.comp.virus in Hybris is discussed in [14].

4 The Distributed Cryptotrojan Phishing Attack

Notation: Let $A \,||\, B$ denote the concatenation of string A with string B. Let $e \in_R S$ denote the operation of selecting an element e uniformly at random from set S.

The distributed phishing that we attack makes use of distributed computation (fault-tolerance), cryptovirology, and public key steganography. In particular a distributed phishing attack is a 3-tuple of malware programs $(transmitter, transponder, receiver)$. These programs are the transmitter application, the transponder cryptotrojan, and the receiver application. In the initial setup phase, the phisher covertly installs the transponders on numerous machines. This is akin to "zombie" machines in a distributed denial of service attack.

The distributed attack is carried out as follows. Let y denote the public encryption key of the phisher and let x denote the corresponding private decryption key. There are M transponders in total. Let B denote a public bulletin board

Transmitter():
Input: none
Output: phishing e-mail that has a forged source e-mail address
NonvolatileStorage: L_1 is a set of e-mail addresses of potential phishing victims
$L_2 = \{(s_1, s_2) : s_1$ is the e-mail address of an impersonated organization and

s_2 is the URL of the phishing page that impersonates s_1}.
1. if L_1 is the empty list then halt
2. select address $a \in_R L_1$
3. set $L_1 = L_1 \backslash \{a\}$
4. select $(s_1, s_2) = t_i \in_R L_2$
5. construct the body of the phishing e-mail message θ that includes a hyperlink to s_2
6. e-mail θ to a using forged source e-mail address s_1

The pair t_i constitutes "identity" of transponder i. Note that when this algorithm is invoked multiple times no e-mail address in L_1 will be phished more than once. Also, observe that it is possible to use an anonymizing service followed by source e-mail forging to untraceably send the e-mail θ to a.

Let $c = \mathsf{StegoEnc}(y, d, m)$ denote the stegotext that results from asymmetrically encrypting m using public key y and embedding the resulting ciphertext in d. We assume that StegoEnc is secure against chosen-ciphertext attacks. Decryption is denoted by $(m, errcode) = \mathsf{StegoDec}(x, c)$. If $errcode = FAILURE$ then c is an invalid stegotext. If $errcode = SUCCESS$ then m is the correct plaintext. Note that this feature (i.e., obtaining an error code from the decryption function) is a standard facility in asymmetric cryptosystems that are secure against chosen ciphertext attacks.

$\mathsf{Transponder_i}(y)$:
Input: public encryption key y of the phisher
Output: stegotext message c
NonvolatileStorage: L_3 is a set of data files that each support steganographic information transfer.
1. if L_3 is the empty list then halt (i.e., site no longer services HTTP requests)
2. present a login prompt and a "sign in" button to users that establish a web connection
3. if the user enters a login and password pair and clicks on "sign in" then:
4. let α denote the login and password pair
5. present a forged page of content, or indicate an HTTP error, etc.
6. choose $d \in_R L_3$
7. set $L_3 = L_3 \backslash \{d\}$
8. set $m = \alpha \mid\mid i$
9. $c = \mathsf{StegoEnc}(y, d, m)$
10. post c anonymously to one or more bulletin boards B

Note that it is possible to have the transponder digitally sign α but we ommit this mechanism since α can always be chosen maliciously by "phishing victims" (e.g., chosen to be bogus). Also, note that no data file $d \in L_3$ will be used more than once. This is to prevent multiple postings of this data to B. The transponder constitutes a cryptotrojan since it contains and uses public key y.

Remark: If a bulletin board is shut down[1] then the attacker will still want to succeed. Therefore, each instance of $Transponder_i$ can have a sequence of bulletin board addresses that it attempts to use one by one until the post succeeds.

[1]This is probably not likely.

Continuing with our description of the attack, the phisher peruses the bulletin board B and downloads c. This is then supplied to algorithm StegoDec along with private decryption key x.

Receiver(x, c):
Input: private decryption key x of the phisher and stegotext c
Output: FAILURE or the login and password pair α along with i
1. $(m, errcode) = \mathsf{StegoDec}(x, c)$
2. if $errcode = FAILURE$ then halt with $FAILURE$
3. extract the login pair α and integer i from $m = \alpha \mid\mid i$
4. output (α, i) and halt

Observe that if α is a proper login and password pair to an account at organization i, then (α, i) gives the phisher access to this account.

5 Ways of Implementing the DPA

It is worth shedding light on some of the details behind how such an attack can realistically be carried out. To install the transponders, the normal attack vectors of Internet attackers can be used. This typically involves one form of exploit or another. Examples include buffer overruns, exploiting improperly mitigated race-conditions, getting a user to execute an attachment, and so on.

In MS Windows operating systems, the asymmetric encryption in the transponder is straightforward to implement. Both Windows 2000 and Windows XP are equipped with the Microsoft Cryptographic API (CAPI). The transponder need only obtain a handle to the desired Cryptographic Service Provider (CSP) at runtime. These operating systems ship with both a 1024 bit RSA CSP as well as a Diffie-Hellman CSP (that can be used to implement ElGamal). The steganographic encoding functionality, however, would have to be included in the phishing transponder.

A bulletin board B that can be used is Usenet. This broadcast medium was originally created and deployed independently from the Arpanet. Only later was it adapted to function on the Internet as well. Originally, the Unix to Unix CoPy (UUCP) protocol transfered newsgroup posts via modem. Usenet was originally controlled (for the most part) by "the backbone cabal," since it was the cabal that incurred the bulk of the long distance phone charges.

However, there is no longer any central point of control as illustrated by the proliferation of the infamous "alt" hierarchy. This is an ideal bulletin board because it is free, does not require an account, and has no central point of control (though local Usenet admins can filter out groups of their choosing). Example newsgroups that currently support pictures, and therefore steganographic broadcasts, include alt.binaries.pictures, alt.binaries.pictures.animals, and alt.binaries.pictures.autos.

An useful approximation of a bulletin board may be a free email address that serves as a collection center for some not too large set of hosts, and which the attacker downloads results from occasionally, from physical locations that are not traceable to his person. If this is used, then the attacker

does not have to use steganographic methods, as the data would not be publicly available; this would also simplify the implementation of the attack.

6 Detecting and Preventing DPAs

We will now outline a possible technique to defend against DPAs. Our focus is not to treat this topic in detail – in fact, we do not have any experimental data to support our suggestions. Instead, the intention of this section is to provide indications that while the problem is severe, it can in fact be addressed. We hope that further research efforts will provide more solid support for this general technique.

Compiling candidate sets. Phishing emails typically contain a common set of keywords and images[2] that allow them to be identified and screened. We propose the installation of simple phishing identifiers on the backbone; these would scan all traffic and report candidate phishing emails to a central processing unit. One may also rely on honey pots and user feedback to identify potential phishing emails.

Identifying DPAs. The processing unit receives as input a large collection of emails. Many of these may be legitimate, and not be part of a phishing attack; others correspond to traditional phishing attacks; and still others are part of distributed phishing attacks. It is relatively difficult to distinguish members of the two first sets from each other with certainty, but we argue that one can distinguish members of the third set from those of the others with a high certainty. Namely, emails corresponding to a DPA are characterized by (1) a high degree of similarity[3] in terms of contents and appearance, and (2) references to URLs with no particular relation to each other. It is possible to also compare the pages *pointed to* by candidate members of the DPA; these are likely to exhibit a large degree of similarity as well.

Generating alerts. Once a set of URLs of DPAs has been compiled, then the corresponding ISPs will be contacted with requests to prevent accesses to the offending sites. In contrast to how such requests are handled today (by phone calls from the targeted service providers to representatives of the corresponding ISPs), the distributed nature of the attack we consider requires automation of this step. Thus, the classification engine (whether run by the attacked service provider or by some central service provider) will generate emails to the hosts of the offending sites. We note that these emails should be authenticated to prevent an attacker from impersonating the classification engine to shut down selected sites. It is also likely to be necessary for them to be processed automatically by the ISP, to avoid inundation of and DoS attacks on these. Finally, it is desirable that the ISP verifies the existence of the offending sites before blocking them; this can be done simply by simulating an access by the victim to the offending site, and automatically compare the contents to some general template associated with the attack (i.e., availability of copyrighted images.)

[2]While the images may intentionally be dithered by the phisher to make matching difficult, one may use OCR tools to compare these to target images.

[3]We note that our proposed defense technique does not rely on the emails being identical: one can determine that two or more texts are highly similar using a tool such as [6].

In addition, it is possible to send alerts to recipients of emails that have been identified as belonging to the DPA. If such a warning is read before the attack email is accessed, this provides us with a second line of defense (in light of the fact that not all ISPs agree to shut down offending sites.)

7 Conclusion

We identified and described a distributed phishing attack (DPA) that circumvents the shutting down of sites run by the phisher. The attack utilizes public key steganography and covert broadcasts for later reconnaisance by the phisher. The use of secure public key steganography assures that the stego channel cannot be detected, and the broadcast channel ensures that the phisher cannot be identified when reading the broadcast.

References

[1] B. Adida, S. Hohenberger, and R. Rivest, "Fighting Phishing Attacks: A Lightweight Trust Architecture for Detecting Spoofed Emails," Draft, February 2005

[2] B. Adida, S. Hohenberger, and R. Rivest, "Seperable Identity-Based Ring Signatures: Theoretical Foundations for Fighting Phishing Attacks," In submission.

[3] L. von Ahn and N. J. Hopper. Public-Key Steganography. In *Advances in Cryptology—Eurocrypt '04*, pages 323–341, 2004.

[4] D. Boneh and J. Mitchell, "SpoofGuard: Preventing online identity theft and phishing," `http://crypto.stanford.edu/SpoofGuard/`

[5] Dan Boneh and John C. Mitchell, "Web Password Hashing," Stanford University, `http://crypto.stanford.edu/PwdHash/`

[6] C. Collberg, S. Kobourov, J. Louie and T. Slattery, "SPLAT: A System for Self-Plagiarism Detection," ICWI 2003, pp. 508-514

[7] V. Boyko, P. MacKenzie and S. Patel, "Provably Secure Password Authentication and Key Exchange Using Diffie-Hellman," EuroCrypt 2000, pp. 156-171.

[8] M. Jakobsson, "Modeling and Preventing Phishing Attacks," Phishing Panel of Financial Cryptography 2005.

[9] M. Jakobsson, S. Myers. "Delayed Password Disclosure to Defend Against Doppelganger Windows Attacks," In preparation.

[10] A. Juels, M. Sudan, "A Fuzzy Vault Scheme," Proceedings of IEEE Internation Symposium on Information Theory, p. 408, IEEE Press, Lausanne, Switzerland, 2002.

[11] S. Katzenbeisser, F. A. P. Petitcolas. Information Hiding Techniques for Steganography and Digital Watermarking. Artech House, 2000.

[12] Phish Report Network, `www.phishreport.net/`

[13] RSA SecurID, `http://www.rsasecurity.com/node.asp?id=1156`

[14] P. Szor. The Art of Computer Virus Research and Defense. Section 9.6—Update Strategies of Computer Worms. Addison-Wesley, Feb., 2005.

[15] N. Weaver, V. Paxson, S. Staniford, R. Cunningham. A Taxonomy of Computer Worms. In Proceedings of WORM 03, ACM, October 27, 2003.

[16] A. Young, M. Yung. Cryptovirology: Extorsion-Based Security Threats and Countermeasures. IEEE Symposium on Security and Privacy, pages 129–141, 1996.

[17] A. Young, M. Yung. Kleptography: Using Cryptography Against Cryptography. In *Advances in Cryptology—Eurocrypt '97*, pages 62–74, 1997.

[18] A. Young, M. Yung. Deniable Password Snatching: On the Possibility of Evasive Electronic Espionage. IEEE Symposium on Security and Privacy, pages 224–235, 1997.