Volume 1, No. 3, Sept-Oct 2010
ISSN No. 0976-5697
International Journal of Advanced Research in Computer Science
RESEARCH PAPER
Available Online at www.ijarcs.info

A Survey on Methodologies and Techniques for Detection and Prevention of Phishing Attacks

M. Nirmala
School of Information Technology and Engineering,
VIT University,
Vellore, India
mnirmala@vit.ac.in

K. Naveen Kumar
School of Information Technology and Engineering,
VIT University,
Vellore, India
naveenwashere@gmail.com

L. D. Dhinesh Babu*
School of Information Technology and Engineering,
VIT University
Vellore, India
lddhineshbabu@vit.ac.in

Abstract: Privacy, security and integrity of the users' data over internet depend on a single piece of user information which is normally a 'password'. It is very important for the user to keep it as secure and safe as possible in order to prevent the information from being revealed to an adversary who can misuse it. But most of the time we fail. Knowingly or unknowingly we tend to give away these secrets to others, resulting in a huge loss or embarrassment. Such secrets, mainly the username and password are often given away to attackers over the internet and become victims of what is known as a phishing attack. Phishing is a technique where the attackers masquerade as a trustworthy entity and trick us to submit our credentials, mostly our usernames, passwords and credit card details etc. In this paper we shall take you around various techniques and methodologies available to prevent such a theft. We also present the reasons why the current techniques and tools could not prevent these attacks.

Keywords: Abuse and crime involving computers, authentication, cyber cash, digital cash, online fraud detection and theft, phishing, security and protection

I. INTRODUCTION

Phishing technique was described in detail in a paper presentation at the International HP Users Group, Interex [26] in the year 1987. The phising attack on AOL in the year 1995 attracted the attention of security experts' world over and it came to limelight as a major security threat. AOL soon came up with security meas-ures to counter this threat. Then evolved the security issue of targeting specific users on the internet, and this was later termed as spear phishing [25]. In certain high profile cases, where the targets were the top offcials of huge organizations, it was given with the name whaling [26]. The use of the term 'Phishing' was first officially recorded in the year 1996 following the AOL episode.

Initially the attackers sent emails with a link, asking the users to follow it to change their usernames, etc, in the name of "Account Verification", "Verify Billing Information", and so on. This slowly spread from emails to targeting users on Instant Messaging (IM) services. This forced AOL to widely advertise informing its users through various messages and IM services telling "no one working at AOL will ask for your password or billing information", etc.

The first known attack over a payment system was recorded in 2001(. It was against a payment system known as E-gold [8]. By 2004 phishing was recognized as a fully industrialized part of the economy of crime. The most recent data [1, 2] tells us that there were 55, 698 attacks on the first half of 2009. The figure [2] here shows the statistics of the year 2009 based on the reports received by them.

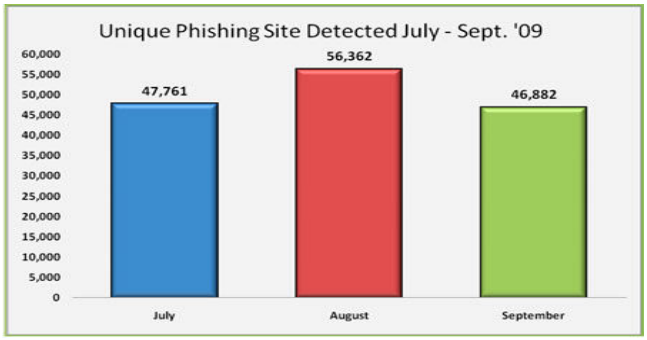

Figure-1: The number of unique phishing websites detected by the APWG during the third quarter of 2009[2]

3.6 million adults lost US $3.2 billion in the year 2007 [1, 26]. Phishing has seen a transition from AOL to IMs, IMs to financial institutions, social networking sites, and file sharing websites and to almost everything on the internet. Apart from the attacks on the web, phishing exists in various other forms as vishing (voice phishing, which is also known as phone phishing), SMSishing (phishing through SMS), etc.

II. SOME KNOWN PHISHING ATTACKS

- Link manipulation: it is a technique where an email is sent by the attacker with a link in it. The anchor text of the link appears to be legitimate but upon clicking, it takes us to a site that looks exactly similar to the orig-inal one. For example the link

http://www.abccorporation.com may appear to be the original website address of ABC Corporation but it doesn't take us to the legitimate site.

- Another old technique is to use the '@' symbol to spoof the sites. For example http://www.mycollege.com@notmycollege.com takes us to a site which is www.notmycollege.com rather than what it claims to be i.e., www.mycollege.com. This is called spoofing [1] a website.
- A further problem of link manipulation is with respect to the visually similar sites. For example, the website www.paypal.com looks similar to www.PayPal.com or www.Paypal.com. Attackers make use of such mistakes committed by the users' to their advantage in order to steal their information. Attackers also took advantage of an already existing flaw known as IDN Spoofing or homograph attack [6], using open URL redirectors on the websites of trusted organizations to disguise malicious URLs with a trusted domain.
- Filter Evasion: many security firms designed techniques to filter out phishing sites based on the content, links, etc in the web pages. But such filters were evaded by the attackers by using images, instead of text [1, 9].
- Website Forgery: many websites that support online payments and secure transactions have unique address bar. They come with either the company's symbol or with a color or a lock symbol, etc, to show that those sites belong to the original company and are secure. Attackers used Java Script commands [1] to change the address bar and deceive the users. It basically dealt with forging the website's scripts to their advantage. These attacks later came to be known as Cross-site scripting (or CSS) [1] attacks.
- Phone phishing: phishng was not limited to forging websites alone; there was a next level of this attack which came to be known as phone phishing. People received phone calls claiming that they were from the bank where they held accounts. After gaining the trust of the user, they got all the sensitive bank information which was enough to clean the users' accounts by claiming that the bank had been experiencing problems with the accounts and the information was essential to sort the problems.
- Pop-up windows: the attackers successfully forwarded the client to the bank's legitimate website. Then they used a pop-up window [26] requesting the username and password, as if it were being asked by the bank itself. This was a tricky one since it was really difficult to check the authenticity of the pop-up window.

Many tools, strategies and solutions were proposed and put in use to overcome phishing. We shall now look at various solutions that came and went till now and those that are still in use.

III. PHISHING REMEDIES

Anti-phishing measures were implemented in various ways. Some were just design solutions for strong authentication; some were used as plug-ins in browsers, toolbars and few others as part of login procedures. Here we divide these anti-phishing strategies into various categories as: user training, security solutions and tools (toolbars and plug-ins). Each one of these techniques is analysed and then discussed below.

A. *Use Training*

As users tended to ignore the warnings given by the tools, an initiative to train the users on the aspects of phishing was taken. Users were given information on how could they be tricked into phishing attacks. The most basic approach was to post articles and materials on websites that taught the users, how to detect and prevent phishing. Whereas the most interactive way was to let users assess their knowledge on phishing through some web-based tests. Few sites put up some flash based games where the user had to indentify which site was a phish and whish wasn't based on some rules. It made the whole training more fun and productive.

Phishing education was also conducted in a class-room setting, as had been done by Robila and Ragucci [14]. The idea of sending fake phishing emails [13] to test users' vulnerability had been explored by several groups. Typically, at the end of such studies, all users were given additional materials to teach them about phishing attacks. This approach had been used with Indiana University students and West Point cadets, as well as with employees at a New York state office. This study was conducted in two phases. In the first phase the participants were tried to detect phishing sites without any training and were tested for their ability. In the second phase, they were given materials on phishing and then tested again. On comparison the studies showed significant improvement.

B. *Security Solutions*

Security solutions could be further classified into two groups such as third party certifications and authentication mechanisms. Most of these solutions could be implemented independent of the browsers used and their versions. We shall now look at various security solutions in detail.

- Humboldt: A Distributed Phishing Disruption System

Humboldt works by submitting poisonous fake data to phishing web sites that cannot be distinguished from the input of actual data submitted by phishing victims [31]. The poisonous data collected by a phisher is in such a way that it produces detectable behaviors when the phisher attempts to use it. This provides a mechanism for tracking activities associated with identity theft. Humboldt is evaluated to show how effective it is in disrupting phishing operations with very low overhead.

Poisonous data from Humboldt is not distinguishable from the data submitted by real phishing victims, not only in terms of the data alone, but also in the way the data is submitted;

- The submission of poisonous data was coordinated among Humboldt clients so that it could prevent detectable behavior which would make postprocessing by phishers easier and also it could avoid the risk of launching DDoS attacks against those innocent machine that hosted the phishing site; and
- Data submission from Humboldt was also automated, without requiring manual intervention from users.

With enough clients, Humboldt could inject a significant amount of fake data into the phisher's database [31]. It either disrupted the phishing campaign or exposed the phishers when they tried to use those fake credentials—which were generated and recorded by Humboldt—on the real web sites they were pretending to be.

The SEFAP Mechanism: The SEFAP system is an extendable, signature-based, and secure email system. The SEFAP [35] consists of three layers: presentation layer, business layer and database information layer. Only the presentation layer is

accessible by users and the other two lower layers are protected and accessible to system administrators only.

SEFAP consisted of a SEFAP client, SEFAP server, and database server which communicate through a secure channel. SEFAP authenticates the origin of incoming email and takes appropriate actions to suspicious phishing email in order to mitigate phishing attacks. SEFAP was designed to adopt signature schemes. Since each original email server already has its unique domain name on the Internet, domain level system parameters were designed to be generated by the SEFAP server located in each physical email server.

The SEFAP server provided six sub-services [35]: sys-tem management module, signature-scheme based parameters setup module, private key extraction module, verification module, synchronization module and dispatcher module. The system management module specifies a signature scheme for the current email server system. It could also add a new signature scheme into the SEFAP system and delete an old signature scheme from the SEFAP system. Thus, SEFAP could be updated for a new signature scheme though uploading the new signature-scheme component to the SEFAP system.

The signature-scheme-based parameter setup module generated domain system parameters under a selected scheme. The private key extraction module generated its user's private key which is then delivered it to its user with a secure channel. The verification module provides the signature verification service even if the outgoing email server uses a different signature scheme from the incoming email server, and instructs the email server to take appropriate actions to unidentified emails. The synchronization module dealt with the domain parameter synchronization operation and publication.

The dispatcher module provided the most efficient process schedule to verify incoming email. The SEFAP client located at the sender's machine was made responsible for signing email when the user instructed the email server to send an email message. The SEFAP client also is in charge of system parameters license synchronization including checking the parameter version, expiration, and signature scheme identification using an efficient synchronization algorithm. These modules provide a tight layer of security to ensure that the emails sent and received between the server and user does not contain any spoofed mails or links to phishing sites.

C. Security Solutions

The strategy of this technique was to provide protection without the giving any burden of work to the user. The phishing sites were detected and removed from the web silently. Also the fraud emails and messages were detected and deleted [6, 15]. But the problem here was that we cannot achieve cent percent accuracy every time. By the time a phishing site was detected, it would have been online for long enough to snare unsuspecting victims. According to the Anti-Phishing Working Group (APWG), phishing sites manage stay online on average for 4.8 days [2].

D. Warning Users

A number of tools were developed to warn users that the website they are visiting was likely to be fraudulent or legitimate. They either provided explicit warnings or provided interfaces that helped people notice that they may be on a phishing website. Ye and Sean and Dhamija [4] and Tygar developed prototype "trusted paths" for the Mozilla web browser that was designed to assist users in verifying that their browser has made a secure connection to a trusted site. More common were few toolbars which provided indication of overall safety of the website by flashing red or green lights on the browsers [7, 11, and 18]. But they had their share of weaknesses:

- First, it required people to install special software (although newer versions of web browsers had such software included).
- Second, user studies showed that users often did not understand or act on the indications or warnings provided by toolbars.
- Third, a recent study shows that some anti-phishing toolbars are not very accurate, and even the best toolbars may miss over 20% of phishing websites.

Even though many techniques already existed to pre-vent phishing, the rapid growth in the attacks called for more better and strong solutions. Since then many solutions were proposed ranging from quick fixes to substantial redesigns. All these proposals were evaluated based on four security properties: the limited human skills property, general purpose graphics property, the golden arches property, the unmotivated user property and the barn door property. Few proposals addressed most of these properties whereas few failed to do so. Attempts to solve the phishing problem were again divided into three approaches: going for Third party certifications, designing direct authentication mechanisms, and Anti-Phishing tools.

E. Third Party certifications

- Hierarchical and distributed trust models

Third party certification includes hierarchical trust models, like Public Key Infrastructure (PKI), which were proposed long ago as a solution for users to authenticate servers and vice-versa. In PKI, chains of Certificate Authorities (CAs) vouch for identity by binding a public key to a entity in a digital certificate. The Secure Sockets Layer (SSL), now known as Trans-port Layer Security (TLS), both rely on PKI. The problem here was that, in the typical use of SSL today only the server is authenticated. Even with the wide use of one-sided SSL (in the form of server digital certificates signed by a trusted CA), there were problems. As examined in their task analysis, certificates had been falsely issued, and most users did not have the know-ledge or skill to understand digital certificates and the delegation of trust.

Other third party approaches included "web of trust" distributed trust models (e.g., Pretty Good Privacy) and the use of third party seals to indicate trusted websites (e.g. Verisign Seal Program and TRUSTe [23]). By displaying seals as graphics that can be easily copied, trusted seal programs ignored the "general purpose graphics" property.

- Trustbar

The "Trustbar" proposal was again a third party certification solution, where websites logos were certified. A "trusted credentials area" was created as a fixed part of the browser window [1]. This area was used to present credentials from the website, such as logos, icons and seals of the brand that had been certified by trusted certificate authorities or by peers using a PGP "web of trust". Strength of this solution was that it did not rely on complex security indicators. However, careful consideration had to be given to the "general purpose graphics" and "golden arches" properties. Because, since the logos do not change, they could be easily copied and the credentials area of the browser could be spoofed (e.g., an attacker can draw an image of the credentials area into the top portion of an un-trusted webpage to make it appear trusted). Therefore, careful consideration had to be given to the design of an indicator for insecure windows so that spoofed

credentials could be easily detected. But there remained an ambiguity in the cases where two sites used similar logos and they were supposed to certify uniquely.

F. Direct Authentication

This approach included user authentication and server authentication schemes.

a)Multi-Factor User Authentication: These schemes used a combination of factors to authenticate the user. The factors can be something you know (for example, a password or ATM PIN), something you have (for example, a token or key) or something you are (for example, biometrics).

- AOL Passcode

America Online's Passcode was proposed as a phishing defense. This program distributed RSA SecurID [15, 16] devices to members of AOL. The device generated a unique six-digit numeric code and displayed it, every 60 seconds. This could be then used as a secondary password while logging into AOL website. This scheme reduced the value of collecting passwords because the passwords were of no use for another transaction. But however, they failed to prevent a man-in-the-middle (MITM) attack where the attacker could lure a user to a spoofed AOL website so that he/she can collect both the primary and secondary passwords. These passwords can immediately be presented by the attacker to AOL in order to masquerade as the user. The Passcode program did however raise the bar for phishing attacks today, but it has its own issues; phishers would soon turn to this type of "live" MITM attack, if the bar was raised everywhere. This again had a problem. This scheme ignored the "limited human skills" property, by not providing the user with any means whatsoever to verify the correct identity of the server.

- Secondary SMS Passwords

One of the other two factor user-authentication schemes was issuing secondary passwords to users through Short Message Service (SMS) as text messages on their cell phones. This was again susceptible to MITM attacks. Originally these two factor user authentication schemes were used to protect the server from fraud rather than protecting the users from phishing. This again ignored the "limited human skills property".

b) Server Authentication using Shared Secrets:

- Passmark and verified by Visa

Shared secret schemes were one of the simplest ways to authenticate web servers. In this technique, the user had to share a secret such as an image and/or a pass phrase. This secret will be later revealed by the server to the user to authenticate itself [24]. The most obvious drawback of this method was that the server had to display this secret in order to authenticate itself to the user. So this gave a chance to the attacker to capture and replay it. But this technique used the concept of cookies. The server placed a cookie on the user machine thus preventing MITM attacks. However, this did not prevent the attack in which a rouge server instructed the browser two identical windows, where one was legitimate and the other one is a phish. By careful placement of the rouge window, the attacker could make the user enter the username and password into the phish rather than the original one. This was done by spoofing the passmark [11] "re-registration" process.

The passmark had to be re-registered in case the user wished to uses a computer in which the cookie is already not set or the cookie had been deleted [23]. Hence, the attacker was able to redirect the user to a page where it claims that the page has been deleted, in order to make the user re-register again. The legitimate page that showed the error always asked the user to ensure that he/she has reached this page by manually typing the URL by hand [23]. The spoofed page however did not include this error.

- ViWiD

M. Topkara et al. proposed a novel scheme 'ViWiD', which was based on watermarking and it is implemented it, for mitigating phishing attacks. This was a mechanism to check the integrity of logo images based on watermarking [30]. The entire computation is performed on the company's web server, by ViWiD. It did not require installation of any tool or storage of any data, such as keys or history logs, on the user's machine. The watermark message was designed so that is was unique for every user and, it carried a shared secret between the company and the user in order to thwart the 'one size fits all' attacks.

Another effective approach to detection of Web page phishing was proposed, which used Earth Mover's Dis-tance (EMD). It was used to measure the Web page's visual similarity [29]. The involved Web pages were first converted into low resolution images and then color and coordinate features were used to represent the image signatures. After doing that EMD [30] was used to calculate the signature distances of the images of the Web pages.

A number of attacks that required more difficulty were possible (e.g., breaking the secure cookie, physical observation of the secret image, discovering the potential range of images and then guessing the image). However, spoofing required the least amount of effort to defeat the most people, and was expected that this type of spoofing attack would become common if systems like Passmark were widely deployed. Evidence suggested that users were able to correctly recognize a large number of images. However, the problem was that if a user is required to remember different images or passphrases for a number of different servers, any difficulty in recognizing an image could be exploited by an attacker. This scheme ignores the "limited human skills", "general purpose graphics" and "golden arches" properties.

Server authentication using self-shared secrets

This authentication scheme required the user to share a secret with his/her own device (for example web browser) rather than the web server.

- SRD (Synchronized Random Dynamic) Boundaries

Ye and Smith proposed "Synchronized Random Dy-namic Boundaries" to secure the path from users to their browser [28]. This scheme used a random number generator to set a bit that determined whether the browser border is inset or outset. The browser border alternates between inset and outset at a certain frequency in concert with a reference window. The strength of this solution was that it was good in recognizing the "general purpose graphics" problem. In this technique, rogue servers could not predict the random number which is chosen by the browser, and therefore it was difficult to create

spoof windows that blink at the correct frequency. But a weakness of this approach was that it ignores the "limited human skills" property; dynamically "blinking" borders may be easy to distinguish for the users, and frequent border changes were likely to prove to be distracting. The security depended on how many border frequency op-tions are available and how many users can differentiate.

- YURL Petnames

In the YURL proposal, the user's browser maintained a mapping of a public key hash to petname. When a user visited a page that is identified by a YURL, the browser displayed the petname that the user previously associated with the public key hash [24]. This helped in recognizing an un-trusted site if the corresponding petname was not present. This was a very simple scheme that required a small degree of personalization for each website. But scheme ignored the "unmotivated user property" because security relies on users to be motivated to customize petnames for trusted sites. One advantage of this technique was that the secret (the pet-name) was shared with the user's browser, rather than with the trusted server. Careful consideration had to be given to the design of the un-trusted state i.e., un-trusted windows had to be clearly marked as having no petname. Otherwise, attackers could spoof the petname display area in the browser and fool many users.

The "limited human skills" property was also important. Petnames relied on user's memory to recognize, and to associate the secret phrase with the correct website. It was expected that users would choose predictable petnames. For example, many users would choose "Google" for google.com. The designers can encourage users to select unique petnames to improve spoof resistance.

G. Anti-Phisihing tools

- eBay Toolbar

The eBay Toolbar is a browser plug-in that eBay of-fered to its customers. It helped keep track of auction sites for them. The toolbar has a feature, known as AccountGuard [5], which monitors web pages that users visit and provided a warning in the form of a colored tab on the toolbar. The tab usually appears grey, but turns green if the user is on an eBay or PayPal site or red if the user is visiting a site that is known to be a spoof by eBay [5]. The toolbar also allowed users to submit suspected spoof sites to eBay. One big drawback to this particular approach was that it only worked for eBay and PayPal websites. Users would not want to maintain too many toolbars that each and every site offers to detect phish. However, it is not difficult to develop a generalized program or tool for this. The main weakness was that there would always be a period of time between the loading of a webpage and the time taken for a spoof to be detected and also when the toolbar can begin detecting spoofs for users. If spoofs are not carefully confirmed, denial of service attacks is possible. This indicates that some percentage of users will still be vulnerable to spoofing. For these users, "the barn door" property means that their personal data will not be protected.

- SpoofGuard

SpoofGuard is an Internet Explorer browser plug-in that examines web pages and warns users when a certain page has a high probability of being a spoof. This calculation is performed by carefully examining the domain name, links and images and comparing them to the stored history and also by detecting common characteristics of spoofed websites [19]. It would make phishers to work harder to create spoof pages, if used. However, SpoofGuard always had to stay one step ahead of phishers, who could test their web pages against it. New detection tests were continuously needed to be deployed as phishers become more sophisticated.

SpoofGuard made use of what is called PwdHash [19]. It was an Internet Explorer plug-in that replaced a user's password with a one way hash of the password and the domain name. So, the web server only received a domain-specific hash of the password instead of the password itself. This was a simple but useful technique in addressing the "barn door property" and preventing phishers from collecting user passwords. Both Spoof-Guard and PwdHash ignored the "general purpose graphics" property by using a security indicator (a traffic light) that can be easily copied.

- Spoofstick

Spoofstick is again a toolbar extension for Internet Explorer and Mozilla Firefox that provided basic information about the website's domain name. That is, if the user was visiting Amazon, the toolbar would display "You're on amazon.com". If the user was at a website site that was spoofed, the toolbar instead displayes "You're on 20.191.132.45". This toolbar helped users to detect attacks where the rogue website had a domain name that is syntactically or semantically similar to a legitimate site [20]. Unfortunately, the current implementation of Spoofstick could easily be fooled by clever use of frames when different websites were opened in multiple frames in the browser window. This ignored the "limited human skills" property, because, users had to be aware of the use of hidden frames on a webpage. Spoofstick does address the "general purpose graphics" property by allowing users to customize the appearance of the toolbar.

However, most of the above tools relied on primarily on blacklists and lists of URLs that have been observed hosting phishing attacks. Blacklists provided no protection from attacks that were not already flagged as phishing. There were considerable numbers of such missed attacks. Researchers had proposed supplementing blacklists with Information Retrieval (IR)-based tools [20]. However, an IR-based approach was assumed to have generating false positives; legitimate websites were being incorrectly flagged as phishing. False positives undermined user's trust in a tool and posed questions of legal liability. This basically added more to the phishing ability of the attackers.

- BayeShield: Conversational Anti-Phishing Interface

To overcome the above problem, Peter Likarish et al, came up with an idea of BayeShield user interface which acted as a front-end to IR-based tools to identify phishing attacks with high probability but still with a few false positives [12]. This required one pre-requisite, to educate users through a series of questions that lead to a conclusion whether the website was legitimate or an attack. This was a lengthy process to be followed every time a site was to be opened. Also, this worked only 65% of the time providing correct solutions. Its tendency to flag legitimate websites as an attack sometimes, pose to be source of confusion to the users.

- iTrustPage

iTrustPage is an anti-phishing tool that avoided full fledged automation and instead went for user assistance to detect phishing [21]. iTrustPage, also relied on external repositories

of information in order to prevent users from filling phishing Web forms. With iTrustPage, users helped to decide whether or not a Web page is legitimate. Since iTrustPage is user-assisted, it avoided the false positives as well as the false negatives associated with automatic detection of phish, to a large extent. It was implemented as a downloadable extension to FireFox. This it-self proved to be a disadvantage as it limited the range of browsers on which this tool could be used. Also, its use becomes difficult in organizational networks where downloading is prohibited, for example universities, etc.

- TruWallet

TruWallet is a wallet-based authentication tool. It improved the previously proposed or used solutions for protecting web-based authentication. In contrast to other wallet-based solutions, TruWallet [17] provided (i) strong protection for users' credentials and sensitive data by cryptographically binding them to the user's platform configuration based on Trusted Computing technology, (ii) an automated login procedure where the server is authenticated independently from (SSL) certificates, thus limiting the possibility of attacks based on hijacked certificates and allowing less dependency on the SSL PKI model, and (iii) a secure migration protocol for transferring wallet data to other platforms. This tool used a small virtualization-based security kernel with trusted computing support and works with standard SSL-based authentication solutions for the web, where only minor modifications and extensions were required. It was made interoperable so that existing operating systems and applications like web browsers could be re-used.

- SpamAssasin

SpamAssasin is an open source spam filter. SpamAssasin [18] identified spam signatures using a wide variety of local and network tests. Using this made it very hard for spammers to identify even a single aspect with which they could craft their messages to work. A well designed, abstract API was used to encapsulate its logic, so it could be integrated anywhere in the email stream. It required very little configuration. It was not required that the users should continually update it with their mailing list memberships, mail accounts, etc. Once classification was done, site and user-specific policies could be applied against spam. Policies could be applied on both mail servers. Later it could be done using the user's own mail user-agent application. This tool helped in filtering a large extent of the spam emails sent by the attackers targeting the users.

- Dynamic security skins

It is an interesting solution which has been proposed by R. Dhamija et al. It involves the use of a so-called dy-namic security skin on the user's browser. It was implemented as a plug-in for Mozilla. It allowed the remote server to prove its identity to the user in a new and unique way. Only the user could verify the server but it was very difficult for a phisher to spoof. The disadvantage of this approach is that it doesn't conform to the "unmotivated user" property. It required effort by the user. In fact, it required the user needs to be aware of the phishing threat and check for any signs that the site he/she is visiting is spoofed or not. It is to be noted that in a later study, R. Dhamija et al. reported that more than 20% of the users ignored the visual clues when surfing and that visual deception attacks could fool even the most advanced users.

- PhishCatch

The basic architecture of PhishCatch [32] consisted of a module that could fetch emails, a module that could filter emails and then classified them as phishing, Alerter that could issue an alert to the user and the data warehouse that stored all the information related to phishing emails.

The PishCatch algorithm was designed to work with POP and IMAP mail servers, to fetch the emails. Whenever a new mail came in, the email was retrieved and split up into headers and body. Once the email is stripped into its component parts, the next step in the algorithm was to apply the phishing filters on the email to detect a phishing email.

Firstly, the email is scanned for the presence of the text filters defined in their algorithm. The number of text filters detected in the email is recorded, which would be the weight of that filter. The weight of the filter is then added to a list, Phishrank. Phishrank is a list which contains a mapping of the phishing filters to their respective weights.

In the next step, the received domain mismatch is checked in the email i.e., the domain similarity between the Received from and 'From' fields in the email is verified. The first Received 'From' and the 'From' fields are obtained from the e-mail header. If both these fields did not have the same domain, then it was assumed that the source address was spoofed in the email and hence the appropriate weight was assigned to the received domain mismatch filter.

The principle behind the ranking system was that a rank was assigned to each link based on the probability of it being a phishing link. The probability was deduced by looking at which filters the phishing link triggers. The identified phishing link was stored and used for the information gathering and for cross verification with PhishTank data. PhishTank is a collaborative clearing house for data and information about phishing on the Internet. PhishTank is a publicly available phishing database that receives phishing links from users. These links are voted upon by the users and based on the number of votes the links receive, they are classified as phishing links or not. Popular web browsers like Mozilla Firefox use PhishTank data to detect phishing links and alert the user about the phishing link.

- AntiPhish

AntiPhish is again a browser extension or a plug-in that provided protection to users' against spoofed website type of phishing attacks [33]. AntiPhish kept track of the sensitive information that belongs to a user and it generated warnings whenever the user accidentally attempted to give away such information to a web site that was actually considered untrusted. Automated form-filler applications were the inspiration behind the development of AntiPhish. We must have experienced many times in browsers such as Mozilla or the Internet Explorer a functionality that allows form contents to be filled automatically whenever a user desires. That information is normally stored by the browser and is automatically shown to us when we attempt to fill the form. Such content is normally protected by a master password. The browser uses symmetric DES algorithm for encryption and decryption purposes. Upon entryof this password, a previously filled login form, for example, will be automatically filled by the browser whenever it is ac-cessed.

This common functionality was taken one step further to track where the information was being sent. Not only this sensitive information was stored but also AntiPhish stored a mapping of where this information actually belonged to. That is, it also stored the domain of the web site where this information was originally entered. The effectiveness of private information preserving approach was totally dependent on users. To keep their private information could prove to be irritating works for the users. And it was not a good idea to store private information which is mostly memorized by users in a computer system.

- PhishingGuard

The idea of PhishingGuard [34] is that a web site can be identified by its IP address itself and most users have not so many URLs related to their credentials or private information. That's why it made use of white list ap-proach.

Phishing URLs that had been reported to Anti Phish-ing Working Group (APWG [1]) were analyzed and classified into three representation types:

Type 1: Explicit Representation
Type 2: Similar Representation
Type 3: Spoofed Representation

When a user accessed a web site, the (URL, IP) informa-tion was passed to what was known as the Access En-forcement Facility(AEF) [34] to check if the web site was a Phishing site, Phishing-suspicious, and DNS record for the URL had been spoofed(pharmed) or not.

Phishing Detection module'a work was to look up a URL in Trust Site List same as passed URL from AEF. If those URLs were same but the IP addresses are found different, then the Phishing Detection module returned a sign of explicit Phishing. On the other hands, for Trust Site List, Phishing Warning module searched a URL that was similar to passed URL from AEF. Phishing Warning message was shown to a user by this similarity check.

Apart from the tools and security solutions shown above there are organizations like Anti-Phishing Work Group (APWG), TRUSTe and NetCraft that provide anti-fraud and anti-phishng services, online privacy seals, etc, to further safeguard our websites and personal information from being stolen by the attackers.

Anti-Phishing Working Group (APWG) [1], is the global pan-industrial and law enforcement association focused on eliminating the fraud and identity theft that result from phishing, pharming and email spoofing of all types.

TRUSTe is an independent, privately held organization based in San Francisco, California. It is best known for its online privacy seals. It has certified more than 3,500 websites which include popular online portals and leading brands like Yahoo, Microsoft, Facebook and Appli Inc. The world's largest privacy seal program is operated by it [22]. TRUSTe's pro-fessional service offerings included consumer dispute resolution, site reputation management, and vender evaluation services as well as privacy policy generation.

Netcraft is an Internet services company based in Bath, England. It provides services like internet security, which includes anti-fraud and anti-phishing services, application testing, and automated penetration testing and code reviews [10].

IV. DRAWBACKS AND CONCLUSION

Most of the anti-phishing tools here seem to have usability problems [28]. Anti-phishing tools were able to identify all fraudulent web sites without any false positives, but because of usability problems, users could still fall victim to fraud. User testing was needed to better understand how users reacted to each different style of warning, for example, in eBay tool bar when it flashes different colors. Future studies on anti-phishing tools should also take into consideration, usability testing. A technically sound tool is of little or no use if users are unsure of what it is trying to communicate to them. Previous research has examined the effectiveness of several techniques for informing users about phishing [27]. However, it did not evaluate the effectiveness of pop-up warnings, or the difference in user reaction upon seeing a warning versus having a web site blocked. Usability problems plague all varieties of software - particularly security software. Poor usability, for an anti-phishing tool, means the difference between correctly taking someone away from a phishing site and then has them ignore the warnings only to become a victim of identity theft.

V. REFERENCES

[1] Anti-Phishing Working group. http://www.antiphishing.org/.

[2] Anti-Phishing Working Group, Global Phishing Survey: Trends and Domain name use in 1H2009, 2009

[3] Core Street, Spoofstick, http://www.corestreet.com/spoofstick/

[4] Dhamija, R. and Tygar, J. D. 2005. The battle against phishing: Dynamic Security Skins. In Proceedings of the 2005 Symposium on Usable Privacy and Security, SOUPS '05, vol. 93, ACM Press.

[5] eBay Toolbar, http://pages.ebay.com/ebay_toolbar/

[6] Evgeniy Gabrilovich and Alex Gontmakher. "The Homograph Attack" (PDF), February 2002, ACM

[7] Federal Trade Commission, Phishing Alerts, http://www.ftc.gov/bcp/conline/pubs/alerts/phishingalrt.htm

[8] Financial Cryptography, "GP4.3 - Growth and Fraud - Case #3 - Phishing", December 30, 2005, https://financialcryptography.com/mt/archives/000609.html.

[9] Mutton, Paul. "Fraudsters seek to make phishing sites unde-tectable by content filters". Netcraft. http://news.netcraft.com/archives/2005/05/12/fraudsters_seek_to_make_phishing_sites_undetectable_by_content_filters.html.

[10] Netcraft, http://news.netcraft.com/

[11] PassMark Security, Protecting Your Customers from Phishing Attacks- An Introduction to PassMarks, http://www.passmarksecurity.com/

[12] Peter Likarish, Don Dunbar, Juan Pablo Hourcade, Eunjin Jung, BayeShield: Conversational Anti-phishing User Interface, Symposium On Usable Privacy and Security (SOUPS) 2009, ACM.

[13] Ponnurangam Kumaraguru, Yong Rhee, Alessandro Acquisti, Lorrie Faith Cranor, Jason Hong, Elizabeth Nunge, Protecting People from Phishing: The Design and Evaluation of an Embedded Training Email System, CHI 2007, ACM

[14] Robila, S. A., J. James and W. Ragucci. 2006. Don't be a phish: steps in user education. ITICSE '06: Proceedings of

the 11th annual SIGCSE conference on Innovation and technology in computer science education. pages 237-241, ACM

[15] RSA Security, America Online and RSA Security Launch AOL PassCode Premium Service, 2004, http://www.rsasecurity.com/

[16] RSA Security, Protecting Against Phishing by Implementing Strong Two-Factor Authentication, 2004, https://www.rsasecurity.com/products/securid/whitepapers/

[17] Sebastian Gajek, Hans Löhr, Ahmad-Reza Sadeghi, TruWallet: Trust-worthy and Migratable Wallet-Based Web Authentication, STC'09, November 13, 2009, ACM

[18] SpamAssasin, http://spamassassin.apache.org/

[19] SpoofGuard, http://crypto.stanford.edu/SpoofGuard/

[20] SpoofStick, http://www.spoofstick.com/

[21] Troy Ronda, Stefan Saroiu and Alec Wolman, iTrustPage: A User-Assisted Anti-Phishing Tool, EuroSys'08, 2008, ACM

[22] TrustE, http://www.truste.org/

[23] Visa, Verified by Visa, http://www.visa.com/

[24] Waterken Inc., Waterken YURL Trust Management for Humans, http://www.waterken.com/dev/YURL/Name/

[25] "What is spear phishing?", http://www.microsoft.com/athome/security/email/spear_phishing.mspx

[26] Wikipedia, http://www.wikipeida.org/

[27] Wu, M., Miller, R. C., and Garfinkel, S. L. 2006. Do security toolbars actually prevent phishing attacks?, Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (Montréal, Québec, Canada, April 22 - 27, 2006), ACM Press.

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

[29] A. Y. Fu, W. Y. Liu, and X. T. Deng, "Detecting Phishing Web Pages with Visual Similarity Assessment Based on Earth Mover's Distance (EMD)". IEEE Transactions on Dependable and Secure Computing, Vol.3, No.4, 2006, pp.301-311, IEEE.

[30] Huajun Huang; Junshan Tan; Lingxi Liu; Countermeasure Techniques for Deceptive Phishing Attack, New Trends in Information and Service Science, 2009. NISS '09, 2009 , Page(s): 636 – 641, IEEE

[31] Knickerbocker, P.; Dongting Yu; Jun Li; Humboldt: A distributed phishing disruption system, eCrime Researchers Summit, 2009. CRIME '09.2009 , Page(s): 1 – 12, IEEE.

[32] Yu, W.D.; Nargundkar, S.; Tiruthani, N.; PhishCatch - A Phishing Detection Tool , Computer Software and Applications Conference, 2009. COMPSAC '09, 2009, Page(s): 451 – 456, IEEE.

[33] Engin Kirda and Christopher Kruegel, "Protecting Users against Phishing Attacks with AntiPhish", 29th Annual International Computer Software and Applications Conference (COMPSAC'05), 2006, IEEE.

[34] JungMin Kang; DoHoon Lee; Advanced White List Approach for Preventing Access to Phishing Sites, Convergence Information Technology, 2007, Page(s): 491 – 496, IEEE.

[35] Qiong Ren; Yi Mu; Susilo, W.; SEFAP: An Email System for Anti-Phishing, Computer and Information Science, 2007. ICIS '07, 2007, Page(s): 782 – 787, IEEE.

AUTHORS

M. Nirmala received the M.Tech in Computer Science and Engineering from Vellore Institute of Technology and is working towards her PhD. She is currently an Assistant Professor in the School of Information Technology and Engineering at VIT University, Vellore. Her research interests include Computer and Network Security, High Performance Computing and Software Engineering.

K. Naveen Kumar is currently pursuing his M.S in Software Engineering at VIT University where he focused on Software Engineering methodologies, Computer Networks and Security, and Computer fraud and Security. He is a CISCO Certified Network Associate. His research interests include Intrusion Detection and Prevention Systems, Malware Analysis, Reverse Engineering and Software Architecture and Design. His previous projects include, developing a Malware detection tool, Web automation tool for functional testing and he is currently working on creating new authentication mechanisms to prevent fraud and identity theft involved in online transactions.

L. D. Dhinesh Babu received the M.E in Computer Science and Engineering from the University of Madras in 2001 and is working towards his PhD at VIT University. He is currently the Division Leader for Software Engineering Division in the School of Information Technology and Engineering at VIT University, Vellore, India. He is also the Programme Manager for M.S.(Software Engineering) at VIT University. His research interests include Cloud, Grid and Distributed Computing, Computer and Software Security, Software Engineering, Design Patterns and Image Processing.