2012

PhishLive: A View of Phishing and Malware Attacks from an Edge Router

Lianjie Cao
Purdue University

Thibaut Probst
INSA de Toulouse, probst@etud.insatoulouse.fr

Ramana Kompella
Purdue University, rkompella@purdue.edu

Report Number:

12-007

PhishLive: A View of Phishing and Malware Attacks from an Edge Router

Lianjie Cao
Purdue Univeristy
West Lafayette, IN, USA
cao62@purdue.edu

Thibaut Probst
INSA de Toulouse
Toulouse, France
probst@etud.insa-toulouse.fr

Ramana Kompella
Purdue University
West Lafayette, IN, USA
rkompella@purdue.edu

ABSTRACT

Malicious website attacks, including phishing, malware, and drive-by downloads have become a huge security threat to today's Internet. Various studies have been conducted to explore approaches to prevent users from being attacked by malicious websites. However, no studies to date exist on the prevalence of and temporal characteristics of such traffic. In this paper, we developed the PhishLive system to study 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). During our experiment of one month, we analyzed over 1 Billion URLs. Our analysis reveals several interesting findings.

1. INTRODUCTION

The rapid development of the Web over the recent few decades has made the Internet a hotbed for a wide range of criminal activities. Numerous types of attacks are hidden behind HTTP connections such as phishing, cross-site scripting, malware, and botnet attacks. The most commonly used solution to defend against such attacks is using blacklisting. A blacklist-based defense system contains a set of uniform resource locators (URLs) that are identified as malicious or suspicious, either through a human-vetting process or using other such mechanisms. When users or software are trying to connect to such web pages, these systems will pop out warnings or block the web page directly. For example, most modern browsers such as Mozilla Firefox, Apple Safari, Internet Explorer warn users of accessing malicious websites for phishing attacks.

Literature is ripe with several studies that focused on documenting the effectiveness of such browser-based techniques in thwarting phishing attacks. For example, [5] discusses the effectiveness of passive and active warnings to users. Similarly, [20] studies the efficacy of different anti-phishing tools. There also exist several papers (e.g., [13, 10, 14, 6]) proposing different solutions for improving the attack detection and defense using enhanced blacklisting techniques. Other content-based techniques have also been proposed (e.g., [21, 4, 15, 7, 12, 16]) for detecting phishing.

Unfortunately, to date, there exists few studies that focus on understanding temporal characteristics of phishing or malware accesses in an edge network such as a campus or an enterprise network comprising of a few 10s of thousand users. For example, to the best of our knowledge, there exists no studies that clearly indicate what fraction of URL accesses in a given campus or edge network comprises phishing or malware hosting sites (together referred to as malicious sites). Similarly, it is not clear whether malicious sites are accessed just once, or a few times, or are repeatedly accessed over time across users, and whether these are hidden between HTTP redirects. While studies such as [6] exist, they are mainly about the user interaction with phishing websites. We believe a study to answer such questions is important for many reasons: First, it can help in sizing the resource requirements of security middleboxes that can be deployed to defend against them. Second, the temporal characteristics can help generate insights to inform future defense mechanisms.

In this paper, we focus on studying and understanding the characteristics of HTTP accesses to malicious sites as seen by the edge router of a large campus network comprising upwards of 50,000 users. A key requirement for our study is the ability to identify whether a given access is to a malicious site or not, for which, we leverage existing blacklisting tools such as the Google Safe Browsing (GSB) back-end server. Thus, we do not invent any new mechanism for detecting phishing attacks, but merely use existing techniques to continuously monitor the network for malicious accesses to phishing/malware websites. Our system called 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, geolocation distribution of attacking IP addresses, attacking hostnames clustering and malicious redirect chain analysis.

We deployed PhishLive on the edge router of a large university for about a month in which we captured and verified about 1 Billion URLs. Some of our key findings are as follows:

- The fraction of URLs that are identified as belonging to phishing/malware sites is relatively small; in our data, it is less than 0.038% of all URLs.
- There is a relatively higher number of malicious URLs accessed during 11:00pm-5:00am compared to other times.
- Most domains (almost 50%) typically existed for less than 1 day. However, close to 10% of the domains were accessed for more than 15 days.
- An extremely small fraction of all HTTP redirection chains contain malicious URLs; in our data less than 2,000 URLs are part of redirection chains out of about 50 million redirection chains.

The remainder of this paper is organized as follows. Section 2 gives an overview of the PhishLive system. In Section 3, we outline our major analysis from the deployment of PhishLive at the university edge router.

2. SYSTEM OVERVIEW

In this section, we describe the design of our system called PhishLive for monitoring the prevelance of and characterizing HTTP traffic for phishing and malware attacks. We envision PhishLive to be deployed at an edge router, such as a campus gateway router, that can track the various HTTP requests issued by a bunch of users. We assume the presence of a standard high-speed capture device (e.g., Endace 10Gbps monitoring card) to collect each packet that is going through the gateway router to the outside world, from which we filter the HTTP traffic (port 80) and extract the URLs from HTTP requests. For verifying whether a given URL is malicious or not, each URL is cross checked with the Google Safe Browsing (GSB) database [1]. Since HTTP redirects are also used to hide malicious content [17, 18, 9] in some attacks, the system is designed to also track and analyze HTTP redirect chains. While access to the outgoing sequence of URL requests is generally sufficient, analyzing redirect chains require parsing the HTTP responses as well which means the PhishLive system requires access to both sides of traffic.

The PhishLive system comprises three components: a capture module, a check module and an update module, that are shown in Figure 1. We describe the functionalities and implementations of these three components in the remainder of this section.

2.1 Capture Module

The capture module of our system utilizes the libpcap library [3] to capture the HTTP requests from the hosts inside the university and redirect responses from

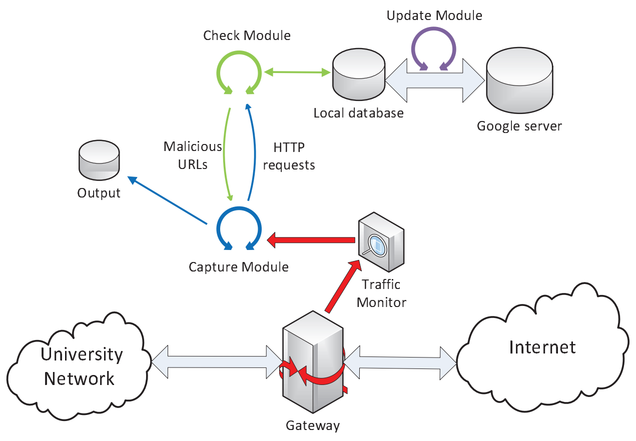


Figure 1: PhishLive system architecture.

the external hosts. As of now, the PhishLive system supports up to 5 types of HTTP requests: GET, HEAD, POST, PUT, DELETE, and 4 types of HTTP redirect responses: 301, 302, 303, 307. The capture module also uses network libraries and regular expressions to extract the URLs from HTTP requests or the URLs in redirect responses, as well as source IP addresses, destination IP addresses, source port number and destination port number. In order to protect the privacy information of users, all user-facing IP addresses are hashed. Each request URL is forwarded to the check module in the form of “SIP DIP URL”.

As part of our analysis, we also wish to study the role of HTTP redirects in phishing and malware attacks. A redirect chain is a sequence of URLs starting from the first requested URL, ending with the last requested URL, that can be represented as follows: $GET(URL_1)$ $\rightarrow REDIRECT(URL_2)$ ... $\rightarrow GET(URL_n)$, where n is the number of different requested URLs in the chain. Although it appears simple, it is a little tricky to track HTTP redirects in an online fashion, since it requires correlation across TCP connections (since each GET request is to a different hostname). Thus, in PhishLive, we build two hash tables (denoted as level-1 hash table and level-2 hash table in Figure 2) to store HTTP redirects. The level-1 hash table holds related information of a HTTP request with a key of $\langle SIP, SP, DIP, DP\rangle$, where SIP/DIP are the source/destination IP addresses and, SP/DP are the source/destination port numbers. Because the request and the redirect packet belong to the same TCP session, when a HTTP redirect response is captured, the capture module checks if there is an existing record in the level-1 hash table with the same key (we just need to reverse the source and destination addresses and ports). If a match is found, it means that this redirect is the response for the matched HTTP request. Then this pair (HTTP request and redirect response) is extracted from the level-1 hash table and inserted into the level-2 hash table with the key of "SIP".

The level-2 hash table keeps record of all HTTP redirect chains observed. The data structure of a redirect chain in the level-2 hash table is a linked list. Each slot in the level-2 hash table corresponds to a user-facing IP address (inside the network). When inserting a pair (HTTP request and redirect response) into the level-2 hash table, it checks if the URL in the HTTP request matches the URL in the last record with the same key. A match means the current request is the HTTP request of the URL in the last redirect response. Therefore, the redirect response is attached at the end of the corresponding redirect chain, if the chain exists. If no existing chain is found or if the URL in the current HTTP request does not match any existing URL, a new redirect chain is built with the current HTTP request as the head and the current redirect response as the second node. Therefore, one slot in the level-2 hash table may contain more than one redirect chain. For instance, slot n in the level-2 hash table includes redirect chains a→b→c and x→y. When a new pair of HTTP redirect y→z is inserted, it searches for the first linked list a→b→c and finds that y does not match c. Then it moves to next linked list x→y. Since URL y is a match, the new pair y→z is attached to the linked list. The second chain becomes x→y→z. The operations of the two hash tables are illustrated in Figure 2 .

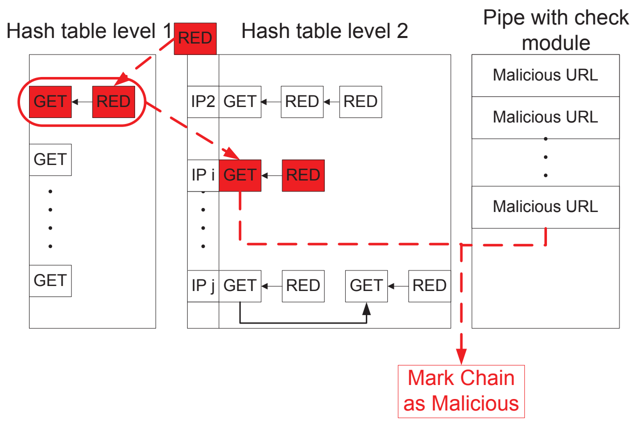


Figure 2: Operations on two hash tables.

The capture module also receives feedback which includes the malicious URL and the victim's IP address from the check module. It then compares the malicious URL with the records in the level-2 hash table. If the URL is found in the level-2 hash table, it will be marked as malicious and dumped to a file later on. The implementation of the capture module constitutes of three threads: one thread captures and extracts URLs from HTTP packets and feeds them to check module; another thread receives results from check module and scans level-2 hash table for a match; the last thread refreshes the two hash tables periodically to prevent them from growing too large and a fatal memory drop-off.

2.2 Check Module and Update Module

The check module is based on the PHP API of Google Safe Browsing database provided by Google. The check module maintains a local database of malicious URLs verified by GSB server and interacts with the capture module through two pipes. Once it receives a URL from capture module, it checks the URL against the local database and feeds it back to capture module through a pipe if the URL is identified as malicious. The check module also produces general real-time statistics. The update module updates the local database with Google server periodically to ensure that the content of the local database is up-to-date.

3. EXPERIMENTAL RESULTS

We deployed the PhishLive system at the edge router of a large university network over 30 days from March 19, 2012 to April 19, 2012, during which the system analyzed more than 1 Billion HTTP requests (as summarized in Table 1). Out of the 1 Billion URLs, only about 0.0381% of all HTTP requests were classified as phishing requests. We also observed about 50 million HTTP redirect chains out of which only about 7,500 included malicious URLs.

Experiment Duration	3/19/12-4/19/12
HTTP Requests	1,038,803,540
Malicious URLs	395,671 (0.0381%)

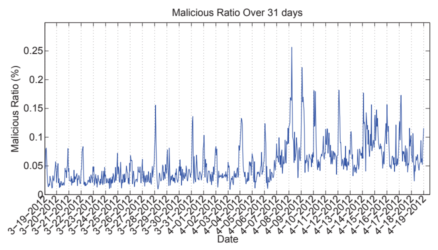
Table 1: Statistics of the Experiment.

Since PhishLive system only captures the HTTP requests from hosts and HTTP redirect responses from servers and verifies the URL by querying GSB database, the accuracy of the dataset drawn from the experiment largely depends on the accuracy of the GSB database. Previous studies [19] indicate that GSB database has a false negative rate of less than 10%; so we believe that the results are more or less accurate.

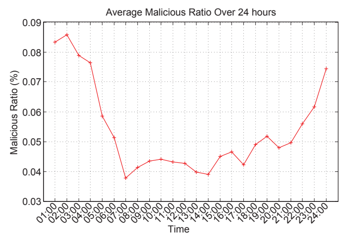
3.1 Temporal and Location Analysis

The PhishLive system computes the number of HTTP requests and number of malicious URLs observed per hour. Host users behaves very differently at different time. For instance, users usually make more HTTP requests during daytime than night. So, it is possible that more malicious website attacks may be observed during daytime. However, it does not necessarily means that the attacks are more active during daytime. Therefore, we present our result in terms of the ratio of the number of malicious attacks to the total number of HTTP requests, which is defined as the *malicious ratio*.

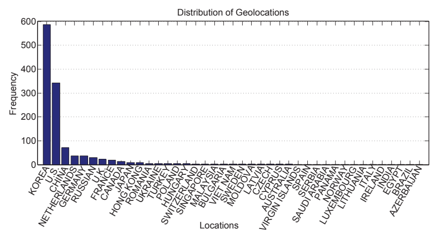
Figure 3(a) shows the malicious ratio over the entire



(a) Malicious ratio



(b) Time of day



(c) Geolocation of IP addresses

Figure 3: Prevalence and location of malicious websites in our dataset. Subplot (a) shows the ratio of malicious URLs to benign over the month. (b) shows per-hour average malicious ratio. (c) shows the geolocation of attacker domains using IP Addresses.

experiment. We notice that the value of the malicious ratio varies from 0.008% to 0.257% for different times. But, generally, the malicious ratio is relatively higher at night compared to daytime. From Figure 3(b) which displays the average malicious ratio for 24 hours, we observe that the malicious ratio is significantly higher from 11 PM to 5 AM. During this time, it appears hosts are more likely to be visit malicious websites. It is also possible that nightly accesses are likely initiated by automatic actions such as malwares and bots.

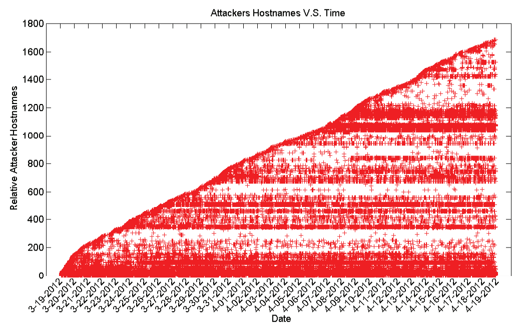
We also studied the geographical location information of the 1231 distinct IP addresses of the attackers by utilizing the service of IPInfoDB [2]. From the results in Figure 3(c), we can see that these IP addresses come from 41 countries, but only 10 countries have more than 10 IP addresses. Among all these countries, Korea covers 47.6% and United States covers 27.8% of the IP addresses. Of course, we cannot generalize this to other networks, but the fact that US itself hosts such a significantly high fraction of malware is kind of surprising.

3.2 Access Characteristics of Victims

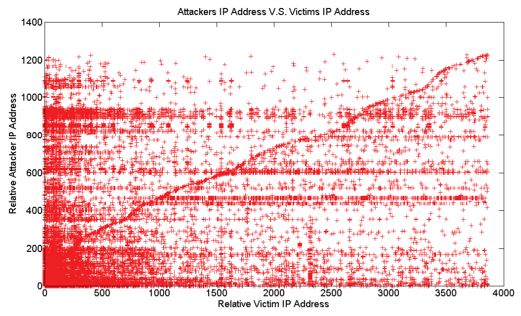
We now analyze the malicious URL access characteristics. Specifically, we focus on the timing of the user accesses to attacker domains (IP addresses) and the relationship between victims (IP addresses) and attacker domains.

Figure 4(a) displays the timing characteristic of when the attacker domains have been accessed by users. The y-axis represents distinct hostnames of attackers we observed in our data, while the x-axis is the date, with a resolution of one day. We can see that there are approximately three types of attacker domains. Type I attacker domains are those that users access frequently over a long period of time; such domains are appear as a horizontal line in the figure. Attacker domains of type II are those that may be intermittently accessed by users. They appear as dashed horizontal line in the figure. Type III attackers scatter attacks infrequently, and mostly appear as sparse points in the figure.

Figure 4(b) shows the scatter plot between victims and attacker domains. From the figure, we can see that a significant number of victims seem to have contacted a few popular attacker IP addresses. Similarly, vertically, a single user seems to have contacted many different attacker IP addresses as well.



(a) Attacker hostnames

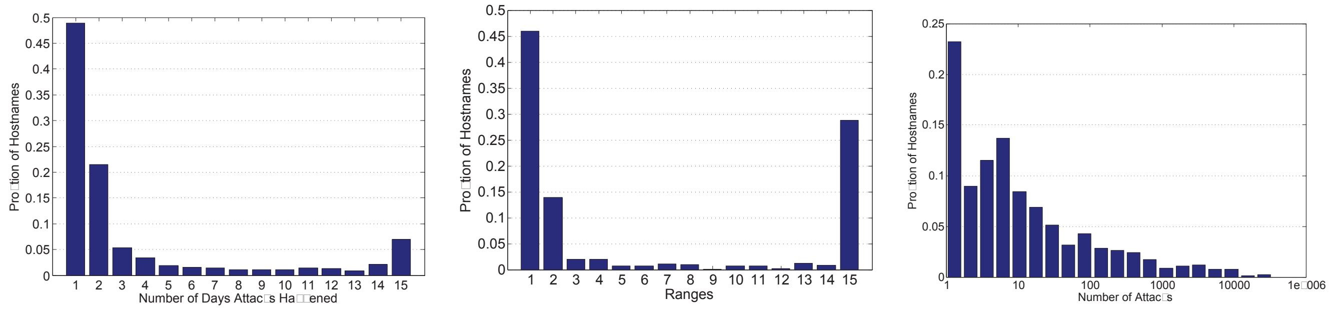


(b) Victim Attacker scatter plot

Figure 4: Behavior of attacker's hostnames over time.

3.3 Persistence of Domain Names

The previous graphs used IP addresses; since a single IP address could host many different domains, we now switch to understanding the persistence characteristics of the various domain names we observed. In Figure 5(a), we plot a histogram of the number of days a particular domain name was observed in our dataset. The x-axis is the number of days an attack domain was



(a) # of Days (b) Range of Days (c) # of HTTP accesses

Figure 5: Temporal characteristics of malicious accesses. Subplot (a) shows distribution of domains in terms of number of days a domain has been accessed. (b) shows the distribution of range, and (c) shows distribution of number of accesses.

observed, while the y-axis shows the fraction of attacks. Since we have collected only one month's trace, we restrict ourselves to the URLs collected in the first 15 days, so each domain has the same chance of appearing in the next 15 days from the first time a domain appears. In other words, we eliminate the fringe bias in our data set by focusing on domains seen in the first 15 days. In Figure 5(a), we found that almost 50% of the attack domains were present for less than 1 day, which confirms the fast-flux like behavior observed in previous studies [8]. But there are a non-trivial number of domains that were accessed persistently for a number of days. Close to 10% of the domains were accessed for the whole period of 15 days; due to lack of data beyond, we cannot conclusively determine how long these campaigns persisted.

A similar behavior can be observed in Figure 5(b), where we plot the range of the days between which we observed the domain. For example, if a domain was seen on day x, and we observe the domain last on day y, within the $x + 15$ days, we categorize this domain as having a range of $y - x$ days. While this plot (in Figure 5(b) has similar characteristics as the previous one (Figure 5(a)), the bar for 15 days is significantly taller (almost 30% compared to only 5%). Note that we included all domains that were active for greater than 15 days in the 30% corresponding to $x = 15$. This shows that even though the number of domains that were active for all 15 days was small (<10%), a significant number of domains were active for a much amount of time (>15 days). Finally, we plot the distribution of the number of HTTP accesses to particular domains in Figure 5(c). From this figure, we can observe that most domains are accessed relatively infrequently. Almost 70% of domains were seen only less than 10 times in our dataset, and 90% of domains were seen less than 100 times in our dataset. A small number of domains seem to be accessed a lot of times, almost as high as about 33,000 times.

3.4 Lexical Similarity of Domains

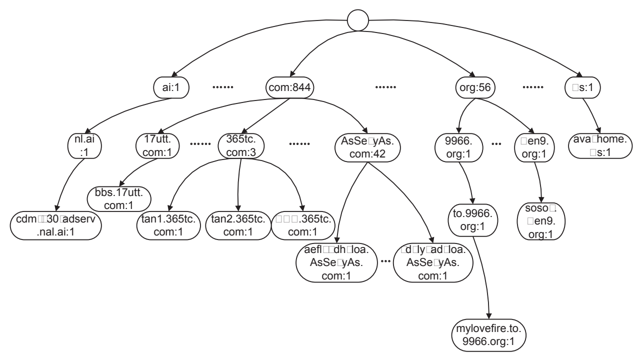


Figure 6: Tree of the captured malicious hostnames.

We now study the lexical similarity in attack domains we have observed in our data set. We decompose a malicious URL into three components: hostname, file path and query string, out of which our main interest is just the hostname. We lexically group together hostnames that share some similarity in the form of a tree as shown in Figure 6. All the 395,671 malicious URLs we observe belong to 1686 distinct hostnames, 37 of which are IP addresses that we exclude in the clustering process. We first split hostnames splits hostnames to several tokens, i.e., is a string between two dots. For example, we break pann.nate.com into pann, nate and com tokens. Then we start to build the tree in the reverse order, starting from the top level domain name (e.g., com, nate and then pann). Each node in the tree is labelled $T : n$ where T consists of a prefix and n is the total number of hostnames that match the prefix.

From the tree, we observed that some prefixes exhibited large subtrees. For example, in our data, we observed PassingGas.net shared by 42 hostnames that differed mainly in the third-level token, such as nealyxadxloa.PassingGas.net, hccayxadxloa.PassingGas.net and

Hostname	Number of Attacks	First Attack	Last Attack%
nealyxadxloa.PassingGas.net	21	3-19-2012 10:45:41	3-19-2012 19:17:11
hccayxadxloa.PassingGas.net	95	3-19-2012 19:42:33	3-20-2012 19:26:35
hmcayfadxeoa.PassingGas.net	9	3-20-2012 06:33:53	3-20-2012 06:33:55
dqzoyxadxloa.PassingGas.net	66	3-20-2012 19:41:14	3-21-2012 19:05:38

Table 2: Changing third-level token in the malicious URL's hostname.

so on. We can observe that their third-level domain names look like they have been generated randomly; this discovery is not surprising as we observe such occurences even in public domain phishing blacklists such as PhishTank. What was interesting, however, is that each unique hostname was accessed for no more than 2 days by hosts in our network, as the Table 2 indicates; such information cannot be obtained by observing blacklists such as PhishTank alone. The old hostnames became invalid, leading to a page indicating that this website is using Sitelutions Redirection Engine and the URL is either entered incorrectly or has been removed by itself. Another feature of these hostnames is that they all share the same IP address (located in Herndon, VA) no matter how they change the third level domain name. Obviously, this indicates that the attackers are manipulating the DNS Resource Records dynamically, which is not surprising as attackers often try to evade detection this way as previous studies indicated (e.g., [13, 11]).

3.5 Redirect Chain Analysis

Researchers in [9] reported that attackers may use long redirect chains to hide malicious content; we therefore study whether redirections are actively used in attacks today. In our dataset, we observed a total of 7,497 redirect chains that contained at least one malicious URL. However, in some of the redirect chains, the original HTTP requests and redirect response belong to the same hostname. If a redirect chain is created by attackers intentionally, then: (i) the URL in the redirect response typically belongs to a different hostname; (ii) the last redirect is malicious; (iii) the redirect chain usually contains more than one redirect. Therefore, we define a redirect chain to be a *effective malicious redirect chain* if it satisfies the first two requirements above.

Note that the third requirement may not be applied to all attacks. For instance, the chain consisting of redirecting http://www.dwnews.com/images/news/blog.gif → http://www.dwnews.com/ is not effective malicious redirect chain since hostname is the same. However, the chain that involves the following redirection, http://grannymovs.in/ → http://lotaz.in/MyTRAFF/apiLINKda.php → http://servantspywarekeep.info/755063395c4a385d/ is an effective malicious redirect chain.

We also noticed a special case that the redirect chains caused by the expiration of the hostnames mentioned before. Redirects related to the expiration of those hostnames occurred 2,065 times, significant enough to be ignored. Among all the 7,497 malicious redirect chains, we identified 1,449 effective malicious redirect chains. The statistics of those effective malicious redirect chains are summarized in Table 3.

Type	**Number**
Total Redirect Chains	50,204,174
Malicious Redirect Chains	7,497
Effective Malicious Chains	1449
Average Number of Redirect	2.221
Number of Chains Longer Than 1	246
Max Number of Redirect	5
Start with Normal Request	988
301 Redirect	231
302 Redirect	1523
303 Redirect	6
307 Redirect	0

Table 3: Statistics of Effective Malicious Redirect Chain

Several conclusions can be derived from our analysis on malicious redirect chains: (1) The number of malicious redirect chains (7497) among all redirects ($\approx$ 50 million) is quite small (<0.0149%). (2) Only a small portion of malicious redirect chains (246 out of 7497) are effective malicious redirect chains and contain more than 1 redirect (about 3.29%) in our experiment. (3) Most of the effective malicious chains (about 988 out of 1449) start from a normal HTTP request and end up with a malicious URL as redirect response.

4. CONCLUSIONS

To date, no studies exist on how prevalent phishing/malware attacks are or on the temporal characteristics of malware accesses in edge networks. We designed the PhishLive system for long-term monitoring of HTTP traffic of a large campus network that enabled us to study various temporal characteristics of phishing/malware attacks. Using a month-long deployment of the PhishLive system at the university gateway router, we observed many interesting characteristics of phishing attacks. For example, we found that malicious accesses are more common during 11:00-5:00pm than during day times. Similarly, we found that most domains appeared only for one day and redirection was not common among many of the malware URLs we detected.

5. REFERENCES

[1] Google safe browsing api. https://developers.google.com/safe-browsing/.

[2] Ipinfodb. http://www.ipinfodb.com/.

[3] libpcap. http://www.tcpdump.org/.

[4] U. Bayer, I. Habibi, D. Balzarotti, E. Kirda, and C. Kruegel. A view on current malware behaviors. In *Proceedings of the 2nd USENIX conference on Large-scale exploits and emergent threats: botnets, spyware, worms, and more (LEET)*, pages 1–11, April 2009.

[5] S. Egelman, L. F. Cranor, and J. Hong. You've been warned: an empirical study of the effectiveness of web browser phishing warnings. In *Proceedings of the 26th annual SIGCHI conference on Human factors in computing systems (CHI)*, pages 1065–1074, April 2008.

[6] S. Garera, N. Provos, M. Chew, and A. D. Rubin. A framework for detection and measurement of phishing attacks. In *Proceedings of the ACM workshop on Recurring Malcode (WORM)*, pages 1–8, 2007.

[7] G. Gu, J. Zhang, and L. Wenke. BotSniffer: Detecting botnet command and control channels in network traffic. In *Proceedings of the 15th Annual Network and Distributed System Security Symposium (NDSS)*, February 2008.

[8] M. Konte, N. Feamster, and J. Jung. Dynamics of online scam hosting infrastructure. In *Proceedings of the 10th Passive and Active Measurement Conference (PAM)*, April 2009.

[9] S. Lee and J. Kim. Warningbird: Detecting suspicious urls in twitter stream. In *Proceedings of the 19th Annual Network and Distributed System Security Symposium (NDSS)*, pages 1–13, February 2012.

[10] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker. Beyond blacklists: learning to detect malicious web sites from suspicious urls. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD)*, pages 1245–1254, June 2009.

[11] D. K. McGrath and M. Gupta. Behind phishing: an examination of phisher modi operandi. In *Proceedings of the 1st Usenix Workshop on Large-Scale Exploits and Emergent Threats (LEET)*, pages 1–8, April 2008.

[12] R. Perdisci, W. Lee, and N. Feamster. Behavioral clustering of http-based malware and signature generation using malicious network traces. In *Proceedings of the 7th USENIX conference on Networked systems design and implementation (NSDI)*, April 2010.

[13] P. Prakash, M. Kumar, R. Kompella, and M. Gupta. Phishnet: Predictive blacklisting to detect phishing attacks. In *Proceedings of the 29th Conference on Computer Communications (INFOCOM)*, pages 1–5, March 2010.

[14] A. Ramachandran, N. Feamster, and S. Vempala. Filtering spam with behavioral blacklisting. In *Proceedings of the 14th ACM conference on Computer and communications security (CCS)*, October 2007.

[15] C. Rossow, C. J. Dietrich, H. Bos, L. Cavallaro, M. van Steen, F. C. Freiling, and N. Pohlmann. Sandnet: network traffic analysis of malicious software. In *Proceedings of the 1st Workshop on Building Analysis Datasets and Gathering Experience Returns for Security (BADGERS)*, pages 78–88, New York, NY, USA, April 2011.

[16] C. Song, J. Zhuge, X. Han, and Z. Ye. Preventing drive-by download via inter-module communication monitoring. In *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security (ASIACCS)*, pages 124–134, New York, NY, USA, April 2010.

[17] S. Webb, J. Caverlee, and C. Pu. Introducing the webb spam corpus: Using email spam to identify web spam automatically. In *Proceedings of the 3rd Conference on Email and Anti-Spam (CEAS)*, July 2006.

[18] S. Webb, J. Caverlee, and C. Pu. Characterizing web spam using content and http session analysis. In *Proceedings of the 4th Conference on Email and Anti-Spam (CEAS)*, July 2007.

[19] C. Whittaker, B. Ryner, and M. Nazif. Large-scale automatic classification of phishing pages. In *Proceedings of the 17th Annual Network and Distributed System Security Symposium (NDSS)*, February 2010.

[20] Y. Zhang, S. Egelman, L. Cranor, and J. Hong. Phinding phish: Evaluating Anti-Phishing tools. In *Proceedings of the 14th Annual Network and Distributed System Security Symposium (NDSS)*, February 2007.

[21] Y. Zhang, J. I. Hong, and L. F. Cranor. Cantina: a content-based approach to detecting phishing web sites. In *Proceedings of the 16th International World Wide Web Conference (WWW)*, pages 639–648, May 2007.