ISSN 2395-1621

Survey Paper on Detecting Phishing Attacks Using PhishStorm

[#1]Sucheta Shinde, [#2]Swarupa Kamble

[1]suchetashinde88@gmail.com
[2]swar.kamble@gmail.com

[#12]Department of Computer Engineering, RMD Sinhgad School of Engineering, Pune, India.

ABSTRACT

In the cyber world theft done by hackers is commonly achieved using phishing. But the phishing techniques are still based on reactive URL blacklisting. This is not sufficient due to the short lifetime of phishing Web sites, while the recent approaches based on real time or proactive phishing URL detection techniques. To overcome this problem, we introduce PhishStorm technique, which can automatically analyse any real time URL to identify potential phishing sites. It can interface with any email server or HTTP proxy. It contains the low level domain which is the registered part of the URL and upper level domain, path and query which is remaining part of the URL. We use a concept intra-URL relatedness, which can evaluate the features extracted from words that compose a URL based on query data from Google or Yahoo search engines. These features are then used to detect phishing URLs from real dataset.

ARTICLE INFO

Article History

Received :30th December 2015

Received in revised form :

31st December 2015

Accepted : 1st December , 2015

Published online :

2nd December 2015

Keywords— search engine query data, phishing detection, word relatedness, Storm.

I. INTRODUCTION

As the URL blacklisting method cannot measure the real time URLs. To solve this problem we define PhishStorm, an automated real time URL phishing system to protect user against phishing content. This method targets identification of phishing URLs that are based on registered domain that are not related to their targeted brand. The phishers may insert many phishing keywords that is famous brand or any attractive keyword into the reaming part of the URL. There are few relationships between the registered domain and rest of the URL of the phishing URLs. It uses the relatedness of words that compose the URL and highlights the difference between legitimate and phishing URLs.

PhishStorm is a technique which can automatically detect any real time URL to identify potential phishing sites. As we know that URL can be composed of two parts that is low level domain which is registered part of the URL and upper level domain, path and query that is remaining part of the URL.

From the phishing URLs it is observed that, there are few relationships between registered domain and rest of the URL. Also the words that compose the rest of the URL that is low level domain, path and query often have many interrelationships. So, PhishStorm evaluates the relatedness of words that compose an URL and highlights the difference between legitimate and phishing URLs. We uses search engine query data from Google and Yahoo to compute this relatedness. Then we extract 12 features from a single URL which are input to machine learning algorithms to identify phishing URLs.

The term intra-URL relatedness defines the relationship between registered domain and the words that compose the rest of URL. Based on search engine query data we built a machine learning based approach to distinguish between phishing and non-phishing URLs. Also we can use the space efficient data structure that is bloom filter to reduce the delay in intra-URL relatedness feature computation.

II. EXISTING SYSTEM

Phishing email will direct the user to visit a website where they are asked to update their personal information such as password, credit card number or bank account number that the legitimate organization already has. This also makes the loss of money, data and time to overcome this problem many techniques are used which can detect the phishing sites by using URL blacklisting, DNS cache poisoning, email spoofing.

There are number of technologies used for phishing. Some of them are described below:

2.1 PhishNet

PhishNet [1] contains two components: URL prediction component and URL matching component. In URL prediction component that works in an offline fashion, examines current blacklists and systematically generates new URLs by employing various heuristics. Further, it tests whether the new URLs generated are indeed malicious with the help of DNS queries and content matching techniques in an automated fashion, thus ensuring minimal human effort.
In URL matching component which performs an approximate match of a new URL with the existing blacklist. It uses novel data structures to perform approximate matches with a n incoming URL based on regular expressions and hash maps to catch syntactic and semantic variations.
But the drawback of this system is that it suffers from low false positive and also have efficiency problem.

2.2 PhishLive

PhishLive [4] defines the behavior of malicious website attacks on users and hosts of the campus network of a large University by monitoring the HTTP connections for malicious accesses (using the Google safe browsing tool). PhishLive monitors the HTTP traffic going through the gateway of the campus network and captures malicious URLs detected by Google Safe Browsing (GSB) database in HTTP requests and redirect responses in real time. It analyzes the statistical characteristics of dataset off-line including distribution of attacks over time, relocation distribution of attacking IP addresses, attacking host- names clustering and malicious redirect chain analysis.
But the drawback of this system is it only focuses on nature of the attacks not prevent it.

2.3 PhishGILLNET

PhishGILLNET [3] tries to catch phishing attacks by the tone, wordings, and other linguistic variations in the content. By serving as a server side filter, PhishGILLNET prevents movement of a phish towards the end user. The first layer of PhishGILLNET (phishGILLNET1) employs PLSA to build a topic model and uses a topic level similarity function for classification. The second layer of PhishGILLNET (phishGILLNET2) employs classifier ensemble technique AdaBoost and topic probabilities as features to build a robust classifier using several base learners. To further expand PhishGILLNET to hand le labeled and unlabeled email data, the third layer (phishGILLNET3) employs Co-Training to build a classifier using topic distributions as features and the best classification technique obtained in the second layer.

But the drawback of this system is it detect attacks as long as content is available.

2.4 Exposure

Exposure [2] was designed to perform passive DNS analysis to detect domains that are involved in malicious activities, such as hosting phishing web pages, dot net command and control servers, drop zones etc. Exposure leverages machine learning techniques to build detection rules that are effectively able to distinguish the DNS behavior of malicious services from the benign ones. In particular, it employs four sets of features that are extracted from anonym zed DNS records: time-based features, DNS answer-based features, TTL-based features and domain name based features.

But the drawback of this system is it cannot detect all kinds of malicious domains. Also it does not provide solution for domain names of web sites.

III.PRAPOSED SYSTEM

3.1 Obfuscation Techniques

URL obfuscation techniques are used to hide the real host and registered domain.

A registered domain consists of two parts: a main level domain and a public suffix. A main level domain (or mld) is the level domain preceding a public suffix. For eq. it may be any product name or company name. A public suffix (or ps) is a domain name suffix under which an Internet user can register a name. It can be just a Top Level Domain like .com, .org or a combination of level domains like .co.uk or .blogspot.com. A registered domain is then: mld.ps. For instance in www.paypal.com/login, com is the ps and paypal is the mld.

Obfuscation techniques mix the original domain name or phishing keywords with the remaining part of the URL. The phishing keywords are usually brand name that may be targeted, services related to the brand and other attractive words such as secure, login, protect, etc.

- URL obfuscation with other domain: The mld.ps is a real domain name, usually registered by the phishers, while the original domain being phished is part of the path, the query or the upper level domain.
- URL obfuscation with keywords: Again the mld.ps is a real domain name, and the brand being phished and related words are part of the path, the query or upper level domain.
- Long Domains: The mld.ps of the URL is the domain being phished but misspelled, with letters or words missing or added, or the domain is pronounced the same way as the original but written differently. The targeted brand can also be combined with other words to create an unregistered domain.
- URL obfuscation with IP address: The URLs hostname is replaced by an IP address and the brand being phished is part of the path or the query.
- Obfuscation with URL shortener: A URL shortening service is used to hide the name of the real host. Such URLs are not meaningful and are mainly used in phishing attacks targeting services that use this kind of short URL, like Twitter.

3.2 URL Word Extraction

We split the URL into 2 parts that is extracting the mld.ps and separate it from the rest. We use Public Suffix List to identify ps as it is composed of multiple level domains and then retrieve the immediately preceding level domain as the mld. For the rest of the URL, a split according to non-alpha-numeric characters is first performed.

Based on these splitting two sets are composed: one, called RD_{url} (for Registered Domain), consists just of two elements: RD_{url} = {mld, mld.ps}. The other, REM_{url} (for REMaining part), is composed of all extracted words from the URL except mld.ps. Given http://sezopoztos.com/paypalitlogin/us/webscr.html?cmd= login-run, the following sets are extracted:

- RD_{url} = {sezopoztos,sezopoztos.com}
- REM_{url} = {paypal, it, login, us, web, src, html, cmd, login, run}

3.3 Search Engine Query Data

We use search engine query data to perform the evaluation of intra-URL relatedness. People generally use search engine to access the services. When they make a search, they type some keywords which is typically brand name or domain name. But these words are used by the phishers to trap the user. We use search engine query data from two top ranked search engines Google and Yahoo which offer services like Google Trends and Yahoo Clues.

Google Trends is an online search tool that allows the user to see how often specific keywords, subjects and phrases have been queried over a specific period of time. Google Trends works by analyzing a portion of Google searches to compute how many searches have been done for the terms entered, relative to the total number of searches done on Google over the same time. That is Google Trends provide the top 10 related searches over time as well as the 10 rising related searches on which interest has increased recently. This allows us to gather up to 20 related terms for one given term.

Yahoo Clues basically gives you insight into the types of people searching for specific keyword phrases and shows related terms based on those searches and searchers. The tool allows you to plug in one or two keyword phrases and it then plots the search trends of those keywords on the page. It shows you keyword popularity over time, searches by age and gender, income level, geographic location, "search flow" and related searches. It offers the search flow, the terms requested just before and after a term. It also provides set of related searches.

We define four sets of words built from URL url: REL_{rd}(url); REL_{rem}(url); AS_{rd}(url) and AS_{rem}(url). REL_{set}(url) consists of all the words related to the words of set that is words included in terms that are results of requests for elements of set. Here set is either RD_{url} or REM_{url}.

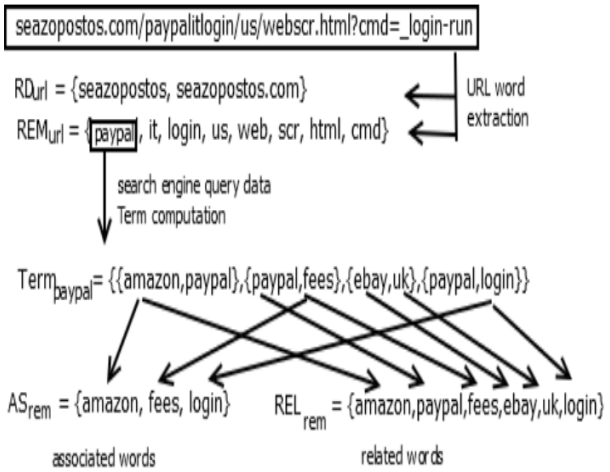

Fig. word extraction and word set composition

3.4 Feature Computation

In the table features 1-6 define intra-URL relatedness by calculating the Jaccard index pair wise between the four sets

	Features	Description
1	$J_{RR} = \frac{\lvert REL_{rd}(url) \cap REL_{rem}(url) \rvert}{\lvert REL_{rd}(url) \cup REL_{rem}(url) \rvert}$	Jaccard index b/w $REL_{rd}(url)$ and $REL_{rem}(url)$
2	$J_{RA} = \frac{\lvert REL_{rd}(url) \cap AS_{rem}(url) \rvert}{\lvert REL_{rd}(url) \cup AS_{rem}(url) \rvert}$	Jaccard index b/w $REL_{rd}(url)$ and $AS_{rem}(url)$
3	$J_{AA} = \frac{\lvert AS_{rd}(url) \cap AS_{rem}(url) \rvert}{\lvert AS_{rd}(url) \cup AS_{rem}(url) \rvert}$	Jaccard index b/w $AS_{rd}(url)$ and $AS_{rem}(url)$
4	$J_{AR} = \frac{\lvert AS_{rd}(url) \cap REL_{rem}(url) \rvert}{\lvert AS_{rd}(url) \cup REL_{rem}(url) \rvert}$	Jaccard index b/w $AS_{rd}(url)$ and $REL_{rem}(url)$
5	$J_{ARrd} = \frac{\lvert AS_{rd}(url) \cap REL_{rd}(url) \rvert}{\lvert AS_{rd}(url) \cup REL_{rd}(url) \rvert}$	Jaccard index b/w $AS_{rd}(url)$ and $REL_{rd}(url)$
6	$J_{ARrem} = \frac{\lvert AS_{rem}(url) \cap REL_{rem}(url) \rvert}{\lvert AS_{rem}(url) \cup REL_{rem}(url) \rvert}$	Jaccard index b/w $AS_{rem}(url)$ and $REL_{rem}(url)$
7	$card_{rem} = \lvert REM_{url} \rvert$	number of words in REM_{url}
8	$ratio_{Arem} = \frac{\lvert AS_{rem}(url) \rvert}{\lvert REM_{url} \rvert}$	ratio of associated words for words in REM_{url}
9	$ratio_{Rrem} = \frac{\lvert REL_{rem}(url) \rvert}{\lvert REM_{url} \rvert}$	ratio of related words for words in REM_{url}
10	$mld_{res} = \begin{cases} 0 & \text{if } \lvert Term_{mld} \rvert = 0 \\ 1 & \text{else} \end{cases}$	whether there is search engine results or not for the *mld* of the URL
11	$mld.ps_{res} = \begin{cases} 0 & \text{if } \lvert Term_{mld.ps} \rvert = 0 \\ 1 & \text{else} \end{cases}$	whether there is search engine results or not for the *mld.ps* of the URL
12	*ranking*	Alexa ranking for *mld.ps*

Fig. Feature Computation

(REL_{rd}(url), REL_{rem}(url), AS_{rd}(url) and AS_{rem}(url)). The Jaccard index is a long-established metric used to calculate similarity and diversity between two sets A and B. The closer J (A, B) is to 1 the more similar are A and B.

These six features quantify the relatedness between the two parts of the URL (mld.ps and the rest) through J_{RR}, J_{RA}, J_{AA} and J_{AR}, as these compute Jaccard indexes between sets extracted from different parts (RD_{url} and REM_{url}). These also measure the relatedness inside each part through J_{ARrd} and J_{ARrem}, as these features are calculated from sets extracted from the same part of a URL.

Features 7-12 reflect the popularity of a URL and its components with the number of words that compose it ($card_{rem}$) and the number of related and associated words found in search engine query data based on these words with $ratio_{Arem}$ and $ratio_{Rrem}$. Features 10 and 11 represent the popularity of the registered domain by giving Boolean values describing whether the mld.ps and mld match results while queried in Google Trends and Yahoo Clues.

IV. PHISHSTORM IMPLEMENTATION

PhishStorm gives a generic solution for phishing URL detection relying on intra-URL relatedness computation. This technique only needs access to search engine query data to operate. However, as the main vector of phishing attacks is spoofed emails embedding phishing URLs, we implement PhishStorm as a centralized phishing email detection tool positioned in front of the email server. Nowadays, spam filtering is performed centrally in many organizations and PhishStorm can be added to such process to increase detection performance.

Figure depicts the implementation of PhishStorm and the four steps of the phishing email detection process. While incoming emails from the Internet reach PhishStorm:

1. Potential embedded URLs are extracted there from. The system then proceeds to features computation thanks to search engine query data and predicts a phishingness score using machine learning techniques
2. A detection threshold is applied to every predicted score, determining if the email must be forwarded to the email server
3. and then to user
4. with its phishingness score or dropped.

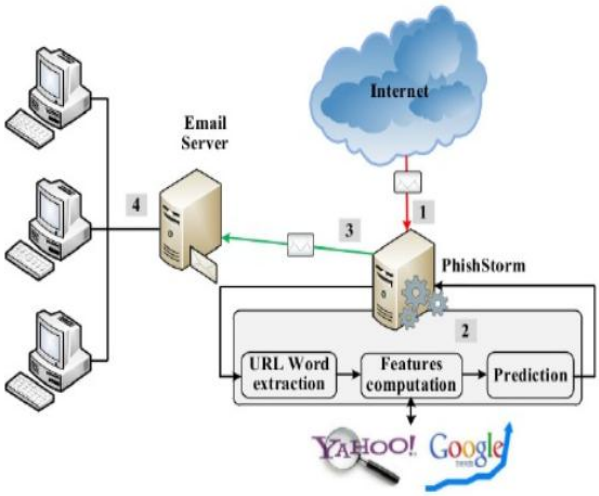

Fig. PhishStorm Implementation

V. CONCLUSION

We introduce PhishStorm, an efficient phishing URL detection system which is based on intra-URL relatedness. It reflects the relationship among words into a URL, specifically part of the URL that can be freely defined and registered domain. We can extract the 12 features by using search engine query data which define its popularity.

REFERENCES

[1] P. Prakash, M. Kumar, R. Kompella, and M. Gupta, "PhishNet: predictive black-listing to detect phishing attacks," in Proceedings of INFOCOM.IEEE, 2010, pp. 15.

[2] L. Bilge, S. Sen, D. Balzarotti, E. Kirda, and C. Kruegel, "Exposure: A passive DNS analysis service to detect and report malicious domains," vol. 16, no. 4, 2014, pp. 128.

[3] V. Ramanathan and H. Wechsler, "PhishGILLNET phishing detection methodology using probabilistic latent semantic analysis, AdaBoost, and co-training," EURASIP Journal on Multimedia and Information Security, vol. 2012, no. 1, pp. 122, 2012.

[4] L. Cao, T. Probst, and R. Kompella, "PhishLive: A view of phishing and malware attacks from an edge router," in Proceedings of the 14th International Conference on Passive and Active Measurement - PAM. Springer-Verlag, 2013, pp. 239.

