

A Forensic Framework for Tracing Phishers

Dominik Birk, Sebastian Gajek, Felix Gröbert, and Ahmad-Reza Sadeghi

Horst Görtz Institute for IT-Security
Ruhr University Bochum, Germany

Abstract Identity theft—in particular through phishing—has become a major threat against privacy and a valuable means for (organized) cyber crime. In this paper, we propose a forensic framework that allows for profiling and tracing the agents involved in phishing attacks as well as the associated criminal network. The key idea is to apply phishing methods against phishing agents: in order to profile phishers' behavior their databases are filled with fingerprinted credentials (indistinguishable from real ones) whose deployment lures phishers to a fake system that simulates the original service. We discuss various characteristics that can serve as forensic information. We believe that our proposed framework will be useful for investigators and service providers in tracing criminal activities.

Keywords: Cyber crime, phishing, forensics.

1 Introduction

With the advent of the Internet and the continuous growth of electronic commerce, offering new commercial prospects and opportunities, criminals have also discovered the Internet as source and ground of illicit business. Cyber crime has entered the digital world and prospered to a severe concern for Internet users. Digital crimes appears in many variations, however, online fraud has proliferated recently and become a major threat. A prominent method of fraud is phishing where, e.g., Internet users are lured to faked web sites and tricked to disclose sensitive credentials, such as credit card numbers and passwords. The attacker steals these credentials in order to illegitimately gain access to the user's account. Federal and state law enforcement link organized crime to phishing attacks that involve various actors as part of a professional criminal network, and that deploy methods similar to those of money laundering [1]. There is not much publicly known about these criminal networks, however, their impact is noticeable: according to a report issued by the Gartner Group [2] financial losses were totaled to \$2.8 billion in 2006; one year ago Gartner approximated total losses to \$1 billion [3].

The presented paper is follow-up work of [1], where we discussed aspects of applying methods of phishing against the agents involved in phishing attacks. The key idea is to fill phishers' credential databases with fingerprinted credentials whose deployment lures phishers to a fake system that simulates the

original service in order to profile their behavior. On the one hand, this may deter phishers from harvesting valid credentials. On the other hand, the forensic information may help to expose the agents involved in the associated criminal network. In this paper, we propose a forensic framework and discuss the related forensic information sources that can support investigators and service providers to pro-actively fight phishing online fraud by identifying and tracing the involved actors. We make first steps towards the design and realization of an architecture that is interoperable to commodity web applications, and that does not have any impact on user’s privacy.

2 Basic Idea

The basic idea is based on fingerprinting objects as done, e.g., in real world where one uses serial numbers of bank notes or any other fingerprint to trace the circulation of money: We fill collection servers of phishers with fingerprinted credentials, which we call *phoneytokens* (*phishing honeytokens*). Phoneytokens are applications of honeytokens [4] to phishing and represent data, which looks like a valid credential to the phisher, but can be identified and traced. We expose active collection servers by methods, such as bounces of phishing mails, alerts of phishing report networks¹ or reverse-engineering of phishing malware. After having sent different phoneytokens to the collection server, one can initiate the server’s shutdown (as it is done currently).

Based on the characteristics collected (see Section 5), we derive three classes of phishing profiles, namely, the *non-phisher*, the *definite phisher*, and the *potential phisher*. The non-phisher profile characterizes an honest user who legitimately authenticates to the service and accesses the real system. The definite phisher is unambiguously identified as an adversary according to the use of phoneytokens and is relayed to the phoney-pot. The potential phisher is assumed to be an adversary according to some similarity to a definite phisher. Depending on the degree of similarity, the potential phisher is either relayed to the phoney-pot or we delay the transaction and request for authorization (e.g., we call the account owner to confirm that transaction).

3 Related Work

The authors of the honeynet project [5] report on experiences of hosting a honeypot that was attacked by phishers. The goal of this project was to learn the tactics phishers deploy to mount a phishing attack. The project uses the standard definition of honeypots, which is to deposit a “weak” system and wait until an intruder is attracted. This approach collects forensic information on techniques used by the phisher and targets on the technical side of the phishing (e.g., setting up phishing sites, sending spam emails, controlling spam botnets). Our approach is different. We collect forensic information in order to trace the

¹ e.g., <http://www.phishreport.net/>

usage of stolen (and fingerprinted) credentials. In contrast, we are interested in the phisher’s agents and not in the technical means used.

In an independent work, Chandrasekaran *et al.* [6] propose to use fake credentials and submit them to phishing sites. However, the authors’ key idea is to detect phishing sites according to the response of fake input. In contrast, our approach uses forensic methods to identify the involved actors and already assumes phishing site to be identified. Therefore we use fake credentials (phoneytokens) to lure these agents to fake accounts and to observe their behavior.

Some financial institutes do fraud auditing, which is closely related to the presented approach (see, e.g., [7]). However, fraud auditing is in general a post hoc method, i.e. fraud has to occur and then investigations may be initiated. The presented approach is proactive. It helps expose criminals without any apriori knowledge and bears no financial risks. Hence, the presented approach may be used to complement existing fraud auditing mechanisms.

4 Realization

Fig. 1 illustrates our proposed framework in conjunction with a real system. The framework comprises two main components: The phoneypot that simulates the real system to infer phishing profiles and a phoneytoken machine that generates and distributes phoneytokens.

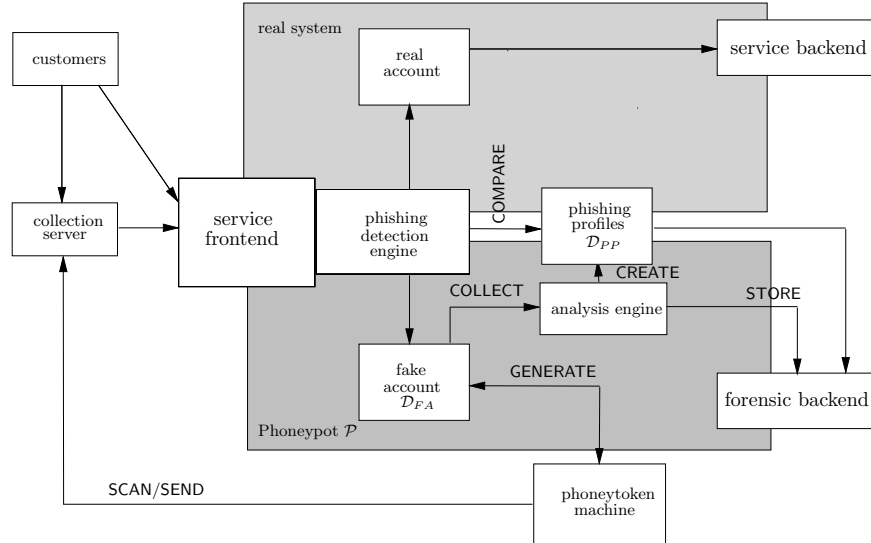


Figure 1. The Architecture of a Phoneypot

4.1 Phoneytoken Machine

The interfaces `SCAN`, `GENERATE` and `SEND` constitute the phoneytoken machine. We use `SCAN` to analyze and extract the attributes required to fill a collection server with phoneytokens. We apply web crawling and form scanning techniques to extract attributes from phishing sites (e.g., [8]); in case of modern web technologies (e.g., Flash) or malware, an analyst has to manually look for the attributes. Typically, the attributes requested are email address, username, password and some service specific credentials, such as credit card number, PIN and TAN or social number. Let \mathcal{C} be the set of attributes requested by a collection server denoted by the address URL , then `SCAN` generates a new table in fake account database $\mathcal{D}_{FA} := (URL, \mathcal{C})$. We use `GENERATE` to create values for a fake account, which ordinary users would normally submit (to a collection server). The phisher is unable to distinguish between user and machine-generated values. We call the entry of \mathcal{D}_{FA} a *phoneytoken* \mathcal{T} where $c_1, c_2, \dots, c_n \in \mathcal{C}$ are the n attributes requested by a collection server URL .

`GENERATE` proceeds as follows: For alphanumerical values, such as names, streets, cities, we create random values of dictionaries or public lists. For (sufficiently long and randomly looking) numerical values, such as PIN, TAN, bank account number, we conceal sequence number and submission date to trace the (jitter) time till the phisher applied this phoneytoken. The jitter time may be used to determine the submission frequency of phoneytokens. If f_k is a function that conceals the sequence number seq and the date of submission $date$, and k a secret key linked to a certain phoneytoken, then numerical values are calculated as concatenation of $date$ and seq . For example, let TAN be an integer, then $TAN = f_k(date||seq)$. There are many possibilities to instantiate the function f . We use the addition modulus 10 for a sufficient long key k (see [9]). A positive side effect is that k also authenticates a phoneytoken \mathcal{T} . This prevents that a phisher creates a phoneytoken \mathcal{T}^* and uses that token to fool the forensics. `GENERATE` also creates additional entries for each phoneytoken \mathcal{T} that are mandatory to build up a fake user account (cf. Section 4.2).

We use `SEND` to submit the phoneytokens \mathcal{T} to the collection server. The phisher should be unable to notice that \mathcal{T} has not been submitted by phishing victims, as an unusual collection of phoneytokens may raise suspicion. Therefore, we require that `SEND` emulates the submission behavior of legitimate customers. We meet the requirement by inferring typical behavior profiles from log files the real system (service frontend) provides and by sending the phoneytokens from different networks. To the authors' knowledge, service providers that are currently target to phishing attacks, such as banks, have the capabilities to provide the phoneytoken machine with different IP addresses, allowing to send numerous phoneytokens from several addresses.

4.2 Phoneypot

The phoneypot is a separated system that is interfaced through the *phishing detection engine* to the frontend of the real web application/service. The phishing

detection engine is integrated into the login and authentication mechanisms of the web application and recognizes users logging in using a phoneytoken. It then redirects this session to the phoneypot. This can be easily achieved with *load-balancing hardware*²

The phoneypot needs to exactly replicate the functionality and the look and feel of the real web application; otherwise phishers may raise suspicion. We call this a *fake account*. This seems at first an involved task, particularly because the modeling of the real web application has to be redone for every application, which should be accompanied by a phoneypot. Fortunately, most security critical web applications have only very few interactive and dynamic elements as a result of non-interoperable browsers (e.g., the web application has to be completely renderable without active content). Therefore, most web sites are purely static and can simply be copied to the phoneypot. Requests for dynamic content is to be proxied by the phoneypot, however, with the slight difference that in order to complete the simulation of a fake user account (e.g., account balance, list of last activities) additional data is derived from the fake account database \mathcal{D}_{FA} .

For profiling, we log all traffic to and from the phoneypot. We use STORE to pass the data to the *forensic backend*, which stores the data in a well documented and secure manner to meet with forensic standards (e.g., as specified by Scientific Working Group on Digital Evidence (SWGDE)³). Also, all components involved in evidence preservation must be thoroughly audited by independent parties.

5 Forensic Data

We use the interface COLLECT to collect information on phishers. This information is passed to the *analysis engine*, where we infer profiles of phishers. The profiles are stored in the phishing profile database \mathcal{D}_{PP} . To build our profiling information, we terminate the communication between the phisher and the phoneypot. We then have access to various link and application characteristics provided by ISO/OSI layers that we take into account to build phishing profiles (see, e.g, [10] for a comprehensive list). The methods we apply are standard passive fingerprinting techniques [11,12,13,14,15]. We do not apply active fingerprinting techniques to avoid being detected by phishers.

ISO/OSI Layer 3 and 4 Characteristics: Connection records of these layers provide many features that are intrinsic to each connection, such as source address and port, destination address and port, and help uniquely identify a connection. There is also additional meta-information available on layer 3 and 4 related to the originating source address. One may estimate the geographical location of the phisher. Inferring the location of an Internet host from its

² Load balancers spread work between many computers, processes, hard disks or other resources in order to get optimal resource utilization and decrease computing time. The phishing detection engine instructs then the load-balancer to transfer the session to the phoneypot.

³ <http://ncfs.org/swgde/index.html>

location has always been a challenging task because there is no correlation between IP addresses and geographical locations. Therefore, we make use of delay measurements of certain packets to infer distance constraints (see [16]). We use the geographic position in order to analyze the requests for granting access to our service. As it has been pointed out in [17], most phishing attacks stem from foreign countries. Vice versa, honest costumers access security critical service usually from fixed locations (e.g., home). This allows for identifying whether a connection comes from a potential phisher, and whether a resolved connection correlates with a given user.

Furthermore, remote device fingerprinting techniques (e.g., [14]) may be used to infer the phisher’s operating system attributes. Device fingerprinting techniques make use of the fact that operating systems implement the TCP/IP stack differently. This leads to delays of network packets that are specific to the configuration of the system. As ordinary users do not often modify their configuration or change their operating system, we make use of these operating system attributes in order to identify a certain machine.

However, we have to take into account that phishers may use masquerading techniques in order to hide their origin, e.g., by having control over a number of anonymization hosts. Therefore, we assume that a single phisher is able to show different networking characteristics on every transaction and falsify the information given on that layer. The proxies used by todays criminal are very often based on botnets, which again mostly consist of regular home PCs [18]. The discussed attributes would then reveal information that identifies the home PC and one would not gain meaningful information on these layers to build up phishing profiles. We increase, however, the efforts that phishers have to make. A phisher has to carefully check for the compromised host and its configuration that he uses to masquerade. Otherwise, anomalous system requests for access to the service are noticed and further investigations can be induced.

ISO/OSI Layer 7 Characteristics: Fingerprinting on layer 7 reveals information about the browser and the operating system used, which is heavily influenced by the browser version, language selection, installed plug-ins, and browser helper objects. For practical purposes the combination of HTTP headers sent by a web browser can often be considered unique to a single installation of that browser.

An additional source of information on the client is the interdependence between different layer 7 connections. This includes domain name requests prior to HTTP requests, the request—or lack of request—for embedded images, additional files (e.g., scripts or flash animations), the order in which they are requested, and finally the timing between these different requests. In particular, the observation of domain name requests considerably increases the chance of identifying the remote communication endpoint. Many anonymization techniques leave these requests untouched and hence such requests stemming from the original machine are not transported via the anonymizing layer [19]. Then the analysis of requests for images and other embedded content helps to distinguish browser-based connections from automated agents, which usually do not

request such objects. Such techniques are already commercially available for user tracking (e.g., Sevenval⁴).

Furthermore, one can apply the concept of cookies. Ordinary cookies are persistent data objects that are stored on the client and allow the server to reestablish a network session, and to identify a certain user. Some protocols use cookies also to authenticate the user. Here one can use a cookie to uncover phishers and set cookies, whenever a phisher logs in using a phoneytoken. However, the downside of cookies is that security policies of web browser might block cookies to safeguard users' privacy. Therefore, one can additionally use *cache cookies* [15]. Loosely speaking, cache cookies allow the issuing server to identify a certain user according to objects previously stored in the browser's cache. In contrast to ordinary cookies, cache cookies are not restricted to browser's policies; however, they are limited to the browser's cache.

Service-specific Data: Whenever a user—be it a phisher or be it an honest customer—makes use of the service (e.g., make a donation), one can profile that behavior. As phishers prefer services that are appealing to clean digital goods into real funds, such as online-banking or auctions, we gain information dealing with aspects of money laundering (see [1]). For instance, in case of online banking, one can intercept the account number, amount due, the date and the recipient of an illicit transaction, where in case of online auctions, one can collect information about goods purchased or sold, contacted persons and messages exchanged using the internal instant messaging functionality, or the methods used to pay for the auction. In any case, one reveals the transactions' destination. As it has been pointed out in [1], direct transactions to foreign accounts in order to clean money are thoroughly verified. Therefore, phishers prefer to address financial agents. However, as hiring financial agents is a tedious task, the number of agents is limited. We may hence assume that same financial agents are likely addressed and that phishers transfer high amounts to financial agents. By contrast to classical money laundering, phishers cannot afford to split the funds and make multiple deposits. Thus, an untypically high transaction sum raises suspicion, making that attribute essential for the detection of a phisher.

6 Conclusion

In this work, we introduce an approach to proactively combat phishing-based identity fraud on the Internet. We propose a forensic framework of luring, trapping and analyzing phishers in order to profile their sources and we have argued how to trace the agents involved in phishing. Currently, we are implementing the framework and evaluate methods used to conduct meaningful statistical results in order to define phishing profiles and to achieve accurate detection rates. It remains an open question whether service providers (e.g., banks) would be willing to deploy our system.

⁴ cf. <http://www.sevenval.com/>

References

1. Birk, D., Gajek, S., Gröbert, F., Sadeghi, A.R.: Phishing phishers—observing and tracing organized cybercrime (2007) accepted for IEEE Workshop on Cyber-Fraud (Cyberfraud'07).
2. Litan, A.: Phishing Attacks Leapfrog Despite Attempts to Stop Them. Gartner (November 2006)
3. Litan, A.: Increased Phishing and Online Attacks Cause Dip in Consumer Confidence. Gartner Study (June 2005)
4. Spitzner, L.: Honeytokens: The Other Honeytrap (2003) <http://www.securityfocus.com/infocus/1713>.
5. The HoneyNet Project and Research Alliance: Know your Enemy: Phishing, Identifying remote hosts, without them knowing (2005) <http://www.honeynet.org/papers/phishing/>.
6. Chandrasekaran, M., Chinchani, R., Upadhyaya, S.: Phoney: Mimicking user response to detect phishing attacks. *wowmom* **0** (2006) 668–672
7. Fawcett, T., Provost, F.: Fraud detection. In Kloesgen, W., Zytrow, J., eds.: Handbook of Knowledge Discovery and Data Mining. Oxford University Press (2002) CeDER Working Paper #IS-99-18, Stern School of Business, New York University, NY, NY 10012.
8. Najork, M., Heydon, A.: On high-performance web crawling
9. Molva, R., Tsudik, G.: Authentication method with impersonal token cards. In: Proc. IEEE Symposium on Research in Security and Privacy. (May 1991) 55–65
10. Zalewski, M.: Silence on the wire. No Starch Press, San Francisco, CA (2005)
11. Zalewski, M., Stearns, W.: Passive os fingerprinting tool (2006) <http://lcamtuf.coredump.cx/p0f/README>.
12. Beverly, R.: A robust classifier for passive TCP/IP fingerprinting. In Barakat, C., Pratt, I., eds.: Passive and Active Network Measurement, 5th International Workshop, PAM 2004, Antibes Juan-les-Pins, France, April 19-20, 2004, Proceedings. Volume 3015 of Lecture Notes in Computer Science., Springer (2004) 158–167
13. Smart, M., Malan, G.R., Jahanian, F.: Defeating TCP/IP stack fingerprinting. In USENIX, ed.: Proceedings of the Ninth USENIX Security Symposium, August 14–17, 2000, Denver, Colorado, USENIX (2000)
14. Kohno, T., Broido, A., Claffy, K.C.: Remote physical device fingerprinting. *IEEE Trans. Dependable Sec. Comput* **2**(2) (2005) 93–108
15. Juels, A., Jakobsson, M., Jagatic, T.N.: Cache cookies for browser authentication (extended abstract). In: SP '06: Proceedings of the 2006 IEEE Symposium on Security and Privacy (S&P'06), IEEE Computer Society (2006) 301–305
16. Gueye, B., Ziviani, A., Crovella, M., Fdida, S.: Constraint-based geolocation of internet hosts. In: Proceedings of the 4th ACM SIGCOMM conference on Internet measurement, New York, NY, USA, ACM Press (2004) 288–293
17. Anti Phishing Working Group: Phishing trend report (2007) www.antiphishing.com.
18. Freiling, F., Holz, T., Wicherski, G.: Botnet Tracking: Exploring a Root-Cause Methodology to Prevent Distributed Denial-of-Service Attacks. Technical Report AIB-2005-07, RWTH Aachen (2005)
19. Dingledine, R., Mathewson, N., Syverson, P.: Tor: The Second-Generation Onion Router (2004) www.citeseer.ist.psu.edu/dingledine04tor.html.