The Phish Market Protocol: Securely Sharing Attack Data Between Competitors

Poster Proposal

Tyler Moore and Tal Moran
Harvard School of Engineering and Applied Sciences
Cambridge, MA 02138
{tmoore},{talm}@seas.harvard.edu

ABSTRACT

A key way in which banks mitigate the effects of phishing is to remove fraudulent websites or suspend abusive domain names. This 'take-down' is often subcontracted to specialist companies. Prior work has shown that these take-down companies refuse to share their 'feeds' of phishing website URLs with each other, and consequently, many phishing websites are not removed because the company with the take-down contract remains unaware of their existence. The take-down companies are reticent to exchange their feeds with each other, fearing that competitors with less comprehensive feeds might 'free-ride' off their efforts and stop investing resources to find new websites, as well as use the feeds to poach clients. To help solve this problem, we propose the Phish Market protocol, which enables companies with less comprehensive feeds to learn about websites impersonating their own clients that are held by other firms. The protocol is designed so that the contributing firm is compensated only for those websites affecting its competitor's clients and only those previously unknown to the receiving firm. Crucially, the protocol does not reveal to the contributing source which URLs are needed by the receiver, as this is viewed as sensitive information by take-down firms. Using the complete lists of phishing URLs obtained from two large take-down companies, our elliptic-curve-based implementation added a negligible average 5 second delay to securely share URLs.

Categories and Subject Descriptors

K.4.4 [**Computing Milieux**]: Computers and Society—*Electronic Commerce, Security*; C.2.4 [**Computer-Communication Networks**]: Distributed Systems—*Distribut ed Applications*

Keywords

phishing, electronic crime, cryptography, secure multiparty computation

1. INTRODUCTION

Phishing is the criminal activity of enticing people into visiting websites that impersonate genuine bank[1] websites, and to dupe them into revealing passwords and other credentials to carry out fraudulent activities. One of the key countermeasures to phishing is the prompt removal of the imitation bank websites. Removal may be achieved by erasing the web pages from the hosting machine, or by contacting a registrar to suspend a domain name from the DNS so the fraudulent host can no longer be resolved.

Although some banks deal with phishing website removal exclusively 'in-house', most hire specialist 'take-down companies' to carry out the task. Take-down companies – typically divisions of brand-protection companies or information security service providers – perform two key services for the banks. First, they are good at getting phishing websites removed quickly, having developed relationships with ISPs and registrars across the globe and deployed multi-lingual teams at 24x7 operations centers. Second, they collect a more timely and comprehensive listing of phishing URLs than the banks are often capable of gathering.

The take-down companies negotiate feeds of raw URLs from several sources: clearinghouses such as CastleCops[2] and the Anti-Phishing Working Group (APWG)[3], individual ISPs, and banks. Additionally, they process their own incoming email spam traps and what turns up at dormant domains that they own. Reported URLs from all sources are examined for accuracy. The companies thereby create their own feed with validated URLs and most false positives removed. The feed is used internally for their own take-down activities, but also occasionally supplied to client banks who perform their own take-downs and to ISPs or domain name registrars who pay to proactively police their own corner of the Internet.

Most take-down companies view their URL feeds as a key competitive advantage over banks and other take-down providers. However, recent work has shown that the feeds compiled by take-down companies suffer from large gaps in coverage that significantly prolong the time taken to remove phishing websites. Moore and Clayton examined six months of aggregated URL feeds from many sources, including two ma-

[1]Although a wide range of companies have been subject to phishing attacks, the vast majority are financial institutions; hence for simplicity, we use the term 'banks' for the firms being attacked.
[2]`http://www.castlecops.com/pirt`
[3]`http://www.antiphishing.org/`

jor take-down companies [2]. They found that up to 40% of the phishing websites impersonating banks hired by take-down companies were known to others but not by the company with the take-down contract. Another 29% of websites were discovered by the responsible take-down company only after others had identified the sites. By measuring the substantially longer lifetimes of these missed websites, Moore and Clayton estimated that at least $330 million per year is being put at risk by the failure to share proprietary feeds of URLs for just the two companies they studied.

But is sharing the answer? If so, then how should an effective sharing mechanism be designed?

Moore and Clayton appealed to the security industry's sense of responsibility and argued that URL feeds should be shared freely between take-down companies and the banks. They point to the precedent of sharing in the anti-virus industry. Anti-virus companies started sharing raw virus definitions in the early 1990's once it became clear that no single company could detect all viruses. Moore and Clayton also claim that most take-down companies (and all banks) stand to gain from increased sharing. Knowing about more websites could offer increased revenue for take-down companies if they charge per site removed. In any case, it would lead to increased customer satisfaction for the banks since websites could be removed more quickly.

However, there are some reasonable objections to a sharing free-for-all. First, it can be argued that competition between take-down companies has driven investment towards developing better techniques for identifying new phishing websites faster. Mandated sharing might undermine the incentive to develop new techniques. Unsurprisingly, most take-down companies would rather see banks purchase the services of several take-down providers to overcome gaps in coverage.

Our proposed solution is *Phish Market*, a protocol that addresses the competitive concerns of take-down companies so that widespread sharing can take place. To bolster the incentive to share, our protocol enables sharing of URLs where the net contributors are compensated without revealing the sensitive details of what is shared to competitors. At a high level, the Phish Market protocol does the following:

1. shares only those URLs that the receiving party wants (i.e., the banks the receiving party works for);
2. does not reveal to the providing party which URLs are given to the receiving party;
3. securely tallies the number of URLs given to the receiving party;
4. does not count URLs the receiving party already has.

Timing is critical when it comes to distributing URL feeds — the longer a phishing website remains online, the more customer credentials may be at risk. Although, in theory, generic multiparty computation protocols can be used to implement this mechanism, they are very inefficient and would introduce significant delays in processing the many thousands of phishing websites. In contrast, our custom protocol is extremely efficient (and still provably secure).

To demonstrate the feasibility of our mechanism, we have implemented an elliptic-curve-based version of the protocol in Java. Using the feeds from two take-down companies during the first two weeks of April 2009, we tested protocol performance in a real-world scenario. We found that our sharing protocol introduces an average delay of 5 seconds to the processing and transmission per phishing URL. In exchange for this very short delay, information on new phishing websites is exchanged between take-down companies so that the overall lifetime of phishing websites may be halved [2] while crediting the contributing firm.

2. THE PHISH MARKET PROTOCOL

An Optimal Ideal-World Protocol.

We describe the task our protocol performs by first explaining how it could be done if we used a trusted third party (TTP) — someone who was entirely trusted by both the contributor (or *Seller*) and the receiver (or *Buyer*). To share data in this ideal scenario, both the Buyer and the Seller would send the data to the TTP; the Buyer's data consists of the URLs she already knows and her list of client banks, while the Seller's data consists of the URLs he is attempting to sell and their classification (i.e., which bank each URL is attempting to impersonate). The TTP could then send the Buyer only those URLs that both impersonate her clients and that she did not already know. The TTP would send the Seller the number of URLs sent to the Buyer. This number would then be used to compute the compensation owed to the Seller. Since the TTP only sends the new "interesting" URLs to the Buyer, she will not learn anything about URLs she was not interested in (and would not have to pay for them). On the other hand, the TTP sends the Seller only the number of URLs sold, not the URLs themselves. Consequently, the Seller will not gain additional information about the Buyer's client list.

Our protocol is intended to provide this functionality, maintaining its privacy properties, but without requiring a third party. Using powerful results from theoretical cryptography, it is known how to convert any task that can be performed with the aid of a TTP to one that does not require third parties. However, these techniques are usually inefficient. In our case, even the most efficient implementations of general techniques (such as the Fairplay system [1]) would be orders of magnitude too slow for practical use.

We give an efficient protocol for executing a single 'transaction' of the following form: the Seller first sends a 'tag' to the Buyer. The tag can be, for example, the name of the bank associated with the URL to be sold. The Buyer uses the tag to decide whether or not she is interested in learning the corresponding URL. She also commits in advance to the set of URLs she already knows. If the Buyer was interested in the tag and did not already know the corresponding URL, the Seller receives a 'payment'. Otherwise, the Seller receives a 'counterfeit payment' (the Seller should not be able to tell whether or not a payment is counterfeit — this would indicate whether or not the Buyer was interested in the URL, and thus give information about the Buyer's client list).

At the end of some previously agreed period (or number of transactions), the Buyer reveals to the Seller how many 'real' payments were sent, and proves that this is indeed the case (without revealing which of the payments were real).

In practice, we envision each pair of take-down companies executing the basic protocol in both directions: when one of the companies acquires a new URL, it would execute the pro-

tocol as the Seller, with the other company playing the Buyer. When the second company acquires a new URL, it would execute an instance of the protocol in the other direction, with the first party as Buyer and the second as Seller.

2.1 Phish Market Protocol Overview

Payment Commitments.

Before we describe the protocol itself, we must clarify what we mean by 'real' and 'counterfeit' payments. Our protocol uses cryptographic commitments as payment tokens. Loosely speaking, a commitment to a value x can be thought of as a public-key encryption of the number 1, for which only the Buyer knows the secret key; the Seller can't tell what x is from the commitment, but the Buyer can 'open' a commitment and prove to the Seller that the commitment is to a specific value. In our protocol, A 'real' payment is a cryptographic commitment to the number 1, while a 'counterfeit' payment is a commitment to the number 0.

The payment commitments used by the protocol have a special property that allows them to be efficiently aggregated, even in encrypted form (they are *homomorphic*). Thus, the Seller can take the 'payments' from multiple executions of the basic protocol and compute a commitment to the total payment (the number of URLs actually 'sold').

The Buyer will eventually open the aggregated commitment. At this point, the Seller will learn only the total number number of 'real' payments received (and not which individual payments were real). This value can be used as the basis for a monetary transaction between the two parties.

Protocol Construction.

One of the more difficult challenges to solve efficiently is that the Buyer should not have to pay for URLs she already knew, while simultaneously protecting the privacy of the Buyer's client list. The known techniques for general secure computation of a function require an expensive public-key operation for each input (or even each bit of the input). In our case, the input would have to include the set of previously known URLs, which may be very large: A typical take-down company could learn an excess of 10 000 URLs per month, making existing systems impractical.

To solve this problem, we let the Buyer perform the database search locally, after learning the URL. If she discovers the URL in the database, she must then prove to the Seller that the URL existed in the database before the start of the transaction. However, this proof cannot use the URL itself, since that would reveal to the Seller that the Buyer was interested in it (thus exposing one of the Buyer's clients). The main idea behind the protocol is to split the proof into two:

1. The first proof is a 'proof of payment'. The payment in this case is a commitment to the value 1; the proof of payment proves that the Buyer can open the commitment she sent to the value 1.

2. The second proof is a 'proof of previous knowledge'. This proof convinces the Seller that the Buyer knew the URL before the start of the protocol.

The essence of the protocol is that we allow the Buyer to 'fake' a proof if she knows a corresponding secret key. The protocol is set up so that the Buyer initially knows a single secret key:

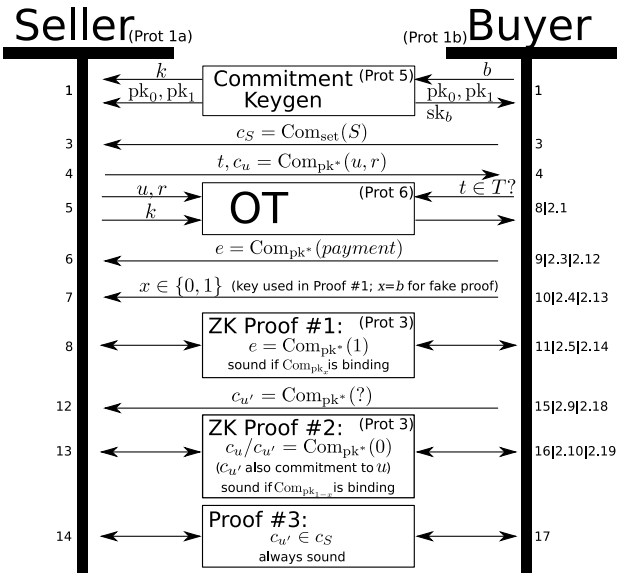

Figure 2.1: Simplified Phish Market protocol overview.

she can fake the first proof or the second proof, but not both. Once the Buyer learns the tag, she must make a choice: she can either learn the corresponding URL, or learn the second secret key (but not both). Thus, if she chooses not to learn the URL, the Buyer can send a counterfeit payment (a commitment to 0), and fake both proofs. If she chooses to learn the URL and did not already know it, she is forced to fake the second proof, and therefore cannot fake the first (so she must send a real payment). This choice is enforced by using *oblivious transfer* (OT), a basic primitive in many cryptographic protocols. The proofs we use are *Zero-Knowledge* (ZK) proofs: the Seller learns nothing from the proof except the validity of its statement. This protects the security of the Buyer (the Seller cannot tell whether or not the Buyer was interested in the URL or whether she previously knew it).

Fig. 2.1 shows a graphical overview of the protocol. We split the second proof into the boxes labeled *ZK Proof #2* and *Proof #3* in the figure. Before the protocol begins, the Buyer sends the Seller a commitment to her set of previously known URLs. ZK Proof #2 proves the Buyer holds a commitment for the URL (this part can be faked using a secret key). Proof #3 proves the Buyer knew the commitment before the protocol began (this part cannot be faked; however, if the Buyer faked ZK Proof #2 she just needs to prove she previously knew some other commitment (of her choice). The reason for the split is that Proof #3 can be performed very efficiently, while Proof #2 requires public-key type operations.

3. REFERENCES

[1] D. Malkhi, N. Nisan, B. Pinkas, and Y. Sella. Fairplay — a secure two-party computation system. In *USENIX Security Symposium*, pages 287–302, 2004.

[2] T. Moore and R. Clayton. The consequence of non-cooperation in the fight against phishing. In *Anti-Phishing Working Group eCrime Researchers Summit (APWG eCrime)*, pages 1–14, 2008.