PhishCage: Reproduction of Fraudulent Websites in the Emulated Internet

Daisuke Miyamoto
Information Technology Center
The University of Tokyo
2-11-16 Yayoi, Bunkyo-ku, Tokyo, JAPAN
daisu-mi@nc.u-tokyo.ac.jp

Yuzo Taenaka
Information Technology Center
The University of Tokyo
2-11-16 Yayoi, Bunkyo-ku, Tokyo, JAPAN
taenaka@nc.u-tokyo.ac.jp

Toshiyuki Miyachi
StarBED Technology Center
National Institute of Information and Communications Technology
2-12 Asahi-dai, Nomi, Ishikawa, JAPAN
miyachi@nict.go.jp

Hiroaki Hazeyama
Graduate School of Information Science
Nara Institute of Science and Technology
8916-5 Takayama, Ikoma, Nara, JAPAN
hiroa-ha@is.naist.jp

ABSTRACT

This paper introduces PhishCage, an experimental infrastructure for phishing detection systems. Due to the short life time of phishing sites, comparative study of effectiveness, especially universality, among the detection systems is difficult. Our key idea is developing a testbed in which preserved phishing sites can be browsed as if they existed in the wild. According to our survey for phishing detection systems, this paper defines the requirements for the testbed, and designs PhishCage to meet with the requirements. The experiment of PhishCage demonstrates our mapping algorithm for 121 phishing sites into the emulated Japanese Internet topology. We confirm that phishing detection systems can obtain the realistic IP address and autonomous system number of the phishing sites in PhishCage, and few modifications enable the systems to work as if they are in the real Internet. With regard to the experimental results, we analyze the limitation of PhishCage, and finally discuss the feasibility of our emulation technique.

Categories and Subject Descriptors

D.2.1 [**Software Engineering**]: Requirements/Specifications - Tools

General Terms

Experimentation, Security

Keywords

Phishing detection, Testbed, Emulation

EMUTools Workshop 2013, March 05-07

DOI 10.4108/icst.simutools.2013.251707

1. INTRODUCTION

Phishing is a fraudulent activity defined as the acquisition of personal information by tricking an individual into believing the attacker is a trustworthy entity. Phishing attackers lure people by using a phishing email, as if it were sent by a legitimate corporation. The attackers also attract the email recipients into a phishing site, which is the replica of an existing web page, to fool a user into submitting personal, financial, and/or password data.

To deal with phishing attacks, a heuristic-based detection method has begun to garner attention. A heuristic is an algorithm to identify phishing sites based on users' experience, and checks whether a site appears to be a phishing site or not. Based on the detection result from each heuristic, the heuristic-based solution calculates the likelihood of a site being a phishing site and compares the likelihood with the defined discrimination threshold.

A current challenge of the heuristics-based solutions is increasing detection accuracy of phishing sites. Due to that the accuracy in the existing heuristic-based solutions was far from suitable for practical use [24], many researchers have attempted to both develop innovative heuristics and refine the calculation algorithms of the likelihood.

However, it is not easy to confirm the effectiveness, especially universality, of their contributions. Due to that the phishing sites tends to have short life time period, each research employed its own unique dataset, which was composed of phishing sites observed during distinct time period; it hinders us to compare the detection accuracy of the phishing sites among the detection methods. Further, modern methods cannot be tested in the past phishing sites. It might be difficult to judge whether the methods are useful only for the modern phishing sites or not - the detection methods have some universality.

This paper aims at constructing an experimental infrastructure, named PhishCage, toward the evaluation of the phishing detection methods. The key idea is the reproduction of the past phishing sites in the realistic network emulation testbed. We design and implement a crawler to preserve phishing content in the database, and reproduce the phish-

ing sites based on the Internet emulation methods [9]. Due to that the methods allows experiments to keep their realism in regard to the facility of the testbed, the phishing content are not only able to browse as if the website exists, but also delivered from web servers which are almost the same as the real phishing server.

Based on these designs, we constructed our testbed by 471 of KVM instances on StarBED's eleven nodes. Our testbed emulated Japan Internet topology with realistic IP addresses, and 112 phishing sites were able to browse. The phishing servers were assigned the phishing sites' IP addresses and were located at the phishing sites' autonomous system, in all of which had been observed in the wild. Even if PhishCage had no connectivity to the real Internet, the phishing detection methods worked regardless of the difference between real Internet and emulated one.

The rest of this paper is organized as follows. Section 2 explains our related work, and section 3 describes the design of PhishCage. Section 4 demonstrates our experiment, and section 5 discusses the limitation. Section 6 shows our future work, and finally summarizes our contribution in section 7.

2. RELATED WORK

This section briefly summarizes our related work. We first describe the current challenges in security testbed, and explain the state of the art in phishing detection systems.

2.1 Security Testbed

A testbed is a common platform for security experiment since it allows observing of cyber attacks in a controlled manner. Testbed nodes are connected only to the controlled networks for keeping them physically isolated from the Internet. However, malwares captured in the wild often behave differently when they run in a testbed, since they are designed to be difficult to analyze.

There are many research contributions for obtaining realistic result. Giving controlled Internet access to the software runs on the testbed is one solution [2], although it has possibility to exert a bad influence to the wild. The other challenge is improving the realism for a security testbed. According Cavlet et al. [5], it should not be possible for software running on a testbed to easily detect that is running on the testbed. They also required that the environmental setup should be as close as possible those of typical equivalent environment in the real world. Miwa et al. improved realism by development of mimetic Internet which gives mimic information when malwares checked if it runs in a testbed [13]. In their environment, malwares are tricked as if they ran in the wild.

Security experiments for network systems also require the realism. For the experiments of the end-to-end network, Sanaga et al. emulated capacity of a network link, delay packets, and introduce packet loss [18]. Hazeyama et al. developed the emulated Internet which had a similar topology of the Internet for evaluating their Autonomous System(AS)-level traceback system [9]. Their idea was allocating one AS to one physical / virtual node. Emulation of the real Internet made a testbed increase realism, however, it was difficult to construct an experimental network which had same size, same facilities, and / or same characteristics of the real Internet. Therefore, some emulation techniques should be taken to solve trade-offs among the scale of a testbed.

2.2 Phishing Detection Systems

There are two distinct approaches for identifying phishing sites. One is URL filtering. It detects phishing sites by comparing the URL of a website where a user visits with a URL blacklist, which is composed of the URLs of phishing sites. Unfortunately, the effectiveness of URL filtering is limited. In 2007, the detection accuracy of URL blacklist-based systems was roughly 70% [24]. In 2009, Sheng et al. reported [19] that URL blacklists were ineffective when protecting users initially, as most of them caught less than 20% of phishing sites at hour zero. The rapid increasing of phishing sites hinders URL filtering to work sufficiently due to the difficulty of building a perfect blacklist.

The other approach is a heuristic-based method. There are many heuristics have been proposed that are categorized into several types as follows.

URL-based heuristics: The phishing sites' URLs and legitimate ones tend to differ. According to the heuristic-based systems [1, 6, 25], the length of the phishing URL is long, the URL uses IP address instead of fully qualified domain name (FQDN), it employs a similar or otherwise legitimate-sounding domain name, and it contains symbols such as an at-mark, many hyphens or dots.

The FQDN appeared in the URL also gives hints to the heuristics. Checking the life time duration of the issued website is well-known heuristic as most phishing sites' URL expired in short time span. Some heuristics observes the abuse of International Domain Name [11].

Whittaker et al. [22] proposed to analyze AS numbers to which the page's hosts correspond using the routing data. Remind that phishing sites are often hosted on botnets whereas the legitimate enterprises use reputable hosting services. They explained that AS numbers gave a more accurate picture of IP address association than looking at IP address subnets.

Content-based heuristics: The phishing content appears look-alike legitimate ones, but the heuristic finds the difference by analysis of the content. According to Neil et al. [6], many phishing sites have silly misspellings and / or grammatical errors. The suspicious hyperlinks, input forms, and the abuse of logo or trade marks are also analyzed.

Several researches [25] employed information retrieval techniques. When their heuristics attempt to identify phishing sites, they feed keywords extracted from the content, and they then check if the site appears the top 30 search results. Xiang [23] updated this heuristics by using name entity recognition techniques.

Visual Similarity-based detection systems [8, 12] compare the visual appearance between a suspected phishing site and a legitimate site which is spoofed. When the two sites are too similar, a phishing warning is raised. The heuristics monitor the overall visual appearance such as text pieces, styles, images, and estimate the similarity with comparison algorithm, e.g., Earth Mover Distance.

The rest of heuristics employ other information sources. Amir [10] monitored the websites' certificate and its issuer, and Netcraft Inc., proposed to use the popularity of the website [15]. Our past research [14] employed the history of users' past trust decision which is also described that labeling as a phish or not.

3. PHISHCAGE

In this paper, we propose PhishCage, an experimental in-

frastructure for phishing detection methods. As we mentioned in section 1, it might be difficult to evaluate both effectiveness and universality of the methods. This paper tackles to the problem based on the emulation techniques. Our key idea is the reproduction of the past phishing sites in the realistic network emulation testbed. Regardless of the short life span in phishing sites, PhishCage enables the methods to work for the preserved phishing sites.

3.1 Requirements

Essentially, the testbed must have realism, that is, the testbed has similar characteristics to the real Internet for evaluating phishing detection systems. From the view point of obtaining realistic results from the experiments, this section explores our requirements for the testbed.

As we explained in section 2.2, the current phishing detection methods require the website's URL, a domain name and its created date, IP address, and AS number in the cases of the URL-based heuristics. On the contrary, the content-based heuristics require web content; when the heuristics analyze a website, its whole contents are able to browse as if they exist. Further, PhishCage should equip a function of a search engine or its search results for heuristics that are based on information retrieval techniques.

Based on these considerations, we show our requirements for the testbed as follows. The requirement 1 is general requirements for the testbed. Our testbed must not have Internet connections. The reproduced phishing sites can be accessed by the experimenters. Testbed nodes have six roles, including AS router nodes, phishing server nodes, information server nodes, and finally a phishing detection node.

An AS router node represents an AS border router. The node is assigned own AS number, and interconnects to its neighbor ASes based on the real Internet topology. It also advertises realistic network routes to the neighbor ASes, and receives the routes from the neighbors.

A phishing server node hosts a phishing site. The node runs HTTP server to respond phishing content to the request from a phishing detection node. The phishing content can be browsed as if the website exists. The IP address of the node is as same as that in the wild.

An information server node provides information for a phishing detection node. It provides the name resolution function and responses the realistic IP address. It also stores the check results of heuristics such as WHOIS and search results in the wild, and responses to the heuristics runs in the test environment.

A phishing detection node runs a phishing detection system to check the phishing site hosted on phishing server nodes. When the site employs FQDN in its URL, the node queries to a DNS server which runs on the information server node. If a heuristics needs to check the creation date of the FQDN, it also asks to the information server node.

3.2 Design

Due to the nature of this study, the reproduced phishing sites must be accessed only by the experimenter. Hence, we decide that the experimental network is quarantined from the Internet.

The study also employs an AS-level aggregation method for the emulation of the phishing sites, because of the limited numbers of the usable nodes in network emulation testbeds. The aggregation is that AS router nodes and phishing server nodes are organized into one machine. In our testbed, an AS node hosts the all phishing sites found at the AS in the wild. For example, if phishing sites were found at the AS #1, the AS node which represents AS #1 delivers these phishing sites. The AS node is also assigned the IP addresses of the phishing server nodes as alias IP addresses in order to accept the connection from a phishing detection node.

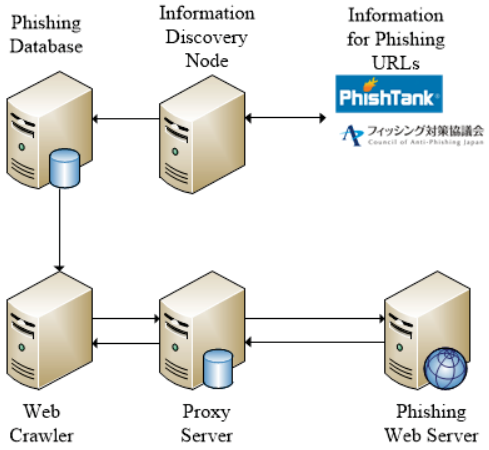

Figure 1: Crawler for phishing sites

As we mentioned in section 3.1, the phishing detection methods requires the website's URL, a domain name and its created date, IP address, AS number, and the content. These factors in the testbed can be as same as that in the wild even if the testbed employs the AS-level aggregation to phishing sites. Accordingly, we assume that there is no loss of realism brought by the aggregation.

3.3 Implementation

We implemented a crawler to preserve phishing content in the database. Based on the phishing crawlers in the past researches [19, 25], we crawled phishing sites by using rendering engine of the modern browser due to the capability of JavaScript. Note that Phishing sites often employ JavaScript, whereas traditional web crawlers do not equip it.

Figure 1 illustrates our implemented system. First, an information discovery node receives the URL of Phishing sites from external data resources, and it then stores the URL into phishing database. A web crawler periodically checks the URL, and accesses to the newly registered URL after it adds an unique identifier for the website in HTTP request header. The proxy server removes the identifier from the request, sends it to a phishing web server. After the proxy server receives web content from the server, it stores the content related to the identifier. Such crawling session will finish when the whole content in the website has loaded or threshold time has passed.

For reproduction of the phishing sites, we also implemented a web server. When the server receives HTTP request from a web browser, the server obtains the unique identifier from its URL by querying phishing database. Based on the identifier, the server sends the preserved content, that are stored when a web crawler visited to the URL, to the

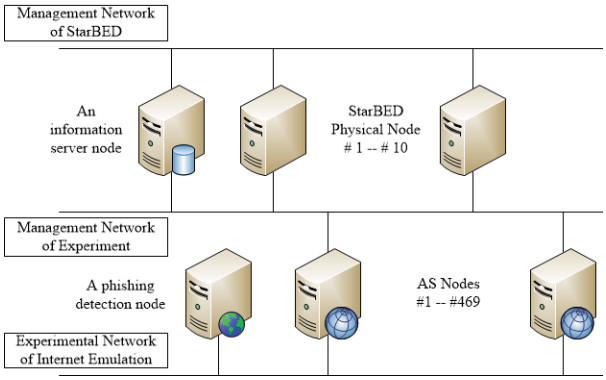

Figure 2: Network topology for our experiment

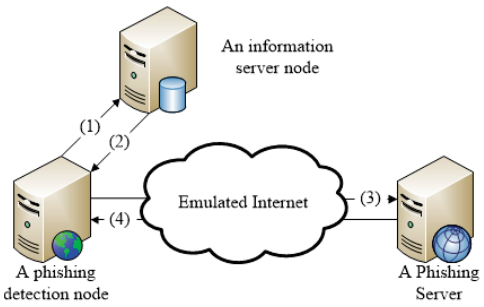

Figure 3: Workflow in the PhishCage

browser. We then configured a DNS server to resolve the FQDN in phishing sites' URL to phishing server nodes. As we mentioned in section 3.2, the nodes are assigned phishing server's IP addresses as the alias addresses, and hence, a phishing detection node were navigated to the suitable phishing server node. While the server nodes ran a reverse web proxy which interconnected to our implemented web server, they were able to output the realistic phishing content.

4. EXPERIMENT OF PHISHCAGE

This section explains how we setup the environment, and it then describes our evaluation of PhishCage. The prior objective of the evaluation was verifying whether or not our idea, the AS-level aggregation was feasible to keep the realism. Our evaluation checked if a phishing sites was able to be browsed and it was delivered from the appropriate AS node.

4.1 Experimental Setup

For constructing phishing server, we collected several information sources as follows.

Configuration of Network Topology: Our testbed employed the emulated Internet for network topology. AnyBed [21] is a useful tool for constructing the emulated Internet. It requires two configuration files. One is the physical network configuration file which describes testbed specific information such as hardware address of nodes and wiring among network switches. Other one is the logical network configuration file which describes network topology. Given CAIDA's AS Relationships Dataset (ASRD) [4] and a Routeviews Prefix to AS mappings Dataset (PFX2AS) [3], AnyBed can generate the network configuration file in which every BGP router advertises the realistic IP addresses. Due to the facility of the testbed, we filtered ASes to extract a subgraph. This paper employed the region based filtering [9] and constructed the network topology which represented Japan Internet including 469 of ASes.

Phishing Dataset: During September to November 2011, we stored the content of 50,451 phishing sites. Phishing URLs were reported phishing sites at PhishTank [16] and Council of Anti Phishing in Japan [7]. Of the 50,451 websites, 854 were found in Japanese ASes, and we obtained the contents of 121 websites; the rest were already expired when we tried to browse. We also checked the websites by heuristics described in section 2.2 and stored the detection results brought by heuristics into a database. The schemes of the database were organized the phishing site's URL, FQDN, its IP address, an AS number in which the IP address belonged, WHOIS result by submitting domain name, search result by submitting keywords, and the date of checking the website.

By using these datasets, we constructed PhishCage as shown in Figure 2. Our experiment configured eleven physical nodes of StarBED [20] that had 2 × Intel 6-Core Xeon X5670 CPU, 48GB memory, 2 × 500 GB SATA hard disk drives and 4 gigabit ethernet controllers. All nodes were installed Debian 6.0. Of the eleven nodes, ten nodes ran KVM instances to setup the emulated Internet. Each instance had 512 MB memory and 10 GB virtual hard disk. They were also installed Debian 6.0 and were equipped with Quagga [17], the most used open routing software. As we explained in section above, we prepared the physical configuration files for the KVM instances and the logical configuration files based on ASRD and PFX2AS, and finally made AnyBed generate the configuration files for OS and Quagga. The rest one physical node was used to manage each instance by providing DHCP, DNS and information for a phishing detection node. In addition, it took a couple of hours for building our test environment, excepting the install of OS to the physical / virtual node. But after we installed one physical node and one virtual node, we were able to copy the installed disk image to other nodes.

4.2 Functional Verification

Based on the phishing sites' IP addresses, PhishCage assigned the address and ran HTTP server for each AS node. When the AS node received an HTTP request, the node delivered the phishing sites by reassembling the stored content. We also configured the information server to run DNS server for resolving the phishing site's FQDN to IP address.

From the view point of the functional verification, we browsed the reproduced phishing sites. For a phishing detection node, our experiment created one KVM instances which connected to the emulated Internet. It then checked if the sites were available by browsing from the phishing detection node. Figure 3 illustrates the workflow. If a phishing URL was not employed an IP address, the phishing detection node (1) resolved its FQDN to the information server node, and (2) obtained the IP address of the phishing site. It then (3) accessed to the phishing server's IP address through the emulated Internet. The phishing server checked if the con-

tent has been stored, and finally (4) sent the entire phishing content to the phishing detection node. This verification showed that almost of all websites were able to browse as they really exist. It also observed that an information server provided the realistic DNS resolution.

We also tested our implemented version of the heuristics-based solution [25] which used eight heuristics. Of the six of eight, the heuristics worked regardless of the environmental difference. The rest of heuristics employed WHOIS results and search engine's result, so we modified our implementation to retrieve them from the information server. Note that the modification was negligible in this case.

We found that two exceptional patterns that we could not browse the sites. One was that the content were not correctly stored. Some phishing sites embedded the contents of the legitimate sites with SSL. Due to that we stored the content at the proxy, the content could not be captured because of the encryption.

The other was similar to the SSL, but it embedded the other phishing site's content via IP address. Assigning every IP address linked in the web page was not the suitable due to the scale of the emulated topology; our experiment emulated the topology in Japan, but the content in the outside of Japan would not be loaded.

Our phishing emulation was that one AS node hosted all phishing sites have been discovered at the AS. The former problem was caused of our approach for storing phishing sites, rather than emulation techniques. However, even if we stored content without using the proxy server, the certificates in the wild and that in the emulated Internet might be different. The latter was caused of the size of the testbed, rather than emulation techniques. Accordingly, we believe that our emulation is feasible to keep the reality in the context of phishing detection systems. Increase of the testbed's scalability is important, but beyond the scope of this paper.

5. DISCUSSION

PhishCage reproduced phishing sites that have the same URL, the same IP address, the same AS number, and the same visual of the phishing site. But, some heuristics required such information that WHOIS results, search result, as shown in section 4.2. The phishing detection systems were still able to work when an experimenter accepted to modify their systems. An alternative approach was to develop functions of search engines for the testbed, however, it might be difficult due to the unknown specifications of them. Instead of browsing search engine, receiving the stored result from the information server would support for the systems to work.

The other approach was giving limited Internet access to the phishing detection node. Remind that the many nodes were assigned and advertised realistic IP addresses, and hosted phishing sites. To avoid the damage suffered from connecting these nodes to the Internet, we defined our requirements to have no Internet access. But, giving Internet access only to the phishing detection node had some advantages. However, even if the system can access to the search engine, its search result might differ from the search result of the day, the site was detected.

However, this approach could help reproducing phishing sites that embedded contents via SSL transactions. Emulating every certificate authority in the emulated Internet was difficult, but answering the verification results that were already stored to the heuristics was feasible.

6. FUTURE WORK

In our future work, we will develop a cooperative detection system between phishing and other incidents. Empirically, phishing sites are often hosted on botnets, networks of compromised PCs (bots) controlled by a bot master. Accordingly, checking if a website hosted on a bot or not, will give hints to identify a phishing site.

We consider that botnets can be categorized like a cloud service and hence, multiple bot masters might share the resources of bots in the case of "public" botnet. If various attacks are simultaneously performed in the bot, the cooperative detection can be established.

Imagine such case that a phishing detection system found a suspicious website, but the system cannot clearly determine the website as "phishing." The system will label it as "not phishing" even if the website is an actual phishing site. However, if the host of the website sent suspicious IP packets seems to be Denial of Service (DoS) attack, phishing detection system would be able to understand that the website is hosted on a bot; it can label the site as phishing.

Accordingly, we assume that the use of the information brought by different countermeasures will increase the detection accuracy. In order to achieve such detection systems, we will extend PhishCage to support the reproduction of multiple incidents. It is difficult to construct a suitable testbed due to the difference in attack vectors, however, the emulated Internet can be a feasible solution for generating DoS traffic [9], as well as phishing sites.

The difficulty in such cooperative methods would be various demands of time granularity. Phishing sites should be discovered in hour zero [19], but countermeasures of DoS often require to estimate the attack source in couple of minutes. The differences should be considered in the cooperative analysis method against multiple incidents.

7. CONCLUSION

This study demonstrated an emulation technique for phishing sites. To the best of our knowledge, this was the first study for reproduction of phishing sites within the network emulation testbed. According to our survey of phishing detection methods, the URL-based heuristics required the phishing site's URL, a domain name and its created date, IP address, and AS number. The content-based heuristics needed that the whole contents of phishing sites can be browsed as if they exist. Some content-based heuristics, that used information retrieval techniques, also needed to obtain search results.

This paper then defined the requirements of testbed for phishing detection systems, aspect from the realism. With regard to the requirements, we designed and implemented systems for preservation of phishing sites and reproduction of them.

For our analysis, we selected 121 phishing sites from observed 50,451 phishing sites. Based on the past studies of emulation techniques such as the emulated Internet [9], we constructed an emulated Japanese Internet topology which consisted of 469 ASes. Phishing servers were assigned the realistic IP address and located at the realistic AS, all of which had been observed in the wild. We confirmed that the almost phishing sites were reproduced as if they existed,

and the stored information, such as WHOIS outputs and search results, were able to be provided to phishing detection systems.

Based on these findings, we believe that our emulation technique was a feasible solution to evaluate the effectiveness and universality of phishing detection systems. To solve remaining problems, we will explore the suitable way for emulation of legitimate services and increase the number of nodes in the testbed. We will also reproduce other attacks in the emulated testbed, for further development of cooperative detection methods among multiple incidents.

8. ACKNOWLEDGMENTS

We acknowledge the financial support from the University of Tokyo Global COE Program "Secure-Life Electronics." We also acknowledge to Security Architecture Laboratory, Network Security Research Institute, National Institute of Information and Communications Technology.

9. REFERENCES

[1] Aburrous, M. R., Hossain, M. A., Thabatah, F., and Dahal, K. Modelling Intelligent Phishing Detection System for E-banking Using Fuzzy Data Mining. In *Proceedings of International Conference on CYBERWORLDS* (Sep. 2009).

[2] Benzel, T. The science of cyber security experimentation: the DETER project. In *Proceedings of the 27th Annual Computer Security Applications Conference* (Dec. 2011), pp. 137–148.

[3] CAIDA. Routeviews Prefix to AS mappings Dataset(pfx2as). Available at: `http://www.caida.org/data/routing/routeviews-prefix2as.xml`.

[4] CAIDA: cooperative association for internet data analysis. The CAIDA AS Relationships Dataset. Available at: `http://www.caida.org/data/active/as-relationships/`.

[5] Calvet, J., Davis, C. R., Fernandez, J. M., Guizani, W., Kaczmarek, M., Marion, J.-Y., and St-Onge, P.-L. Isolated virtualised clusters: testbeds for high-risk security experimentation and training. In *Proceedings of Workshop on Cyber Security Experimentation and Test* (Aug. 2010).

[6] Chou, N., Ledesma, R., Teraguchi, Y., Boneh, D., and Mitchell, J. C. Client-side defense against web-based identity theft. In *Proceedings of the 11th Annual Network and Distributed System Security Symposium* (Feb. 2004).

[7] Council of Anti Phishing. Provide of Phishing sites' URL. Available at: `http://www.antiphishing.jp/enterprise/url.html`. (in Japanese).

[8] Fu, A. Y., Wenyin, L., and Deng, X. Detecting phishing web pages with visual similarity assessment based on earth mover's distance. *IEEE Transactions on Dependable and Security Computing 3*, 4 (2006), 301–311.

[9] Hazeyama, H., Suzuki, M., Miwa, S., Miyamoto, D., and Kadobayashi, Y. Outfitting an Inter-AS Topology to a Network Emulation TestBed for Realistic Performance Tests of DDoS Countermeasures. In *Proceedings of Workshop on Cyber Security Experimentation and Test* (Aug. 2008).

[10] Herzberg, A., and Gbara, A. TrustBar: Protecting (even Naïve) Web Users from Spoofing and Phishing Attacks. Tech. rep., Jul. 2004.

[11] Krammer, V. Phishing Defense against IDN Address Spoofing Attacks. In *Proceedings of the 4th Annual Conference on Privacy, Security, and Trust* (Oct. 2006).

[12] Medvet, E., Kirda, E., and Kruegel, C. Visual Similarity-Based Phishing Detection. In *Proceedings of the 4th International Conference on Security and Privacy in Communication Networks* (Sep. 2008), pp. 1–6.

[13] Miwa, S., Miyachi, T., Eto, M., Yoshizumi, M., and Shinoda, Y. Design and Implementation of an Isolated Sandbox with Mimetic Internet Used to Analyze Malwares. In *Proceedings of DETER Community Workshop on Cyber Security Experimentation and Test* (Aug 2007).

[14] Miyamoto, D., Hazeyama, H., and Kadobayashi, Y. HumanBoost: Utilization of Users' Past Trust Decision for Identifying Fraudulent Websites. *Journal of Intelligent Learning Systems and Applications 2*, 4 (2010), 190–199.

[15] Netcraft. Netcraft Anti-Phishing Toolbar. Available at: `http://toolbar.netcraft.com/`.

[16] OpenDNS. PhishTank - Join the fight against phishing. Available at: `http://www.phishtank.com`.

[17] Quagga. Quagga Routing Suite. Available at: `http://www.nongnu.org/quagga/`.

[18] Sanaga, P., Duerig, J., Ricci, R., and Lepreau, J. Modeling and Emulation of Internet Paths. In *Proceedings of the 6th USENIX Symposium on Networked Systems Design and Implementation(NSDI)* (Apr. 2009).

[19] Sheng, S., Wardman, B., Warner, G., Cranor, L. F., Hong, J., and Zhang, C. An Empirical Analysis of Phishing Blacklists. In *Proceedings of the 6th Conference on Email and Anti-Spam* (Jul. 2009).

[20] StarBED Technology Center. StarBED Project. Available at: `http://www.starbed.org/`.

[21] Suzuki, M., Hazeyama, H., Miyamoto, D., Miwa, S., and Kadobayashi, Y. Expediting Experiments across Testbeds with AnyBed: A Testbed-Independent Topology Configuration System and Its Tool Set. *IEICE Transactions on Information and System E92-D*, 10 (Oct. 2009), 1877–1887.

[22] Whittaker, C., Ryner, B., and Nazif, M. Large-Scale Automatic Classification of Phishing Pages. In *Proceedings of the 17th Annual Network and Distributed System Security Symposium* (Feb. 2010).

[23] Xiang, G., and Hong, J. I. A Hybrid Phish Detection Approach by Identity Discovery and Keywords Retrieval. In *Proceedings of the 17th World Wide Web Conference* (Apr. 2009).

[24] Zhang, Y., Egelman, S., Cranor, L., and Hong, J. Phinding Phish: Evaluating Anti-Phishing Tools. In *Proceedings of the 14th Annual Network and Distributed System Security Symposium* (Feb. 2007).

[25] Zhang, Y., Hong, J., and Cranor, L. CANTINA: A Content-Based Approach to Detect Phishing Web Sites. In *Proceedings of the 16th World Wide Web Conference* (May 2007).