Gemini: An Emergency Line of Defense against Phishing Attacks

Zhang Xu
College of William and Mary
Email: zxu@cs.wm.edu

Haining Wang
University of Delaware
Email: hnw@udel.edu

Sushil Jajodia
George Mason University
Email: jajodia@gmu.edu

Abstract— **This paper proposes a simple but very effective approach called Gemini to prevent victim users from exposing sensitive credentials to a phishing site. As an emergency line of defense, Gemini assumes that a victim user is already deceived into a phishing site and starts the user authentication procedure. Gemini springs into action once the username field is filled in, and tackles the phishing problem from a new perspective. In particular, by exploiting username input, Gemini is able to provide more accurate detection of a phishing site and much stronger protection for a password, the most confidential and crucial information for user authentication. To validate the efficacy of Gemini, we implement different prototypes of Gemini as a browser extension for IE, Firefox, and Chrome, respectively, and conduct extensive live experiments over various legitimate and phishing websites for more than one month. Our experimental results show that Gemini can achieve zero false negative rate and less than 1% false positive rate, and Gemini can effectively block the access to a phishing site before a victim user begins to enter in a password. Moreover, Gemini is complementary to existing anti-phishing tools. The performance overhead induced by Gemini is minor and has a negligible effect upon users' browsing activities.**

I. Introduction

Aiming at stealing Internet users' credentials, phishing plays an instrumental role in many online frauds. In a phishing attack, a phisher sets up a fake website that imitates a corresponding legitimate website. Then the phisher lures users to visit the phishing site by publishing advertisements or sending out spam, and acquire the victim users' credentials like passwords. Although numerous defense mechanisms have been developed, phishing attacks are still rampant on the Internet nowadays and cause significant financial loss to victim users. There were nearly one million phishing URLs online reported during 2012 [8], and there were more than 50,000 newly booted phishing sites per quarter of 2012 [9], [10]. According to the RSA report, during the first eight months of 2012 there were over 30,000 successful phishing attacks globally every month [11], resulting in a total loss of more than $687 million [12].

As the most common way for user authentication, the text-based username and password are the two key user credentials. While username is less sensitive information and sometimes even publicly available to a third-party like an email address, password is much more confidential and crucial information for user authentication. A strong protection to keep password from falling into a phisher's hand is essential to defend against phishing attacks. Thus, various anti-phishing tools have been proposed to achieve this goal, such as embedding a user-selected image onto the login form and password hashing [30]. Complementary to all these existing anti-phishing solutions, we propose an emergency line of defense called Gemini to block victim users from releasing passwords to a phishing site. Here "emergency" means that a victim user is already deceived into a phishing site and starts the user authentication procedure.

Instead of detecting a phishing site based on its appearance including contents and URLs, we leverage the important information a victim user already typed in—username—for more accurate phishing detection. For a legitimate website that requires online authentication, each registered user must have a unique username. The tuple of {username, domain name} provides very useful information for detecting phishing sites. Given a complete list of {username, domain name} pairs, if the username in the current login form is a valid username but the currently visited domain name does not match any corresponding domain names associated with the username in the list, we can infer that the currently visited website is a phishing site with very high confidence. For example, say a user has ”monkey” as the username on the legitimate websites of domainA.com, domainB.com, and domainC.com. When the user is detected to use ”monkey” as the username in the login form of a newly appeared website domainX.com, it is highly likely that the website of domainX.com is a phishing site.

Based on the observation above, we develop a browser extension called Gemini to protect victim users from phishing attacks. To make Gemini work well in the real world, we have to build a complete list of {username, domain name} mappings. We first initialize the mapping list by collecting the majority of ground truth data and then continue to accumulate the newly appeared and least-frequently-used mappings while Gemini is in action. Thus, Gemini can obtain most mappings within a short period and keep track of newly appeared and least-frequently-used mappings throughout its lifetime.

To validate the efficacy of Gemini, we implement different prototypes of Gemini as a browser extension for IE, Firefox, and Chrome, respectively, and conduct extensive experiments over various legitimate and phishing websites. Our experimental results show that Gemini can achieve zero false negative rate and less than 1% false positive rate; and Gemini can effectively block the access to a phishing site before a victim user begins to enter a password. Moreover, Gemini is transparent to users and complementary to existing anti-phishing tools. The induced overhead of Gemini is minor and has negligible effect upon user browsing.

The remainder of the paper is organized as follows. Section 2 surveys related work. Section 3 describes the basic idea of Gemini and presents our study on user login behaviors and features of login pages. Section 4 details the design of Gemini, while Section 5 presents its implementation. Section 6 shows the evaluation results of Gemini, and finally Section 7 concludes the paper.

II. Related Work

To defend users against phishing attacks, both industry and academia propose various techniques to identify phishing websites and protect users' passwords.

Gemini provides a defense against phishing during the user login process. Similarly, some anti-phishing techniques have been developed that aim at securing the login process to prevent users from submitting their credentials to a phishing site. Some legitimate sites place security indicator on their web pages or URL bars for users to easily identify the legitimage sites [31], [35]. For instance, Sitekey [14] has been employed by some online banking sites such as Bank of America [3]. Users can select a personal image, i.e. a ”key” that will be presented to users every time they attempt to log in. If the key is absent or incorrect, a user is supposed not to continue the login process. However, recent user studies demonstrate that in the real world even if such a security indicator is absent or mis-presented, few users refrain from entering their passwords [31].

Dhamija et al. [17] proposed ”dynamic security skins” to enable a remote server to provide an identity that can be easily verified by users. A phishing site will fail to prove itself and therefore users can easily detect the phishing site during the login process. Parno et al. [28] developed an anti-phishing tool using trusted devices, such as smart phones, for mutual authentication. By enforcing trusted authentication, even if a user runs into a phishing site, their approach can prevent phishers from stealing user credentials. Compared with these existing defenses, Gemini does not require additional devices and is more transparent to users.

Similar to our work, some previous anti-phish systems such as Antiphish [22] and Webwallet [37] aim at identifying the true intention of user browsing to help users be aware of phishing attacks. Different from these approaches, instead of using password information, we exploit the input of username to activiate the anti-phishing procedure to prevent users from entering their passwords. Moreover, Gemini is capable of coping with more comprehensive real world scenarios and is more user transparent.

Some previous anti-phishing technqiues are able to protect users' credentials even after a user submits its credentials to a phishing site. Yue et al. [39] introduced a transparent way to protect credentials submitted to a phishing site by concealing the real credential among bogus credentials. Their approach can make the phisher hardly extract the real credential before victims are alerted by their legitimate sites. Some password manager tools such as PwdHash [30], Password Multiplier [21], and passpet [38] provide password hashing for enforcing password strength. By salting users' passwords with domain specific features such as site name, a phisher cannot reuse the stolen password on a legitimate site due to the different hashing. Unfortunately, a usability study [16] reveals that the effectiveness of these password manager tools is not as high as expected and there exist several usability problems.

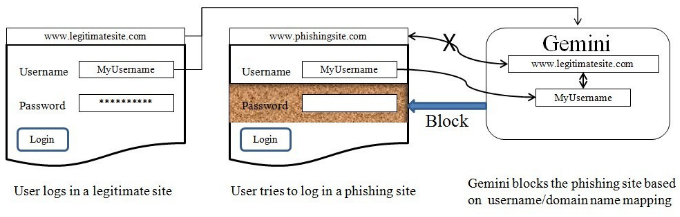

Fig. 1: The basic idea of Gemini

Florêncio et al. [19] proposed a scheme to rescue the passwords stolen by phishers through client-side reporting and server-side aggregation. However, their solution takes effect only after a certain number of users become victims of a phishing attack. Birk et al. [15] proposed a technique to inject fingerprinted credentials to phishing sites in order to trace those stolen credentials and reveal the phisher's identity.

There are several large online databases such as phish tank [8] and OpenDNS [6] that maintain lists of reported phishing sites. These blacklists are populated by mainstream browsers [4], [5] and commercial security tools such as umbrella [13]. Once a website is detected to be within a blacklist, the site will be blocked and users will be alerted. There are also some studies to enhance the effectiveness of blacklists [18], [33].

The URL information and page contents have been widely used to identify a phishing site. The structural, lexical features, as well as host-based features, such as IP address and time of registration of URLs within a domain, can be used to classify legitimate sites and malicious sites [20], [25], [26], [32]. Some existing approaches are able to detect a phishing site based on its content information, such as lexical features, layout similarity with legitimate sites, and content anomaly [27], [29], [34], [40]. Many machine learning based approaches can identify phishing sites based on both URL information and page contents [24], [36].

III. Gemini: an Emergency Line of Defense against Phishing Attacks

The usability studies have shown that with the existing phishing detection techniques, vulnerable users still fall into the trap and visit a phishing site. With the goal of preventing these vulnerable users from further exposing their passwords to a phishing site, we propose the Gemini system as an emergency line of defense to detect the phishing site in a timely fashion and then block the password field from user access.

A. *Basic idea of Gemini*

As illustrated in Figure 1, the main idea of Gemini is simple: a user will always have a username (or multiple usernames) corresponding to a legitimate website. Since a phishing site cannot make its domain name identical to the corresponding legitimate site, by checking the pair of {username, domain name} appearing on a website, we can identify whether the website is a phishing site in an accurately and timely manner.

Despite of the simple idea, Gemini needs to address the following technical challenges.

1) What are the inherent features of a login page? To extract the username information, Gemini first should correctly identify a login page. Unfortunately, due to the various designs and implementations of login pages, there is no precise way to identify a login page at present. To make Gemini work well, we should seek a reliable way to identify a login page and make Gemini tolerant to inaccurate login page identification.
2) How to initialize Gemini with a complete list of {username, domain name} pairs, i.e., how to collect ground truth data? Gemini works based on an accumulative user history data collection, and a clean initialization process is required to garner legitimate {username, domain name} mappings.

To handle these challenging tasks well, we conduct a survey on user login behaviors and systematically characterize various login pages including both legitimate ones and phishing ones.

Question	Number of Users
How many online accounts do you have?	
10 and below	6
10 to 30	32
30 to 50	5
50 to 100	5
100 and above	2
How many distinct usernames do you have?	
5 and below	8
5 to 20	35
20 to 50	6
50 to 100	1
100 and above	0
Can you recall all of your account names all at once?	
Yes	2
No	48
How often do you log in your least-frequently-used account?	
Every week	0
Every two weeks	3
Every one month	18
Every three months	8
Every half a year	20
Every a year and above	1
Will you always input username and password before typing in verification code?	
Yes	50
No	0

TABLE I: The results of user behavior survey

B. A survey on user login behaviors

We conduct a user-behavior survey of 50 web users. These users include ordinary workers, non-computer science students, computer science major students, and professional IT engineers. Table I shows the questions asked and the users' responses. These questions are designed to provide some useful guidelines for the design and implementation of Gemini.

The survey results show that users usually have a moderate number of online accounts and usernames. Among the 50 users, 43 of them have fewer than 50 online accounts and 49 of them have fewer than 50 distinct usernames. This indicates that the information Gemini needs to record is small. Thus, the search through the {username, domain name} mapping list should be a swift process and the memory consumption of Gemini should be insignificant.

The survey results also indicate that collecting ground truth data for Gemini can hardly be a one-time task, as 48 out of 50 users cannot recall all of their usernames and corresponding domain names. Therefore, it is impractical to require users to complete the mapping information manually. The collection of ground truth data should be done in an automatic and accumulative manner. Moreover, many users have accounts that will be accessed very rarely, and 29 out of 50 users will log in their least-frequently used accounts every three months or even longer. This implies that the collection of ground truth data should persist for the lifetime of Gemini.

All 50 users admit that they will always input their usernames before a verification code. This observation can provide a guideline for extracting username from a login page. Within a login form, there are some input fields that might be similar to the username field. The verification code field is the most confusing field, since it has the same input type as the username field. According to the survey results, to distinguish a username field from a verification code, we have to identify their locations within a form and the order in which the user types them in.

C. Features of legitimate login sites and phishing sites

To design an effective Gemini system, we need to understand the genuine differences between legitimate sites and phishing sites. We select 50 legitimate login pages and 50 phishing sites for this comparison purpose. The 50 legitimate sites are selected from the most popular websites, including online shopping sites, online banking, college account systems, email, and utility payment websites. The 50 phishing sites are selected from Phishtank [8].

We observe that a browser usually caches the usernames for legitimate websites where a user has successfully logged in. For mainstream browsers, once a user successfully logs in a legitimate website, its username or even password can be cached so that the user does not need to input them again when the site is re-visited. This observation can help to identify a legitimate site: every cached username and its corresponding domain name should be a legitimate mapping.

For both legitimate and phishing sites, a login page always presents some information to navigate user through the browsing. For instance, among the 100 login pages, 92 of them contain the keywords ”username” and ”password” either in page content or in source code. They also commonly contain some keywords such as ”sign in” and ”onlineID”. This observation provides a strong hint to identify a login page: if a page contains certain keywords, it should be considered as a login page.

Some phishing sites place an image of legitimate URL over the address bar so that vulnerable users cannot observe the real URL they are visiting. Such a deceptive URL can easily mislead users if the true domain name is not presented to

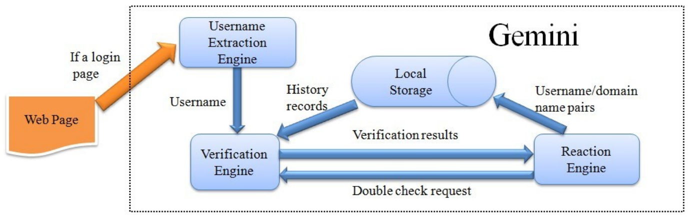

Fig. 2: Architectural Overview of Gemini

users. However, a phishing site can never make its domain name identical to a legitimate domain name, which ensures the {username, domain name} mapping is a reliable way to distinguish phishing sites from legitimate sites. Gemini will never be deceived by the contents or domain names of phishing sites, since it uses both the server side information (domain name) and the client side information (username) that a phisher cannot control.

IV. Design of Gemini

This section describes the architecture of Gemini, the collection of ground truth data in {username, domain name} mappings and the two running states of Gemini.

A. *Architecture of Gemini*

As Figure 2 shows, the basic framework of Gemini consists of four major components: the username extraction engine, the verification engine, the reaction engine, and the local storage.

1) Username extraction engine: At the very first step, Gemini needs to identify whether the current page is a login page. If it is not a login page, Gemini will stop working on this page to reduce overhead. If the current page is verified as a login page, the extraction engine will extract all the potential login forms along with their username fields. The extraction engine will place an event listener at the potential username fields. When a user inputs into the username fields, the extraction engine extracts the username on the fly and passes it to the verification engine.

The identification of a login page is relatively complicated, which will be detailed in Section 5. By contrast, extracting the username fields within a login page is relatively easy. As long as (1) it is a text/email type input field and (2) its location within a form is not after the password field, it will be identified as the username field. It is noteworthy that if a page is identified as a login page, multiple input fields within the page might be identified as possible username fields. For instance, assume there are two forms called form A and form B within a page, only form B contains the password field. In this case, all the text/email type input fields within form A are also considered as potential username fields. Such a design is to capture the username field in some specially crafted phishing pages. For legitimate sites, this design will not mis-identify a user input field. On a legitimate login page, the username field must be always placed before the password field and some other inputs such as verification code are always located after the password field. More importantly, such an order remains intact even in these specially crafted phishing login pages to convincingly imitate the appearance of legitimate websites, and hence our username extraction engine will never miss the username field in a phishing site.

2) Verification engine: The verification engine works in the background and listens to the messages from the username extraction engine. Upon receiving the username from the extraction engine, the verification engine will search the current {username, domain name} pair across the mapping database. If the username has appeared previously but the {username, domain name} mapping does not exist, Gemini will classify the current domain as *suspicious phishing*. Once the verification engine makes the identification decision on the current page, it will notify the reaction engine for further processing. The verification engine also has the obligation to further check the site after receiving a request from the reaction engine. In certain cases, the reaction engine will request the verification engine to further ascertain the sites with some other phishing detection techniques. If other phishing detection techniques also identify the current site as a phishing site, Gemini will classify the current site as *confirmed phishing*.

3) Reaction engine: The reaction engine takes the responsibility of providing proper feedback to a user when a phishing site is detected or allowing the user to continue browsing normally if the site is identified as legitimate. For a newly appeared legitimate site, the corresponding {username, domain name} pair will be recorded into local storage, and the website will be included into the whitelist.

Once a site is identified as suspicious phishing by the verification engine, the reaction engine will block the site and alert a user. As discussed later, Gemini will be in two different states: initialization and in-action. Depending upon the running state of Gemini, the reaction engine will have different interactions with the user. In the initialization state, Gemini will block the suspicious phishing site but allow the user to click a "continue" button to resume the login process. However, in the in-action state, once Gemini locks a suspicious phishing site, the user has to go through a much more complicated unlock process to resume the login process.

4) Local storage: Gemini needs to permanently store information such as {username, domain name} mappings and the whitelist into a local storage. Since these data should be available across the lifetime of Gemini, some traditional data cache mechanisms such as session cookies are unusable because the data will be lost when the browser reboots. Fortunately, HTML5 introduces a new way to permanently store browser data locally in what is called local filesystem/storage [7]. Each browser application can have at most 5MB space to store application-related data. This browser-based local storage is isolated from the disk-based local file system and storage used by other browser applications. Therefore, the data are well protected from other malicious extensions or malware.

B. *Ground truth data collection*

According to our survey, it is very difficult, if not impossible, to make users recall all their usernames and corresponding domain names when Gemini is installed. Therefore, we have to complete the collection of {username, domain name} mapping data in an accumulative manner. Such ground truth data collection mainly involves how to handle those usernames that appear for the first time. Here we classify "first time event" into three categories.

The first category covers the cases where the username is brand new but the target site is a site where the user has logged in successfully with a different username before. In this case, we treat the target site as a trusted site and record the new {username, domain name} pair.

The second category often takes place when a user has used the machine and browser for a period of time before installing Gemini. Since the user has logged into the legitimate sites before, the browser may cache the username. Therefore, although the username and domain name are new to Gemini, as long as the username is correctly cached, the user does not need to input the username manually. The same situation applies to those websites with auto-login functionality. Our keyboard monitoring and cookie monitoring components can easily identify such cases. These sites with cached usernames will be regarded as legitimate and the {username, domain name} pairs will be recorded.

The third category includes the cases where both the username and domain name are new and the username is not cached. In this scenario, we classify the current mapping as "pending mapping". In general, a user will not be deceived to log into a phishing site before he has ever owned an account at the corresponding legitimate site. However, in case that a user logs into a phishing site before logging into the legitimate site, Gemini should double check the site in the background with other anti-phishing tools. If the site is found to be a phishing site, the user will be alerted and the site will be blocked. If the site is verified not to be a phishing site, the current mapping will become a "candidate mapping". A candidate mapping will become a legitimate mapping only when the same username appears on the same website for the second time. It is noteworthy that the username in the candidate mapping is also considered as an appeared username. Every time the verification engine searches a username across the database, the current username will be compared with these usernames in both legitimate mappings and candidate mappings.

C. Running states of Gemini

According to our survey, in general users log into most of their online accounts over a relatively short time span. However, there are some legitimate accounts that users log into less frequently, maybe every several months or even longer. Based on such observations, Gemini consists of two different states. One short-period state is set for Gemini to gather most of the ground truth data, which is called the initialization state. Once the initialization is done, Gemini runs into a long period of the in-action state.

1) Initialization state: The initialization of Gemini is to gather the majority of the ground truth data in a relatively short period of time. According to our survey, many users use duplicated usernames for different online accounts. Such username duplication might lead to a false positive when an unpopular legitimate site (not in the whitelist) is logged into for the first time with a duplicated username. In general, the frequency of account access should follow a long-tail distribution. The most frequently used accounts should be email accounts, job-related accounts, social networking accounts, online shopping accounts, and online banking accounts. Many of these accounts, such as email accounts and job accounts, are logged in daily while the others, such as online shopping accounts, are likely be accessed every week. These frequently accessed accounts are the most attractive targets for phishing attacks. This observation implies that Gemini will encounter the "first time events" mostly within a short period after it is installed and the majority of false positives should occur during this period as well.

Therefore, Gemini performs its initialization in a short time span using simple user interaction. The Initialization state only lasts for one week to two weeks. The user interaction in initialization is a simple click-through alert. When a page is identified as suspicious by the verification engine, the site is temporarily blocked and an alert page will appear and inform a user that the username is new to this site and the domain name is highlighted for the user to confirm. If the user can ensure that this site is trusted and wants to continue the login process, it can click the "continue" button. Meanwhile, Gemini will double check the temporarily blocked site in the background using some other anti-phishing techniques like blacklist checking to lower the user's risk of falling into a phishing trap. Only if the double check process does not identify the site as a phishing site can the user continue browsing and complete the login task.

2) In-action state: After the initialization, Gemini will enter the in-action state. In this state, Gemini should have already collected sufficient ground truth data and the users should have logged in most of their online accounts. Therefore, Gemini now can identify a phishing site with higher confidence and accuracy than its initialization state. Instead of using the simple click-through user interaction in the initialization state, Gemini recruits more rigorous user interaction when a suspicious site is detected.

When Gemini is in action, once a site is identified as suspicious, the site will be "locked". A user has no direct option to continue normal browsing on the current page. Every time when the user inputs its username or attempts to input its password, Gemini will pop up alerts and prevent users from accessing the login fields. In this way, a user can view the locked page but cannot input its credentials. To unlock the page, the user has to input an "unlock code" provided by Gemini into the username field of the page and submit the form. During this process, the user will be alerted to assure the legitimacy of the current site for several times and Gemini will leverage other anti-phishing tools for further confirmation.

D. Exceptional user behavior handling

Although in most cases Gemini can easily extract a username from a login page and verify the validity of the current website, there are some exceptional user input behaviors that require Gemini to use special handling.

The natural user logging behavior should consist of typing in username before password. However, in very rare cases, a user might type in password before username. If a user types in password first and the phishing site places a keyboard monitor on the password field, the attacker will obtain the user's password before Gemini can react.

To address this special situation, Gemini will force users to type in username before password. On a login page, the password field will be temporarily locked by the username extraction engine before the username field is filled. As mentioned above, a legitimate site might cache the username for a user so that the user does not have to input the username manually. Therefore, as long as Gemini detects that the username field is filled, it will unlock the password field.

It is also very rare but still possible that a user types in a wrong username with a correct password and then submits the pair of credentials without being aware of it. In such a case, the phisher can acquire the incorrect username and correct password. If the phishing site can verify the validity of the credential pair in realtime and further prompt the user for login input again, the phisher could obtain the correct username via the keyboard monitor while the user is making the second login attempt.

To address this exception, once a new username is detected, the verification engine of Gemini will calculate the similarity between the current username and the existing usernames by employing Levenshtein distance [2]. If any distance within 2 is detected, we will regard the current username as an incorrectly typed username and remind the user to assure the correctness of the typed username.

V. Implementation

We implement the prototypes of Gemini as a browser extension for Firefox, Chrome and IE respectively. In this section, we first give a detailed description on how to identify a login page, which is one of two major challenges for Gemini to address. Then, we present the technical details to achieve the portability of Gemini and highlight the cooperation of Gemini with other anti-phishing defenses.

A. *Identifying a login page*

As the first step of Gemini, it is essential to accurately identify a login page. For a legitimate website, it is relatively straightforward to identify a login page, since the login page should contain at least the password field and a form to submit. Thus, to identify a login page, we can just search the source of the page for the segments of "<form" and "type='password'". This method can also identify most of the login pages in the phishing sites.

Gemini works well in all frames or iframes within a page. Therefore, even if the login page is within a pop-up window, Gemini can successfully capture it.

1) Special designs of phishing log in page: Although most of the login pages in the phishing sites have the same or very similar design as those in the legitimate sites, some phishing sites can have special designs in an attempt to evade detection. Gemini should cover those cases where a phishing site has a specially crafted login page.

```
<form>
  <input type="hidden" name="secret" value="">
  <input type="text" name="mockingPwd" onKeyPress="
   this.form.secret.value += String.fromCharCode(event.keyCode);
   event.keyCode = 183;">
</form>
```

Fig. 3: An example of how a phishing site can mock a password field.

A phishing site does not necessarily contain a password field. Some phishing sites use the javascript to mock a password field. Figure 3 shows such an example. The phishing site sets up a text field and attaches a keyboard listener to it. No matter what character a user types in, the character will appear as an asterisk or dot mark in the page. In this way, the user will consider this text field as a password field. Similarly, a phishing site can place isolated input fields that are not wrapped by any form. The event listener attached to the input fields or an enclosing tag can send the input information directly to a phisher without the user submitting any form.

To detect such a special phishing login page, we need to ascertain whether an input field is deployed with an event listener. If any of the input fields or tag enclosing input fields is deployed with an event listener, the page is classified as a login page.

A phishing site could forge username or password fields in a way that <input> tags will not appear in the source, resulting in the feint that there is no input field on the page. For instance, <div contenteditable=true></div> can function as input fields. To handle this case, Gemini will identify such pages with "fake input fields" as login pages as well.

A phishing site can also isolate a username input page from a password input page. The phishing site can first ask a user to input its username, and then direct the user to the next page to input its password. In such a scenario, we need to use a more aggressive identification algorithm. We will search for the keywords such as "log in", "sign in", "username", and "passcode" on the page. If a page contains any form and certain combination of these keywords, the page is also classified as a login page. Note that a few legitimate sites such as Bank of America also isolate the username input from the password input in order for users to verify the site before entering a password.

B. *Portability*

To make Gemini portable to multiple user-owned machines, the implementation of Gemini enables the collected ground truth data to be imported or exported from one machine to another machine. For user privacy, the exchange of user data is encrypted. During the installation of Gemini, a welcome page will remind a user to initialize a "key". Then the key will be used to encrypt and decrypt the user data. Only after the user confirms the key, will Gemini start running on the browser.

Since Gemini works purely at the client side, to export a file, Gemini creates a file on the fly using a DataURL scheme

	IE	Firefox	Chrome
Login	329	1611	1481
Input	62	254	312

TABLE II: Statistical data of real world user study

	IE	Firefox	Chrome	Overall
Login fp	2(0.68%)	2(0.68%)	2(0.68%)	0.68%
Phish fp	2(3.23%)	1(0.39%)	1(0.32%)	0.64%

TABLE III: The false positive rate of Gemini.

and stores the file in local memory. At the same time, Gemini provides the link to the data URL so that the user can download the file to local disk. However, IE does not support DataURL through version 7 while IE 8 and 9 can only use DataURL for images [23]. Therefore, Gemini has a different implementation for IE. In the IE environment, ActiveXObject is used to create the file. By contrast, importing a file is relatively easy; simply reading the file contents and storing into the extension database will complete the task. The export and import functions are added into the context menu (Firefox and Chrome) or side bar (IE) for users to easily use.

C. Cooperation with other defenses

As an emergency defense line, Gemini is complementary to existing defenses and can work seamlessly with existing anti-phishing tools, such as blacklist and whitelist. We recruit the blacklist during the double checking process so that Gemini can prevent users from unexpectedly continuing to browse on a confirmed phishing site. Meanwhile, the use of whitelist significantly helps Gemini in validating the legitimacy of a new site. Currently, Gemini is deployed with the Phish Tank blacklist [8] and the whitelist from Alexa [1]. The other more advanced defense techniques can also be used to enhance the effectiveness of Gemini.

VI. Evaluation

To validate the efficacy of Gemini, we conduct two sets of experiments on both legitimate and phishing web sites. In the first set of experiments, we evaluate the effectiveness of Gemini against phishing attacks in terms of false positive rate and false negative rate in real-world environments. In the second set of experiments, we investigate the performance impact induced by Gemini upon a running browser and its user.

A. Effectiveness against phishing attacks

Here we first describe our experimental methodology, and then present the experimental results in detecting phishing sites.

1) Methodology: The experiments to evaluate Gemini's effectiveness can be further divided into two groups: one for basic functionality evaluation and the other for real world user study.

The basic functionality evaluation is to verify whether Gemini can correctly identify a login page. We select 50 legitimate websites with a login page, 50 phishing sites with a login page, and 300 non-login regular web pages. The 50 legitimate login pages are selected from top visited sites [1], while 50 phishing login pages are selected from Phishtank [8]. The 300 regular web pages are also selected from top website list [1]. We visit these web pages with IE, Firefox, and Chrome installed with Gemini and test whether Gemini can identify these login pages and regular pages correctly.

	IE	Firefox	Chrome	Overall
Login fn	0(0%)	1(1%)	0(0%)	0.33%
Phish fn	0(0%)	0(0%)	0(0%)	0%

TABLE IV: The false negative rate of Gemini.

	Toshiba Laptop
CPU	2*AMD V120, 2.20 GHz
Memory	2*1GB DDR2
Operating System	Windows 7 Premium

TABLE V: The configuration of testbed

To thoroughly evaluate the effectiveness of Gemini in real world, we invite 20 active web users to conduct real world user studies. Among the 20 volunteers, we have 2 users using IE under Windows, 3 users using Firefox under Windows, 3 users using Chrome under Windows, 3 users using Firefox under Linux, 3 users using Chrome under Linux, 3 users using Firefox under Mac OS, and 3 users using Chrome under Mac OS. All these users are asked to install Gemini in their most frequently used browsers. The users are notified about the functionality of Gemini, but we do not teach the users how to use Gemini in detail. Instead we provide documentation and user guidelines along with Gemini for the users to read when necessary. This is to ensure that the users are acting exactly as if Gemini is an extension they download from the webstore and use it from scratch as normal.

The real world user study consists of three phases: the training phase, the safeguard phase, and the anti-phishing test phase. The training phase corresponds to the initialization state of Gemini. The initialization state of Gemini is set to 10 days during our user study. The safeguard phase, in which Gemini is in action, lasts for 30 days in our user study. During the training and safeguard phases, we keep track of the false positives of Gemini. When Gemini wrongly identifies a legitimate site as a phishing site, a false positive is reported. The users are not aware of the training and safeguard phases since all they have to do is to browse websites and log into their accounts normally. Nevertheless, Gemini did work in full action to protect users from phishing attacks. As reported by one user, the user was deceived to a real phishing site during the evaluation. Thanks to Gemini, that phishing site was detected and blocked in time.

The anti-phishing test phase is different from the former two phases. In this phase, every user is provided with 20 phishing sites corresponding to the legitimate sites on which the users have online accounts. The users are directed to make an attempt to log in these phishing sites with their usernames but our provided fake passwords. Note that the users have the right to decide when the anti-phishing test is conducted as long as Gemini is in the in-action state. We record the results of their attempts to log into phishing sites. If Gemini fails to block or lock a phishing site before a user types in password, then a false negative is reported. To protect users' privacy, we only record statistical data and we do not retrieve any account information

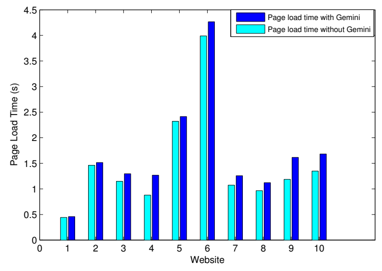

Fig. 4: The page load time of 10 websites on IE before and after Gemini is installed.

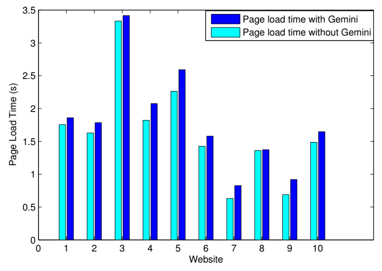

Fig. 5: The page load time of 10 websites on Firefox before and after Gemini is installed.

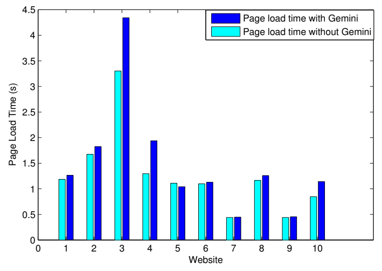

Fig. 6: The page load time of 10 websites on Chrome before and after Gemini is installed.

or browsing footprint of the users.

Table II lists some statistical information of the user study, including how many times users log into their accounts (login) and how many times users manually input their usernames and passwords to log in their accounts (input).

2) False positives: There are two types of false positives: (1) incorrectly identifying a non-login page as a login page and (2) incorrectly identifying a legitimate site as phishing site. Table III shows the false positive rates of Gemini in different browsing environments, IE, Firefox, and Chrome, respectively.

In the identification of a login page, to detect those specially crafted phishing sites, our login page identification algorithm is designed to be aggressive. Therefore, it introduces a few false positives. For instance, the Wikipedia page about "password" is taken as a login page since the Wikipedia page provides a form for users to type in search words and the page contains sensitive keywords of "password", "username", and "log in". However, from the results we can see that the false positive rate is still very low (below 1%). Moreover, such a false positive in the login page identification does not interfere with users' normal browsing activities and only induces negligible performance overhead as shown in Section 6.2.

In the identification of a phishing site, only few legitimate sites are wrongly classified as phishing sites. The overall false positive rate is below 1%. The cause of these false positives is due to the facts that (1) several users logged into some small websites that were not included in our whitelist; and (2) it happened that the usernames they used were the same as the usernames they used on the other legitimate sites. The overwhelming majority of the false positives occur in the training phase. As reported by the users who experienced a false positive, Gemini did draw their attention and they carefully checked the site before they continued to log in. All these users successfully continued to browse and log in.

3) False negatives: There are also two false negatives: (1) failing to identify a login page and (2) failing to identify a phishing site. Table IV shows the false negative rates of Gemini when identifying login pages and phishing sites, respectively, under different browsing environments.

From the table, we can see that Gemini is able to achieve very low false negative rate in the identification of a login page. The only false negative appearing on Firefox is due to a networking problem. The login-related contents of the page were not correctly downloaded so that Gemini failed to recognize it. Frankly, this false negative is not caused by any technical problem of Gemini. If the user further refreshes the page in order to conduct a normal login procedure, Gemini can still successfully detect the login page after it is downloaded correctly. Meanwhile, our experimental results clearly show that Gemini is able to identify all the phishing sites with zero false negative, indicating its effectiveness against phishing attacks.

B. Performance impact

As a client side approach, the performance overhead of Gemini should be as low as possible to avoid degrading user experience. We conduct a series of experiments to measure the overhead of Gemini at the client side. Due to the wide use of laptops running Windows as personal computing platforms, all these experiments are conducted on a laptop with Windows OS. The experimental configuration is listed in Table V.

1) Page load time: Gemini works in a per-page granularity, and its overhead will increase page load time. To evaluate the additional latency induced by Gemini in loading a page, we select 10 popular sites, each with a login page, and make Gemini work in full action on them. We measure the page load time of these websites on IE, Firefox, and Chrome before and after Gemini is installed, respectively. Since the overhead of Gemini is on parsing and displaying pages, the measured load time is the time for a browser to fully load a cached web page. In this way, we can eliminate the network effects on our results. For each site, we test its load time for 10 times and take the average.

Figures 4 to 6 illustrate the experimental results. It is evident that in all the three browsers, Gemini only induces minor to moderate overhead. The average additional latency caused by Gemini is less than 10%. In very few sites, the page load time increased by Gemini is beyond average. This is because the login pages in these sites are very complicated with many user input fields. Thus, Gemini has to do more work including attaching event listeners to many input fields, resulting in relatively high overhead. However, as demonstrated by our results, overall the page load time increased by Gemini is imperceptible to users, and hence does not degrade the user experience.

2) Overhead of phishing verification: When a user inputs its username into a website, Gemini will check the {username, domain name} mapping in the background. It is critical to ensure that the delay caused by the background verification

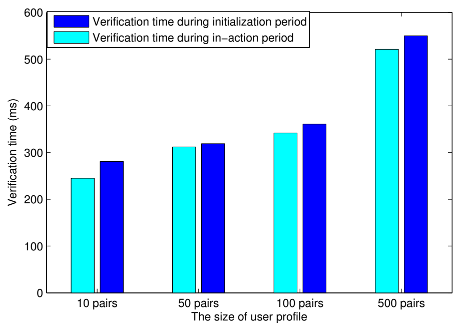

Fig. 7: The time taken to verify a visited site based on {username, domain name} mapping when Gemini is in initialization and in-action state.

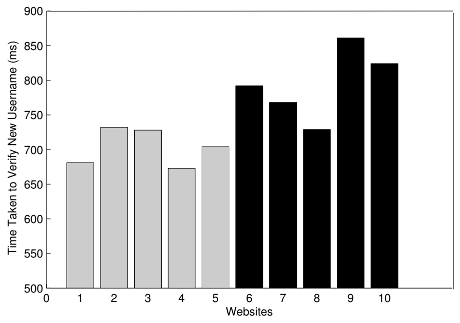

Fig. 8: The time taken for Gemini to verify a site when a new username and a new domain name is detected. The first 5 websites are legitimate sites while the rest 5 are phishing sites. The data size is set to be 50 {username, domain name} mappings.

is insignificant so that Gemini can block or lock the login page before the user enters in its password. We evaluate the overhead of the background verification in two different scenarios: (1) a used username paired with a new domain name and (2) a new username paired with a new domain name. Note that if the domain name is not new, it is most likely a legitimate site and Gemini can make a quick decision simply using the previous classification on that domain.

In the first scenario, the username has appeared before and the verification engine of Gemini verifies whether the current domain name is within the corresponding mapping list. This could happen while Gemini is in either initialization state or in-action state. We measure the time cost of the verification process when Gemini is in initialization state and in-action state, respectively, with different sizes of user data. The size of user data is set to be 10 {username, domain name} pairs, 50 pairs, 100 pairs, and 500 pairs, respectively.

As shown in Figure 7, even with the eccentrically large size of user data of 500 {username, domain name} pairs (i.e., a user has 500 online accounts), Gemini takes only around 0.5 second to complete the verification. The results demonstrate that Gemini can always block or lock a suspicious phishing site before a user enters into the password field. The experimental results here also concur with the results of our real world user study. In all cases, when a user tried to log in a phishing site, Gemini successfully blocked or locked the page before the user could type a single character into the password field.

The second scenario is that both the username and the domain name have not appeared before. In such a scenario, Gemini has to (1) search the username through the database to find that it is new, (2) calculate the Levenshtein distance between the current username and the existing usernames, (3) check in the background whether the site is a phishing site by using the blacklist technique, and (4) either record the current mapping as a candidate mapping or block/lock the current site. Given this scenario, we measure the overhead of Gemini on 5 legitimate sites and 5 phishing sites, respectively. We intentionally input a new username into these websites and record the time for Gemini to complete the verification process. Figure 8 illustrates the time consumed by the verification process, indicating that the verification is time-efficient, so that Gemini can react in time before a user types in a password.

VII. Discussion

Given the efficacy of Gemini against phishing attacks, Gemini still has two major limitations as described below.

Gemini aims at protecting the most crucial and confidential user credential—password—from falling into a phisher's hand. However, Gemini can only detect a phishing site after a user inputs its username. Thus, Gemini does not provide much protection on a user's username. But only knowing a user's username, the phisher has gained little to figure out the corresponding password. Moreover, Gemini can block the phishing sites before a user submits the credential, implying that the only way for a phisher to steal a username is to monitor the username input field. Nevertheless, as username is in plain text, a phishing site does not need to monitor a username input field. For all the phishing sites we have studied, none of them attach a keyboard monitor to a username input field.

There are some non-standard login pages that might evade the detection of Gemini. For instance, if a phishing site constructs a login page with Flash, Gemini will not be able to identify it. However, such a non-standard login form is not popular both for legitimate sites and phishing sites. For legitimate sites, a non-standard login form may cause accessibility and usability problems. For phishing sites, a Flash-made login page will make the host phishing sites themselves different from the imitated legitimate sites, leading to easy detection and prevention. For all the popular websites and phishing sits we studied, none of them are using the Flash login page.

VIII. Conclusion

In this paper, we have presented a browser extension based anti-phishing tool called Gemini to construct an emergency defense line against phishing attacks. Our approach leverages a new source, username, to identify a phishing site with high accuracy and minor overhead. By conducting a survey on user login behavior and characterizing the features of login pages, we have elaborated the design of Gemini to accurately identify login pages including those specially crafted phishing sites and to collect ground truth data. We have implemented prototypes of Gemini as a browser extension for IE, Firefox, and Chrome, respectively. To validate Gemini as an effective and efficient defense against phishing attacks, we have conducted a series of live experiments including a real-world user study lasting for more than one month. Our experimental results show that Gemini can successfully identify all phishing sites while achieving less than 1% false positive rate. The performance overhead induced by Gemini is insignificant and can hardly be perceived by users, indicating that Gemini will not cause any degradation on user browsing experience. For the future work, we plan to integrate more advanced anti-phishing techniques into Gemini to make the overall defense more effective. After more complete quality test, we will place Gemini into a Web app store and we plan to introduce Gemini to APWG or computer security enterprises as well.

REFERENCES

[1] Alexa: Top sites in united states. http://www.alexa.com/topsites/countries/US.

[2] Algorithm implementation of levenshtein distance. http://en.wikibooks.org/wiki/Algorithm_implementation/Strings/Levenshtein_distance#JavaScript.

[3] Bank of america, sign up for the sitekey service. http://www.bankofamerica.com/privacy/passmark/.

[4] Firefox: phishing and malware protection. http://www.mozilla.org/en-US/firefox/phishing-protection/.

[5] Google chrome: phishing and malware detection. http://support.google.com/chrome/bin/answer.py?hl=en&answer=99020.

[6] Opendns. http://www.opendns.com/.

[7] The past, present & future of local storage for web applications. http://diveintohtml5.info/storage.html.

[8] Phish tank. http://www.phishtank.com/.

[9] Phishing activity trends report, 2012 1st quarter. http://docs.apwg.org/reports/apwg_trends_report_q1_2012.pdf.

[10] Phishing activity trends report, 2012 2nd quarter. http://docs.apwg.org/reports/apwg_trends_report_q2_2012.pdf.

[11] Phishing in season: A look at online fraud in 2012. http://blogs.rsa.com/phishing-in-season-a-look-at-online-fraud-in-2012/.

[12] Rsa: Phishing attacks net 687 m to date in 2012. http://threatpost.com/en_us/blogs/rsa-phishing-attacks-net-687m-date-2012-082412.

[13] Umbrella: Block malware, contain botnets and stop phishing. http://www.umbrella.com/explore/internet-security/.

[14] Rsa sitekey solution for enterprise. http://www.RsaSecurity.com, 2007.

[15] BIRK, D., GAJEK, S., GROBERT, F., AND SADEGHI, A.-R. Phishing phishers - observing and tracing organized cybercrime. In *Proceedings of the Second International Conference on Internet Monitoring and Protection* (2007), IEEE, pp. 3–11.

[16] CHIASSON, S., VAN OORSCHOT, P., AND BIDDLE, R. A usability study and critique of two password managers. In *Proceedings of the 15th USENIX Security Symposium* (2006), pp. 1–16.

[17] DHAMIJA, R., AND TYGAR, J. The battle against phishing: Dynamic security skins. In *Proceedings of the Symposium on Usable Privacy and Security (SOUPS)* (2005), ACM, pp. 77–88.

[18] FELEGYHAZI, M., KREIBICH, C., AND PAXSON, V. On the potential of proactive domain blacklisting. In *Proceedings of the 3rd USENIX conference on Large-scale Exploits and Emergent Threats* (2010), USENIX, pp. 6–6.

[19] FLORÊNCIO, D., AND HERLEY, C. Password rescue: a new approach to phishing prevention. In *Proceedings of the 1st USENIX Workshop on Hot Topics in Security* (2006), USENIX, pp. 2–2.

[20] GARERA, S., PROVOS, N., CHEW, M., AND RUBIN, A. A framework for detection and measurement of phishing attacks. In *Proceedings of the 2007 ACM Workshop on Recurring Malcode* (2007), ACM, pp. 1–8.

[21] HALDERMAN, J., WATERS, B., AND FELTEN, E. A convenient method for securely managing passwords. In *Proceedings of the 14th international conference on World Wide Web (WWW)* (2005), ACM, pp. 471–479.

[22] KIRDA, E., AND KRUEGEL, C. Protecting users against phishing attacks with antiphish. In *Proceedings of the 29th Annual International Computer Software and Applications Conference (COMPSAC)* (2005), IEEE, pp. 517–524.

[23] LAW, E. Ie9 beta minor changes list. http://blogs.msdn.com/b/ieinternals/archive/2010/09/15/ie9-beta-minor-change-list.aspx.

[24] LUDL, C., MCALLISTER, S., KIRDA, E., AND KRUEGEL, C. On the effectiveness of techniques to detect phishing sites. In *Proceedings of the 4th international conference on Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA)* (2007), Springer-Verlag, pp. 20–39.

[25] MA, J., SAUL, L., SAVAGE, S., AND VOELKER, G. 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* (2009), ACM, pp. 1245–1254.

[26] MA, J., SAUL, L., SAVAGE, S., AND VOELKER, G. Identifying suspicious urls: an application of large-scale online learning. In *Proceedings of the 26th Annual International Conference on Machine Learning (ICML)* (2009), ACM, pp. 681–688.

[27] PAN, Y., AND DING, X. Anomaly based web phishing page detection. In *Proceedings of the 22nd Annual Computer Security Applications Conference on Annual Computer Security Applications Conference (ACSAC)* (2006), IEEE, pp. 381–392.

[28] PARNO, B., KUO, C., AND PERRIG, A. Phoolproof phishing prevention. In *Proceedings of the 10th international conference on Financial Cryptography and Data Security* (2006), Springer-Verlag, pp. 1–19.

[29] ROSIELLO, A., KIRDA, E., FERRANDI, F., ET AL. A layout-similarity-based approach for detecting phishing pages. In *Proceedings of 3rd International Conference on Security and Privacy in Communication Networks (SecureComm)* (2007), IEEE, pp. 454–463.

[30] ROSS, B., JACKSON, C., MIYAKE, N., BONEH, D., AND MITCHELL, J. Stronger password authentication using browser extensions. In *Proceedings of the 14th Usenix Security Symposium* (2005), USENIX, pp. 2–2.

[31] SCHECTER, S., DHAMIJA, R., OZMENT, A., AND FISCHER, I. The emperor's new security indicators: An evaluation of website authentication and the effect of role playing on usability studies. In *Proceedings of IEEE Symposium on Security and Privacy (S&P)* (2007), pp. 51–65.

[32] SHANMUGHANEETHI, V., ABRAHAM, R., AND SWAMYNATHAN, S. A robust defense mechanism to prevent phishing attack using parse tree validation. In *Proceedings of the 2011 international conference on Advanced Computing, Networking and Security (ADCONS)* (2011), Springer-Verlag, pp. 551–557.

[33] SINHA, S., BAILEY, M., AND JAHANIAN, F. Improving spam blacklisting through dynamic thresholding and speculative aggregation. In *Proceedings of the 17th Annual Network & Distributed System Security Symposium (NDSS)* (2010).

[34] WENYIN, L., HUANG, G., XIAOYUE, L., MIN, Z., AND DENG, X. Detection of phishing webpages based on visual similarity. In *Special interest tracks and posters of the 14th international conference on World Wide Web (WWW)* (2005), ACM, pp. 1060–1061.

[35] WHALEN, T., AND INKPEN, K. M. Gathering evidence: use of visual security cues in web browsers. In *Proceedings of 2005 Graphics Interface (GI)* (2005), Canadian Human-Computer Communications Society, pp. 137–144.

[36] WHITTAKER, C., RYNER, B., AND NAZIF, M. Large-scale automatic classification of phishing pages. In *Proceedings of the 17th Network and Distributed System Security Symposium (NDSS)* (2010).

[37] WU, M., MILLER, R., AND LITTLE, G. Web wallet: preventing phishing attacks by revealing user intentions. In *Proceedings of the 2nd Symposium on Usable Privacy and Security (SOUPS)* (2006), ACM, pp. 102–113.

[38] YEE, K., AND SITAKER, K. Passpet: convenient password management and phishing protection. In *Proceedings of the 2nd Symposium on Usable Privacy and Security (SOUPS)* (2006), ACM, pp. 32–43.

[39] YUE, C., AND WANG, H. Bogusbiter: A transparent protection against phishing attacks. *ACM Transactions on Internet Technology (TOIT) Vol.10 n.2, p.1-31* (2010).

[40] ZHANG, Y., HONG, J., AND CRANOR, L. Cantina: a content-based approach to detecting phishing web sites. In *Proceedings of the 16th international conference on World Wide Web (WWW)* (2007), ACM, pp. 639–648.