Someone in Your Contact List: Cued Recall-Based Textual Passwords

Noura Alomar, Mansour Alsaleh, Abdulrahman Alarifi

***Abstract*—Textual passwords remain the most commonly employed user authentication mechanism, and potentially will continue to be so for years to come. Despite the well-known security and usability issues concerning textual passwords, none of the numerous proposed authentication alternatives appear to have achieved a sufficient level of adoption to dominate in the foreseeable future. Password hints, consisting of a user generated text saved at the account setup stage, are employed in several authentication systems to help users to recall forgotten passwords. However, users are often unable to create hints that jog the memory without revealing too much information regarding the passwords themselves. We propose a rethink of password hints by introducing SỲNTHIMA, a novel cued recall-based textual password method that reveals no information regarding the password, requires no modifications to authentication servers, and requires no additional setup or registration steps. SỲNTHIMA makes use of users' contact lists, so that mapped password hints extracted from a user's contacts are automatically generated while the user is typing the password. We create formal models for relevant aspects of the password hint mechanism, define its threat model, and analyze the security and usability of SỲNTHIMA. We also present the results of an in-lab user study of SỲNTHIMA on 30 participants to evaluate its effectiveness and usability. The results demonstrate that SỲNTHIMA minimizes the number of incorrect login attempts and improves long-term password recall, with acceptable login times and positive user feedback. We summarize the lessons learned from the user study, with the hope of provoking further insights regarding the design of effective cued recall-based textual password schemes.**

***Index Terms*—Authentication, Passwords, Password Hint, Password Recall, Cued Recall-Based Textual Passwords.**

I. Introduction

In the vast majority of authentication systems, textual password schemes are the dominant choice for authenticating end users, despite the well-known security issues concerning passwords, and the inconvenience incurred by end users in remembering multiple passwords for different accounts [1], [2], [3]. Typically, users tend to choose easy-to-remember passwords that are also easy for adversaries to guess [4], [5], [6]. In addition, security vulnerabilities, phishing of credentials, and poor security practices in storing password-related files have led to large-scale security breaches and an ongoing online trade of hundreds of millions of stolen usernames and passwords belonging to various accounts [7], [8], [9], [10]. For example, a massive 272.3 million stolen user names and passwords were recently traded on the internet, including some from the biggest email providers (e.g., Google, Yahoo, and Microsoft) [11]. In fact, passwords are to blame for many recent data breaches. A recent report considered the ineffective use of passwords as a major factor that enabled top IT security attacks in 2016 [12].

End users are often compelled to choose *"strong"* passwords (e.g., through password meters [13]). However, security and usability trade-offs (e.g., password strength vs. memorability and password strength vs. reuse) limit not only the ability of users to create unique and strong passwords for their accounts [14], but also increase the likelihood that users find such processes burdensome and irritating [15], [16], [17]. Furthermore, existing password schemes lack mechanisms providing users with means to narrow down the range of candidate passwords as users are only provided with *"all-or-nothing"* feedback upon submitting login credentials [18]. This in turn can leave legitimate users with no option but to guess their passwords through a *"trial-and-error"* process, submit password recovery requests, or resort to other verification methods to regain access to their accounts. This eventually increases the amounts of cognitive effort, time, and resources required to memorize new passwords, or to contact system administrators and request their help in resetting passwords.

An extensive body of literature exists regarding alternative to passwords (e.g., [19], [20] and [21]), although very few alternatives have been adopted for commercial use, and none have become more popular than textual passwords. In part, this is due to deployability and usability issues concerning such alternatives, and a lack of knowledge regarding the direct financial and data losses resulting from password-related security issues [1], [22]. Furthermore, the majority of authentication systems (e.g., online services) may face greater challenges in adopting new authentication solutions that are less likely to achieve widespread adoption, as these would be unfamiliar to most users.

Given the persistence of passwords, at least in the near future, it seems prudent to invest considerably greater research effort in minimizing security and usability issues regarding textual passwords [3], [23]. Using password hints to help users recall passwords is a widely adopted technique in many authentication systems (e.g., Windows and iOS user authentication). In most hint schemes, the user is required to submit some text (either a word or a sentence) when setting up the account, which can then be used to jog the user's memory at the login stage if the password is subsequently forgotten. The positive impact of employing personalized and self-chosen cues for jogging users' memories has also been demonstrated by Smith et al. [24].

The main dilemma concerning such password hints is that the more information a user's chosen hint provides regarding

Version: May 27, 2017.

N. Alomar is with the College of Computer and Information Sciences, King Saud University, Riyadh, KSA, e-mail: nnalomar@ksu.edu.sa
M. Alsaleh and A. Alarifi are with King Abdulaziz City for Science and Technology, Riyadh, KSA, emails:{maalsaleh, aarifi}@kacst.edu.sa

the password, the more likely it is that an attacker can guess the password by reading the password hint. While some user-generated hints may be very helpful in assisting the users in remembering their passwords, we argue that a good password hint not only assists with recalling a password, but also reveals little or preferably no information regarding the password.

Motivated by the expected persistence of textual passwords, as well as the importance of mitigating security and usability issues relating to them, this paper proposes a novel password hint scheme called SỲNTHIMA[1], which makes use of a user's contacts list, which constitutes an available and familiar information source to the user, to automatically generate an on-the-fly, easy-to-remember password hint that is learned upon the first login. To use SỲNTHIMA, a user must only mentally associate contact names from her contact list with the correct passwords. Other than setting up the SỲNTHIMA application at the installation stage, no additional setup or registration steps are required for a new user account. Contrary to classic password hint schemes, SỲNTHIMA reveals no information regarding the password in question, and requires no modifications to authentication servers. Moreover, SỲNTHIMA stores no information regarding user passwords or account details, and maintains no look-up tables or any mapping data between passwords and corresponding hints. Rather than creating a hint that somehow relates to the chosen password, and thus might provide clues for guessing the password, SỲNTHIMA helps the user to minimize the number of *incorrect login attempts* while improving long-term password recall.

By associating passwords with names of people from users' contact lists, we propose the first mechanism that aids the retrieval of textual passwords from human memory by presenting unique hints while users are typing their passwords. Based on the *associative-strength theory* [25], and because users of SỲNTHIMA would be able to associate their passwords with hints each time they successfully log into a given system, we propose that stronger ties between passwords and cues can be developed in users' memories as the frequency of successful logins increases. We also examine the impact of presenting hints when recalling textual passwords, which is assumed to be effective according to the *encoding specificity theory* [26], [27].

Contributions. Our main contributions are as follows:

1) We formulate the notion of a password hint, and discuss the optimal features of an effective password hint system under various conflicting constraints.
2) We propose SỲNTHIMA, a novel and sound cued recall-based textual password method that reveals no information regarding the password, requires no modifications to authentication servers, and requires no additional setup or registration steps.
3) We develop an implementation of SỲNTHIMA, perform an in-depth security analysis of the system, and conduct an in-lab user study with 30 participants and a total of 120 login trials to evaluate the system's effectiveness and usability, whilst simultaneously providing insights into user behavior, perception, and eagerness regarding our system and password hint systems in general.

[1]SỲNTHIMA is a Greek word (σύνθημα) meaning 'cue'.

Our in-lab user study demonstrated that there was a 30% decrease in the number of failed login trials when contact names were used as password hints. The results also suggest that there was a decrease of at least 27% in the number of incorrect login attempts when the password hints mechanism was utilized. Given that the conducted statistical tests did not signal significant differences in the time required to log in with and without using password hints, and that our participants' overall subjective evaluation was positive, we do not expect the application of cued-recall in textual passwords to have any negative implications on usability.

II. Background and Related Work

Prior work has focused on addressing the limitations of textual passwords, including memory load issues relating to the management of many passwords for different user accounts [2], [28], [7]. Some researchers have measured the recall rates of authentication secrets for a number of user authentication methods and studied the relationship between password strength and password recall errors [29], [2], [30]. For example, Mazurek et al. discovered that there is a positive correlation between the number of recall errors and password strength [2].

Given the cognitive overload that results from the number of unique passwords that a user must remember, variations in the frequency of use of each one, and the possibility of confusing different passwords with each other, it has been suggested that the password overload problem be examined by studying the nature of human memory [31], [32]. Based on the types of memory retrieval utilized in knowledge-based user authentication techniques, prior work has classified these techniques into recognition, recall, and cued recall-based passwords [27]. For graphical passwords, in terms of designing recognition-based authentication schemes, researchers have taken advantage of the fact that human memory more effectively retrieves images than textual descriptions [33], [20], [34].

Cued recall-based graphical passwords have also received considerable attention in the authentication literature (e.g., [35], [20]). However, few research studies have leveraged recognition and cued recall in textual password schemes, despite their advantages in terms of reducing login times and aiding the password retrieval process [36], [37]. For instance, Lu et al. proposed a scheme that presents some characters of a user's password as a cue to aid password recall [18], [38]. However, the question of how to choose effective password cues that maximize users' abilities to recall their passwords and do not increase the vulnerability of passwords against guessing remains open. In contrast to SỲNTHIMA, most existing password hint methods either ask users to formulate their hints or store them for later retrieval [39], [40], [41]. The security implications of these design considerations obviously include increasing the vulnerability of the corresponding methods to human errors, social engineering attacks, and compromised password hint files or databases.

Given human memory limitations and the fact that there is a general lack of techniques for helping users to circumvent some incorrect password guesses before submitting login credentials [18], we propose a mechanism for aiding users in recalling textual passwords without requiring them to change their behavior or use the maximum number of allowed login attempts. Our proposal exploits the fact that people are better at remembering things that they are familiar with [29], by employing password hints that associate users' passwords with names of people that they know.

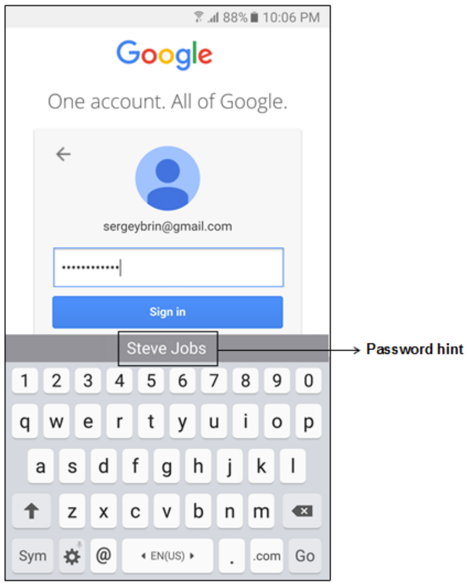


Fig. 1: A password hint (Steve Jobs in this example) generated by SỲNTHIMA.

III. SỲNTHIMA Overview

For the purpose of this study, an Android application was developed with the password hint mechanism implemented as a plug-in to an open source web browser. As shown in Fig. 1, the underlying mechanism functions by mapping each entered password to a name of a person that already exists in the user's contact list. Although this implementation of SỲNTHIMA utilizes contact names as password hints, we note that the mechanism employed by SỲNTHIMA can also be adjusted to allow the user to choose other items of contact-related information as password hints (e.g., email addresses, images, or phone numbers). At the time of installation, the mobile application requests the user's permission to read her contact list. As prior research has determined that most passwords are at least six characters long [6], we designed SỲNTHIMA such that it presents a contact name as a hint while typing every character after the fifth character. Starting from the sixth character of the typed password, different hints are automatically generated each time the user adds or deletes a character. It is also worth noting that SỲNTHIMA can be adjusted by users to allow them to specify the minimum length required for triggering password hints. However, we believe that displaying hints starting from the sixth character helps in achieving a good balance between the usability and security of textual passwords, as presenting hints while typing each character is expected to overload users' memories with many hints that could be unnecessary for a successful password recall. From a security perspective, a better security level is achieved as the number of hints displayed per password decreases, as each displayed hint might contribute to reducing the size of the corresponding password space in the case of shoulder surfing attack (see Section IV). With the growing number of accounts that users create (i.e., 25 accounts for the average user [4]) and the fact that most users often resort to choosing a password from a collection of maintained passwords (i.e., 6.5 passwords for the average user [4]) rather than creating a new one, we expect displaying a hint starting from the sixth character to achieve the intended goals of SỲNTHIMA as users would mostly rely on the last couple of presented hints to jog their memories.

To further illustrate how SỲNTHIMA facilitates password recall, let us assume that a user is typing a password that is eight characters long. Once the user enters the sixth character of the password, SỲNTHIMA will display one of the contact names stored in the user's contact list as a password hint (e.g., *Bill Gates*). As the user types in the seventh and eighth characters, SỲNTHIMA will display other contact names as password cues. Note that if the user is presented with a contact name that she is not used to seeing when correctly logging into her account, this provides an indication that there is a mistake in the entered password, and allows the user to correct this mistake before submitting the login credentials. One possible correction scenario is that the user will press the backspace button and correct the mistake by typing in a different character. Therefore, if SỲNTHIMA displays the same contact name that the user is used to seeing during previous successful login attempts, the user can confirm that the entered password is correct, and submit the login form without having to use any of the failed login attempts allowed by the corresponding login system.

It is also worth noting that the underlying mapping function does not store any password-contact name associations (e.g., in a file or a database) for further processing. Instead, real-time mapping is performed each time a user types a password into any login form. This approach was specifically chosen to achieve greater levels of security against attackers who could gain access to a user's computer or smartphone. Therefore, the objectives of SỲNTHIMA are (1) to aid the password retrieval process without introducing security vulnerabilities that could be exploited by adversaries (i.e., hints generated by SỲNTHIMA provide no clues that could help adversaries to infer or predict correct passwords); (2) to facilitate easier password recall and memorization by creating mental associations that help to trigger passwords in users' memories as a result of repeatedly linking passwords with hints in previous successful logins based on the assumption that stronger mental associative ties will be developed over time as the number of times passwords are encountered with corresponding hints increases, which is supported by the *associative strength theory* [25]; and (3) to provide users with a mechanism allowing them to confirm the correctness of a typed password before submitting a login form [42]. We also expect the application of SỲNTHIMA to facilitate the implementation of more restrictive limiting of login rates, which would make it difficult for automated bots to access user accounts through brute-force, while at same time providing legitimate users with hints to help them remember their passwords without using all of the allowed login attempts.

A. Scheme Description

In this section, we begin by formally defining the password hint problem, as follows.

Definition 1. *A password hint scheme $\Im : \mathcal{P} \rightarrow \mathcal{H}$ is a one-way function, where $\mathcal{P}$ and $\mathcal{H}$ represent the sets of possible passwords and hints, respectively, and there is quite small likelihood of accidental collisions resulting from finding $p_1, p_2 \in \mathcal{P}$ with $p_1 \neq p_2$ such that $\Im(p_1) = \Im(p_2)$.*

The proposed password hint scheme consists of the following components.

- **List of contacts.** A contact list is a collection of contact information relating to individuals whom a user initiates communication with. SỲNTHIMA maintains a copy of a user's contact list and the content of the list is used to automatically generate password hints. In particular, a full name from the contact list is displayed as a hint, so that over time a mental association between the password and the contact name is retained in the user's memory. In this study, we utilized contact lists stored in Android smartphones to test the effectiveness of SỲNTHIMA (see Sections V and VI). However, it should be noted that SỲNTHIMA can be applied to any mobile or desktop application that maintains contact lists (e.g., email address books). Contact lists that are automatically synchronized across various devices for a given user can also be utilized by SỲNTHIMA, allowing the same hint for a given password to appear to the user on multiple devices (see Section III-B1).
- **Password salting function.** The salting function (i.e., $SaltedPass$ in Algorithm 1) generates a salt value and appends it to the password in question. Concatenating a password with a salt value increases the length of the password and thus increases the difficulty of conducting successful dictionary attacks without affecting the convenience of the users [43]. For example, when salting a password (e.g., $Password1$) using $SaltedPass$ function, the function would return the result of concatenating the password with a salt value (i.e., $Password1.Salt$). If the user successfully logs into her account for the first time using SỲNTHIMA, the salt would randomly be generated for the entered password and stored on the user's device. In subsequent logins, SỲNTHIMA would retrieve the salt value from the record previously created for the password in the first successful login and attach it to the entered password. A better security level is achieved as the length of the salt value increases.
- **Cryptographic one-way hash function.** The hash function is used to convert salted passwords from plain text to numerical values (i.e., hash values), which can be manipulated or encoded later to generate corresponding hints. The hash function should be a one-way, cryptographically secure function (e.g., SHA-3), so that it is impossible to invert or generate the salted passwords from the hash values. For better security guarantees, the hash of the salted password could be iterated multiple times [44]. The exact number of hash iterations should be chosen so that no delays are introduced while displaying password hints.
- **Modulo operation.** This operation is required to convert the resulting hash values to smaller values that fit the contact list size. The modulo operation also helps in masking the generated hash values, so that reversing mechanisms of cryptographic hash functions (e.g., through precomputed tables) will not be possible (see Section IV for more details).

Algorithm 1 PasswordHint

```
Require: Contacts-list {Contacts list}
Require: $pwd$ {Typed password}
Require: $H$ {Cryptographic one-way hash function (e.g., SHA2 or SHA3)}
Require: $SaltedPass$ {A function that generates a salt value, appends it to the typed password and then stores it in the tuple created for the password (see Algorithm 2)}
Ensure: Show the hint for the currently typed password so that every typed character after the 5th one will trigger this algorithm
1: $d \leftarrow H(SaltedPass(pwd))$ {Digest which is a hexadecimal string}
2: $n \leftarrow Size($Contacts-list$)$ {Determine the divisor}
3: $i \leftarrow Mod(d, n)$ {Modulo operation}
4: $hint \leftarrow$ Contacts-list$(i)$
5: Display $hint$
```

A simple form of our proposed password hint scheme is presented in Algorithm 1. First, the salted password is hashed. Then, the $hash\_value\ MOD\ number\_of\_entries$ in the contact list is used as an index pointing to an entry in the list, which will be displayed as the hint.

B. Maintaining Contacts

Given the dynamic nature of contact lists, an update to the list (i.e., insertions, deletions, or edits of contact entries) might result in a different hint being displayed for the same password for some accounts. However, for such a hint scheme to be useful, a consistent output (i.e., continuing to show the same hint as previously to the user) should be expected, regardless of whether the *Contacts-list* has been altered. Because displaying a deleted contact entry as a password hint would not be preferred by most users, it should also be noted that the deletion of a contact entry used as a hint for a password would result in linking the password with a different contact entry. Therefore, the algorithm must take into consideration all of the various scenarios in which the contact list can be updated.

1) Insertion of New Contact Entries: In order to deal with the insertion of a new entry into the contact list, we can consider two possible solutions. First, we could simply fix the value of n after the password hint application is installed, so that the value of n will not be affected by the insertion of new entries into the *contact-list*. That is, new contact entries will not be used as hints for any password. However, given that the higher the value of n the lower the chance of showing the same hint for an incorrect password (see Lemma 1), this solution would prevent the hint scheme from leveraging expected increases in the value of n, as most users' *contact-lists* are expected to expand over time.

LEMMA 1. *Given a password hint scheme $\Im : \mathcal{P} \rightarrow \mathcal{H}$ and $p_1, p_2 \in \mathcal{P}$ with $p_1 \neq p_2$, the probability that $\Im(p_1) = \Im(p_2)$ decreases when $|\mathcal{H}|$ increases.*

Proof. Assuming the values of $\Im$ are distributed evenly across the available range, so that the probability for $\Im(p_1)$ and $\Im(p_2)$ to refer to different values in $\mathcal{H}$ is the probability of randomly selecting two values out of $\mathcal{H}$

$$Pr[(\Im(p_1) \neq \Im(p_2)] = \frac{|\mathcal{H}| - 1}{|\mathcal{H}|}$$

And thus,

$$\begin{aligned} Pr[(\Im(p_1) = \Im(p_2)] &= 1 - Pr[(\Im(p_1) \neq \Im(p_2)] \\ &= 1 - \frac{|\mathcal{H}| - 1}{|\mathcal{H}|} \\ &= \frac{1}{|\mathcal{H}|} \end{aligned}$$

It is therefore clear that the probability of collision decreases when $|\mathcal{H}|$ increases. □

An alternative solution is to associate every password with its corresponding hint that was generated originally. That is, a tuple $t_i\{ServiceID, UserName, Salt, n\}$ is created for every password (where $ServiceID$ represents the identification number for the service in which the password was created, $UserName$ represents the user name associated with the password, $Salt$ represents the salt value, and n represents the size of the *Contacts-list* when the hint was generated for the first time for a particular password). By maintaining a record of the size of the user's *Contacts-list* each time a new password is linked with a hint (see Algorithm 2), a consistent presentation of contact names over time is guaranteed. Therefore, even if new contact entries are added to a user's *Contacts-list*, employing the value of n retrieved from the tuple associated with each service in the modulo operation ensures an index pointing to the same contact name as encountered at the user's first login to a particular service is obtained. This approach also allows newly created passwords to use recently added contact entries. As new entries are inserted at the end of the user's *Contacts-list*, all of contact entries that exist in the *Contacts-list* when a hint is displayed for the first time are considered when computing the value of n associated with a new password (see Algorithms 2 and 3). Note that Algorithm 2 (together with Algorithm 3 for updating the contact list) does not allow the use of newly inserted contact entries for generating hints for previously used passwords. The list of tuples created for a user is maintained in the client-side and can be cleared by the user at any time. For example, attempting to clear browsing history data would trigger a message asking the user whether to keep the data used by `SỲNTHIMA` or delete it. However, to display same password hints across various devices for a user, the list of tuples and the user's contact list would need to be maintained on a server as well. In this case, there is a trade-off between the usability and security of `SỲNTHIMA`, as having this information exported to a server increases the attack surface to include attacks on the server as well (see Section IV for more details on the security consequences of compromising `SỲNTHIMA` server).

2) Deletion of Existing Contact Entries: When an existing contact entry $contact\text{-}list[i]$ is deleted, the succeeding entries ($contact\text{-}list[j]$, where $j > i$) need to be shifted back. Because moving these entries changes their indices, the corresponding hints (i.e., those associated with passwords that are hashed to those entries) will change as well, causing different hints to be displayed to the user.

Given that the algorithm keeps its own local copy of the contact list, one way to solve this issue is to avoid shifting the contact entries, and instead to only tag the entry as deleted and keep its index unused. In this manner, the remaining entries will not be affected, and thus their corresponding password hints will also not be affected. Upon the deletion of a contact entry, users will be presented with a message notifying them that the deleted name may be associated with a password as a hint, and that the new hint will be different from the previous one (i.e., the next undeleted entry in the *contact-list*). Despite the fact that displaying a new hint may initially confuse the user, it maintains the consistency between displayed password hints and the list of contacts maintained by the user, and thus increases the user's ability to associate passwords with names of people from her contact list in the long term.

The indices of deleted entries are also kept unused even for contacts inserted in the future (see step 11 in Algorithm 2), because reusing these indices would require labeling indices of deleted entries as used or unused which is an insecure design option (i.e., `SỲNTHIMA` does not keep track of used and unused entries). That is, in order to decide if inserting a new entry at a location that occupied an already deleted entry could cause any changes to hints by overwriting existing contact names, the password hint application must keep track of used (i.e., a contact associated with a password) and unused entries by `SỲNTHIMA`. In this manner, any sequence of indices of deleted entries that have already been occupied by new contact names followed immediately by an unused index of a deleted entry can be considered safe to overwrite. However, we chose not to label entries as being associated with passwords or not (i.e., used or unused entries), because this information could be exploited by attackers who gained access to a user's machine and contact list in order to reduce the space of possible passwords by a ratio of $\frac{t}{n}$, where n is the size of the *Contacts-list* and t is the number of entries that are tagged as being unused by the password hint scheme. In other words, labeling contact entries that are being used as password hints could help in reducing the password space to $\frac{n-t}{n}$ of its original size.

3) Alteration of Existing Contact Entries: In the case of an updated entry, the content of the entry is directly updated, and the user is alerted of this. In the case that there is a hint associated with that entry, the user should be aware that the

Algorithm 2 PasswordHint2

```
Require: Contacts-list {Contacts list}
Require: $ServiceID$ {ID of service provider (e.g., a URL)}
Require: $UserName$ {username for logging}
Require: $EnrolledList$ {a list of tuples <$ServiceID$, $UserName$, $Salt$, $n$>}
Require: $pwd$ {Typed password}
Require: $H$ {Cryptographic one-way hash function (e.g., SHA2 or SHA3)}
Require: $SaltedPass$ {A function that generate a salt value, appends it to the typed password and then stores it in the tuple created for the password in the $EnrolledList$}
Ensure: Show the hint for the currently typed password so that every typed character after the 5th one will trigger this algorithm
1: $n \leftarrow Size(Contacts\text{-}list)$
2: $T \leftarrow Find(EnrolledList, ServiceID, UserName)$
3: {Find $< ServiceID, UserName, Salt, n >$ in $EnrolledList$}
4: if $T = null$ then
5:     {First time to enter a password for this service and username}
6:     insert $< ServiceID, UserName, Salt, n >$ into $EnrolledList$
7:     $T \leftarrow < ServiceID, UserName, Salt, n >$
8: end if
9: $d \leftarrow H(SaltedPass(pwd))$ {Digest which is a hexadecimal string}
10: $i \leftarrow Mod(d, T.n)$ {Modulo operation}
11: while Contacts-list(i) is marked as deleted do
12:     $i \leftarrow Mod(i + 1, T.n)$
13: end while
14: $hint \leftarrow Contacts\text{-}list(i)$
15: Display $hint$
```

Algorithm 3 UpdateContactList

```
Require: Contacts-list {Contacts list}
Require: $Op$ {Operation to be executed (Insert, Delete or Alter)}
Require: $Contact$ {Contact to be updated}
Ensure: Update the contact list while maintaining the password hint scheme
1: if $Op = Insert$ then
2:     {Insert new contact to the Contacts-list}
3:     $Contacts\text{-}list(n + 1) \leftarrow Contact$
4:     $n \leftarrow n + 1$
5:     {Insert the new contact at the end of Contacts-list}
6: else
7:     if $Op = Delete$ then
8:         {Delete a contact from Contacts-list}
9:         $index \leftarrow Location(Contact)$
10:        Tag $Contacts\text{-}list(index)$ as a deleted entry
11:        {The user will not be able to see the contact in the contacts application anymore}
12:        Present a notification message to the user that passwords associated with $Contacts\text{-}list(index)$ will refer to different contact entry
13:    else
14:        {Alter a contact in the Contacts-list}
15:        $index \leftarrow Location(Contact)$
16:        Present a notification message to the user that $Contacts\text{-}list(index)$ will be altered and it might be used as a hint for a password
17:        $Contacts\text{-}list(index) \leftarrow Contact$
18:    end if
19: end if
```

newly updated contact name will replace the old hint. Because SỲNTHIMA does not keep track of entries that are used, the user must be notified of any update. The notification message will be delivered to the user regardless of whether the entry is used as a hint or not.

In order to handle all of the above operations for maintaining contacts, we have constructed Algorithm 2. In this new algorithm, the $EnrolledList$ is maintained to keep track of the value of n when a hint is generated for the first time for a particular $ServiceID$ (e.g., a URL) and $UserName$. The new contacts can then be inserted without changing the previously displayed hints. Algorithm 3 handles the various operations that could be performed on the contact list.

IV. Security Threat Model

The threat model assumes that the major objective of an adversary is to utilize SỲNTHIMA to reduce the effort required to guess a target password. In order to achieve this goal, the adversary will compile the data obtained to accomplish a reduction in the search space for guessing the password. In this section, we identify and describe the threat model, and present several attack scenarios that could be initiated by adversaries to compromise SỲNTHIMA and gain useful information that could help them to guess the targeted password. These attack scenarios vary from simple ones (such as when the adversary physically observes the user) to more complicated attacks (when a root privilege is obtained by the adversary). We discuss the effects of each attack scenario on security, and how these effects can be mitigated. We assume that such attacks might not be detected by the end user, and therefore that additional prevention measures will not be taken (such as changing the compromised passwords before the adversary can use them).

A. *Password hint observation*

The adversary could manage to observe the user while typing her password either in an opportunistic or systematic way. In this section, we describe the consequences of different attack scenarios that fall under these two password observation approaches.

Shoulder surfing. In this type of attack, the adversary observes a target user entering a password, in order to directly observe the hint displayed on the screen. While the adversary might also have access to other important information, such as the user name and the service type, our focus in this analysis is on whether the hint can help in revealing the password.

The proposed system involves the use of a cryptographic one-way hash function, such as SHA-3, to generate hints from passwords. For the sake of argument, we can assume that the adversary will use precomputed hash tables (e.g., rainbow tables) to launch a dictionary attack. Here, a list of dictionary words and their associated hints are generated and stored for later use as a lookup table, in order to collect corresponding dictionary words of a specific hint (i.e., these dictionary words will be possible guesses for the salted password). In addition to the password hint, the adversary also needs to know the size of the contact list in question, as well as the content of the list. Otherwise, the adversary will not be able to perform steps 3 and 4 in Algorithm 1.

Given that a hint displayed by SỲNTHIMA reveals no information regarding the password associated with it, knowing a password hint only would not be of value for the adversary. For an adversary who has access to the size and full content of the contact list, knowing the password hint will only help in reducing the search space of the dictionary on average to $\frac{1}{n}$ of its original size. As SỲNTHIMA displays more than one hint per password (e.g., for an 8-character password, a hint would be displayed after typing the 6th character, another hint after typing the 7th character and then another one after typing the 8th character), knowing the hint associated with any character of the password might contribute to reducing the search space further. That is, if the attacker managed to observe h number of hints associated with different characters of the password in question, the search space would be reduced by $\frac{1}{n^h}$.

Although this reduction might be considered significant, the remaining search space is still quite large because of the salt. Furthermore, an attacker with the capabilities to observe the sequence of hints displayed while typing in a password using a touch screen keyboard might probably find it easier to directly observe typed characters without having to use SỲNTHIMA to facilitate such attack. In case of observing the user while typing in a password using a keyboard that is not on screen and if the adversary has already accessed the contact list of his victim, knowing the hint attached to the targeted password would only help in reducing the password space to $\frac{1}{n}$ of its size, which is still large due to the expansion in the password space resulted from salting passwords in SỲNTHIMA. What could make it even harder to explore such large search space is that most authentication servers protect against brute-force attacks by implementing account lockout policies (e.g., *the three-strikes policy* [42]), which usually lock user accounts after a predefined number of unsuccessful login attempts is exceeded [45], [46].
Note that concatenating a random salt value with each password before hashing it in SỲNTHIMA breaks the link between passwords and the hints associated with them. This increases the difficulty of running successful dictionary attacks, even if an adversary already knows a password hint and has full access to the victim's contact list. There is also a possibility that the actual salted password may not even exist in the adversary dictionary. As using the same password stems as a basis for different passwords is a commonly used strategy for simplifying later password recall, salting passwords in SỲNTHIMA also enables displaying different contact names while typing in each character of passwords used for different user accounts even if same password stems are used [47]. Thus, attempting to observe a user while entering different passwords including the same stem (e.g., starting with the same characters and ending with simple added variations) would not leak any information about the passwords in question. Furthermore, given that the chance of displaying different contact names for completely different passwords increases as the size of the contact list increases (see Algorithm 1), there is a security-usability trade-off that is entailed by the size of the contact list. That is, having sufficiently long contact list would positively affect the usability of SỲNTHIMA, as linking different passwords with different hints is expected to simplify later password recall. However, from a security perspective, linking a contact name to multiple different passwords would make it harder for an attacker who knows a hint to guess the password associated with it.

To illustrate the effect of precomputed dictionary attacks considering the existence of the modulo operation and the cryptographic hash function, let us assume that passwords are restricted to k characters, are case sensitive, and are alphanumeric. For the sake of argument and as each password is attached to a salt, let us also assume that the value of the salt is 32-bit long and that there are t printable characters that could be used to formulate passwords. This allows for t^k possible password combinations and 2^{32} possible values of the salt. Assuming that SHA-3 (SHA-256) is used as the cryptographic one-way hash function with an output of 256 bits, this allows for 2^{256} possible hash values. On the other hand, using the modulo operation converges the large number of hash values into a smaller number of hints n (i.e., contact list size), which increases the number of salted passwords associated with each given hint. Using the modulo operation to make precomputed dictionary attacks would be less effective, because each hint is associated with a large number of dictionary words ($\frac{t^k \times 2^{32}}{n}$) relative to the hash value, in which the probability of two passwords being hashed to the same value is almost zero. Note that even though the theoretical value of t is 62 (assuming that only characters and numbers are used to formulate passwords) and that the actual password space utilized by users is smaller than 62^k [7], [4], the search space is still considerably large given that an adversary would also have to consider all possible salt values attached to the password in question. Note also that attaching salts to passwords makes it infeasible for an attacker to use precomputed hash tables, as salt values were not included while computing these tables. Thus, conducting a dictionary attack also requires an adversary to compute hash tables in advance while considering all possible salt values (i.e., 2^{32} hash tables), which in turn requires considerable time and computational resources (e.g., storage requirements). If iterated hashing is used [44], the attacker needs also to know the exact number of hash iterations in order to build such tables.

Another possible attack scenario might involve an attacker attempting to guess a password by using his knowledge of the hint associated with the password, the full content of the contact list of the victim as well as the value of the salt attached to the targeted password. Given that SỲNTHIMA keeps no records of hashes of salted passwords, obtaining access to all of these data elements would only help the adversary in reducing the size of the corresponding password space to $\frac{t^k}{n}$ of its size and there would still be a large number of passwords associated with the observed hint.

Systematic password observation. This type of attack involves an adversary performing a password stealth observation. In this attack, the adversary manages to compromise the user's device by injecting a malicious code that can run in the background (e.g., by installing a background app with certain permissions) without being detected. The injected malicious code is assumed to track and record certain activities within the device, and send this data back to the adversary. For example, if the adversary was only able to capture the generated hints and link them with the corresponding usernames, then this type of attack can reduce the complexity of guessing passwords by the same extent as those described previously. However, the advantage of such an attack is that it enables the adversary to track user activities over a large time window. On the other hand, if the adversary is able to capture valuable data such as touchstrokes, then the adversary will be able to obtain copies of typed passwords directly, without requiring to compromise SỲNTHIMA. An adversary could also manage to obtain a copy of a user's contact list using a malicious application. In this case, the adversary would also need root privileges in order to access the salt value attached to the password in question and thus build the corresponding hash table. However, an adversary with the capability to obtain root access on the victim's device might find it much easier to capture the targeted password directly (e.g., by using a keylogger) without having to utilize SỲNTHIMA to guess the password.

B. Breaching of SỲNTHIMA data

In this type of attack, the adversary targets data used by SỲNTHIMA to generate password hints, whether it exists in an online contact service (e.g., Gmail or iCloud) or in the user's device itself. The possible data elements that could be breached are a user's contact list or the $EnrolledList$ maintained by SỲNTHIMA for each user. One possible attack scenario could involve an adversary seeking to utilize the contact information to guess a user's password by determining the relationship between the password and the accessed contact information, thus being able to obtain the target password from the hint shown. As described for the previous types of attacks, obtaining access to contact lists only would not be of much value for the attacker given that building the correct dictionary requires accessing the salt attached to the password in question. The adversary could also obtain access to the $EnrolledList$ of the victim which includes the salt values, the IDs of services to which a user is registered and the usernames of the user's accounts. Note that obtaining the correct salt value of a password in addition to the hint associated with it and the contact list of the victim would only help in reducing the size of the corresponding password space, and thus each observed contact name would still be associated with a very large number of possible salted passwords.

Note that in case of choosing to display same hints on all devices for a user, the user's contact list and the $EnrolledList$ maintained by SỲNTHIMA for the user would need to be made available on a server. This in turn could increase the attack surface of SỲNTHIMA to include the possibility of compromising SỲNTHIMA data stored on the server. Similarly to the case of compromising a user's device, compromising SỲNTHIMA server would help the attacker in obtaining access to a user's contact list and the $EnrolledList$ kept for the user. Obtaining access to the salt value of the target password and the full content of the user's contact list could help the adversary in reducing the size of the password space and be considered a possible breach of the user's privacy. However, it is important to note that hashes of salted passwords are not stored anywhere by SỲNTHIMA as compared to most typical authentication servers, which not only store password salts but also keep records of corresponding password hashes. Therefore, even when considering the increase in the attack surface resulted from synchronizing SỲNTHIMA data across a user's devices, this design option could still be considered a more secure one relative to typical authentication servers.

C. Root attack

In this scenario, the adversary is assumed to have root privileges that can be exploited to obtain a copy of the data stored in the compromised device (including contact information and salt values). Although obtaining access to a user's contact list and the salt value of the targeted password could help in reducing the size of the corresponding password space (as described in Section IV-A), an adversary with the capabilities to mount such root attack might probably find it easier to keylog typed passwords directly before hashing them. Therefore, this threat is already present without SỲNTHIMA. Note that in the online case, it is possible that an adversary who is able to mount a root attack could capture a password on-the-fly from SỲNTHIMA (i.e., while it is being typed by the user, and before the entered password is converted into a hash value). Given that an attacker with such capabilities might also be able to capture the user's touchstrokes directly from corresponding apps, as is also the case for typical textual password schemes, this threat is already present without SỲNTHIMA. In the offline case, the fact that the generation of password hints in SỲNTHIMA is an on-the-fly process makes it infeasible for adversaries to compromise passwords or even discover password-contact name associations, as hash values of salted passwords are not stored anywhere in the system. Thus, the advantage gained from this attack for guessing passwords can also be reduced to the same level as in the previously described cases.

V. Hypotheses

A user study was conducted to evaluate the effectiveness of SỲNTHIMA from multiple perspectives. For this purpose, we established the following hypotheses: password hints will minimize the number of incorrect login attempts **(H1)**, password hints will improve password recall in the long term **(H2)** and the effectiveness of password hints will increase as the password strength increases **(H3)**. As utilizing hints while typing passwords might introduce a slight increase in the time required to recall a password per one login attempt and since users spend a relatively considerable amount of time on incorrect login attempts, we also hypothesized that the decrease in the number of incorrect login attempts resulted from trying fewer number of passwords when hints are present will reduce the overall time to authenticate **(H4)**.

We recruited Android users to perform the tasks described in Section VI. The password hint application was installed on the participants' smartphones before the start of the user study. We implemented a Java module to collect data regarding the participants' successful and failed login attempts. For each login trial, we recorded the number of times that a participant submitted a correct or an incorrect password, the length of each entered password, the number of times a participant retyped her whole password or deleted a character from that password, and the time taken for each login attempt (i.e., between opening a login screen and clicking on the login button). After each login form was submitted, the values of all of the required metrics were calculated and uploaded to a web server. In order to evaluate the extent to which the generated password hints helped our participants to recall their passwords, the implemented module also included a feature allowing us to enable or disable SỲNTHIMA for each login trial. Thus, we could compare our participants' abilities to memorize their passwords when hints were shown and when they were not.

VI. User Study Design

To test the effectiveness of SỲNTHIMA, and the extent to which the presented hints help users to recall their passwords, we conducted an in-lab user study including three login sessions conducted in person with each participant. The participants were asked to return to the same lab environment for each login session. We asked each participant to create two new Gmail and Evernote accounts. Then, we allowed them to become familiar with the functionality of SỲNTHIMA and link their newly created passwords with hints. Next, we examined the effects of SỲNTHIMA on password recall by asking the participants to recall their passwords twice; once with SỲNTHIMA enabled and the other time with it disabled. The user study was reviewed and approved by the KACST Experimentation Ethics Committee, and consent for participation was obtained from each of the 30 participants prior to the start of the user study. The steps carried out in our user study are summarized as follows:

Pre-test questionnaire. After explaining the purpose of the experiment, as well the functionality of SỲNTHIMA, we collected data regarding the demographics of our participants, the criteria they usually follow for creating passwords, and the number of unique passwords and user accounts that they had. The pre-test questionnaire was built based on the questionnaire presented by Chiasson et al. [48].

Registration phase. Participants were asked to create two new user accounts in Gmail and Evernote. Because most real world systems ask users to choose passwords that have a minimum length of eight characters [6], [49], participants were also asked to choose a password with a minimum length of eight characters, which they had not used previously. We also asked the participants to make sure that they chose passwords that were realistic and difficult to guess. To obtain a reasonable match between the conditions of our experiment and real world login scenarios, we tried not to influence our participants' decisions when choosing passwords for their newly created accounts.

To avoid the possibility of a password remembered because the user chose the same password for both accounts, and to accurately understand the effect of employing SỲNTHIMA, the participants were instructed not to choose the same password for both newly created accounts, but this requirement was not technically enforced. The participants were also not allowed to write down the passwords that they chose for their accounts. To measure the strength of users' passwords, and associate the strength with the extent to which the hints offered helped with password recall, users were asked to re-enter their passwords in a web-based tool that determines password strength based on several metrics, including character variety and frequency of occurrence[2] [51].

As a distraction activity lasting for one minute, each participant was then asked to choose a number greater than five and calculate its factorial. The aim of this step was to clear our participants' textual working memories and direct their attention to other cognitive tasks, as suggested in prior work [52], [32]. We then asked our participants to log into their two newly created accounts in Gmail and Evernote with SỲNTHIMA enabled. This step allowed our participants to familiarize themselves with SỲNTHIMA, learn how SỲNTHIMA works, and link their passwords with the hints generated by SỲNTHIMA before testing the participants' abilities to recall their passwords in the next two phases of the user study. Including this step before examining the participants' abilities to recall their passwords was necessary in order to accurately test the effectiveness of SỲNTHIMA, by ensuring that our observations are not influenced by any learning effects associated with using SỲNTHIMA. At this stage, we reminded the participants to look at the generated password hints and link them to their passwords. Again, the participants were not allowed to write down the hints linked to their passwords by the implemented mobile application.

First login. A few days after the registration phase, which allowed the participants to link their passwords with the hints generated by SỲNTHIMA, each user was asked to log into Gmail while SỲNTHIMA was disabled, and to log into Evernote while SỲNTHIMA was enabled. The average number of days to elapse before starting the first login session was five. A maximum of five login attempts were allowed for each account. Those who failed to log into Gmail or Evernote for each of their five allowed attempts were asked to reset

[2]Note that password meters sometimes do not measure password strength accurately [50].

their passwords and log in again, in order to associate their new passwords with new hints. This allowed us to proceed to the second login phase, where we were able to test users' memories of passwords while SỲNTHIMA was enabled or disabled.

Second login. A few days after the first login phase, each user was asked to log into Gmail while SỲNTHIMA was enabled, and to log into Evernote while SỲNTHIMA was disabled. A maximum of five login attempts were allowed for each account. The reason behind the decision to switch the order of enabling and disabling hints between the first and the second login sessions was to explore the effect of password hints in two different time periods (see Section VII-A). For the Gmail accounts, the hints were enabled in the second login session to observe the effect of a longer passage of time since the registration phase. However, the enabling of SỲNTHIMA for the Evernote accounts was allowed after a shorter period in the first login session. The average number of days that had passed between the first and the second login sessions was seven.

Post-test questionnaire. Each participant was asked six five-point Likert-scale questions (see statements **A** to **F** in Table I). For questions **A**, **B**, and **D**, an answer of 1 denotes that the proposed mechanism was not at all helpful, while an answer of 5 means that the mechanism was very helpful. Answers to question **C** ranged from 1, which corresponds to *very risky*, to 5, which corresponds to *not at all risky*. For questions **E** and **F**, the answers ranged from *very difficult* to *very easy*. Participants were also asked two *yes* or *no* questions after completing their second login session. These were: (1) did you look at the generated password hints while you were trying to remember your password? and (2) are you going to use SỲNTHIMA for other accounts (e.g., online banking systems or social media sites)?

A. Recruitment

Twenty-four female and six male participants were recruited to test the formulated hypotheses. Participants were either recruited from King Saud University campus or via social media sites (i.e., Twitter and Facebook), where announcements were posted to recruit volunteers for the study. Eighty percent of the participants were aged between 18 and 35 (see Fig. 2).

TABLE I: THE 6 LIKERT SCALE STATEMENTS INCLUDED IN THE POST-SESSION QUESTIONNAIRE

A	To what extent do you think that password hints helped you confirm the correctness of your passwords?
B	How do you feel about using names of people you know as password hints?
C	To what extent do you think that accessing your address book in order to associate names of people you know with the generated hints would not threaten your information security and privacy?
D	To what extent do you think that password hints would allow faster recall of passwords?
E	How did you feel about the ease of using SỲNTHIMA?
F	How did you feel about the ease of recalling your passwords when SỲNTHIMA was enabled?

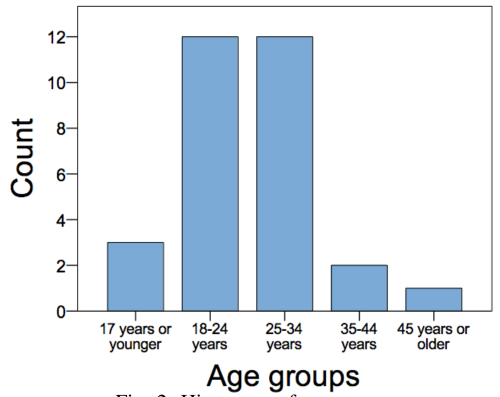


Fig. 2: Histogram of age groups

We asked each of the participants about their qualifications, and our results showed that 27% of the participants were graduates, 13% held a master's level academic degree, 33% were undergraduate students, and 27% were either high-school, or mid-school students at the time of the study.

We also gathered data regarding our participants' familiarity with technology, their perceptions about protecting their information on the web, and their overall usage of online services. When asked to rate their technical expertise levels on a scale from 1 to 5 (where 1 denotes a beginner and 5, an expert), around 10% of our participants considered themselves experts, and 60% provided a rating of 3 or 4. When asked whether they cared about the security of their information while surfing the web, 19 participants indicated that they always care, four stated that they do not, and seven said that they sometimes do. In addition, 73% stated that they reuse the same passwords for different accounts. However, they were asked not to reuse those passwords for the user accounts created during this user study. When asked about the criteria they follow when creating passwords, 21 indicated that they choose authentication secrets that are easy to remember, 10 said that they make sure that their passwords are difficult to guess, 15 said that they use similar passwords for different user accounts, and two stated that they use passwords originally suggested by software systems. We also asked about the number of unique passwords that our participants used and the number of accounts that they possessed. On average, our participants used at least four unique passwords for nine different user accounts (see Fig. 3).

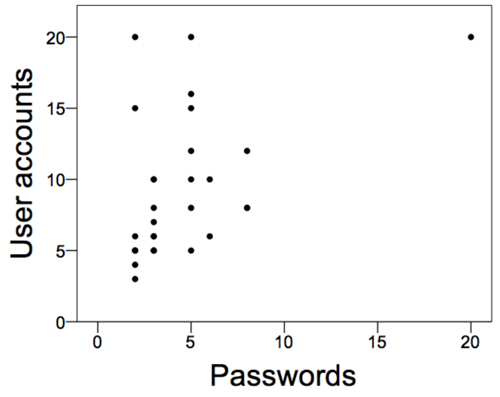


Fig. 3: Distribution of the number of passwords and user accounts used by the participants

VII. Results and Analysis

This section reports on the effects of applying cued recall to textual passwords based on the results of the conducted user study. Over the course of 28 days, we collected data relating to the login behavior of the 30 participants, who completed all of the tasks listed in Section VI. In this paper, because users often perform multiple failed attempts before correctly typing in their passwords, we use the term *login trial* to refer to the entire login process for a given user, including incorrect *login attempts* that precede the successful one. We obtained detailed data on 235 correct and incorrect login attempts for 120 unique login trials. The average password length for the 60 created accounts was 11 characters (see Fig. 4). After running two t-tests, we found no statistically significant differences between the distributions of password lengths and strengths of Evernote and Gmail passwords (p-values are 0.939 and 0.903, respectively). This allowed us to combine Evernote and Gmail data and conduct further analysis in the later phases of the user study.

A. Success Rates

To test the effect of SỲNTHIMA on the number of incorrect login attempts (**H1**), we compared the login success rates and the number of incorrect login attempts for the instances in which password hints were shown with those when hints were not shown (see Table II). The login success rate for Evernote dropped from 87% to 77% when we disabled SỲNTHIMA (see Table II). Given that we collected data regarding each successful and failed login attempt made by each participant, and allowed a maximum of five login attempts for each user account, we note differences in the number of incorrect attempts when SỲNTHIMA was enabled compared with when it was not. For the Gmail accounts, there was a 27% decrease in the number of failed attempts when we enabled SỲNTHIMA in the second login session (see Table II). For the Evernote accounts, there was a 62% increase in the number of incorrect login attempts when SỲNTHIMA was disabled in the second login session. These observations support **H1** with respect to minimizing the number of incorrect login attempts. We also observed that over all of the 120 different login trials, seven participants failed the authentication for Gmail or Evernote while SỲNTHIMA was enabled, and 10 participants failed the authentication with the mechanism disabled. Therefore, there was a 30% decrease in the number of participants who failed to complete the authentication process when we enabled SỲNTHIMA.

TABLE II: STATISTICS ON LOGIN SUCCESS RATES AND NUMBER OF INCORRECT LOGIN ATTEMPTS FOR GMAIL AND EVERNOTE

		With hints	Without hints
Success rates	Gmail	90%	90%
	Evernote	87%	77%
# Incorrect attempts	Gmail	24	33
	Evernote	29	47

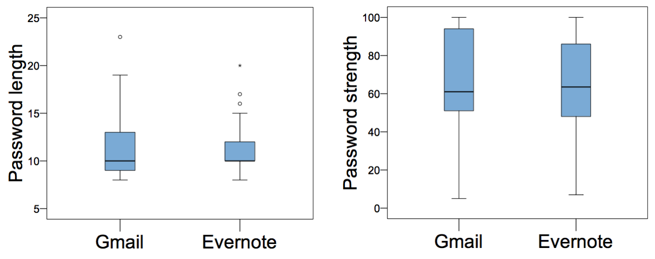
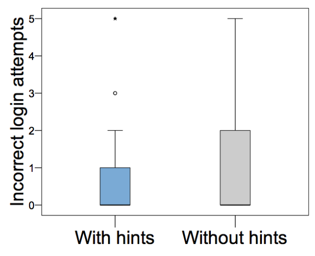


Fig. 4: Distributions of password lengths and strength for the 60 Gmail and Evernote accounts

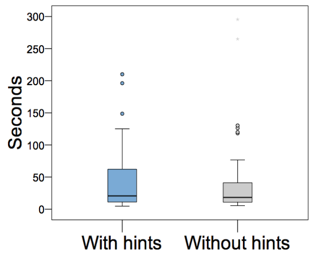
We further investigated whether the number of failed login attempts would increase in accordance with the number of days that had passed since our participants' previous logins. To explore this correlation, we used the data collected for the 120 different login trials to conduct a Pearson's correlation test, with the variables being the number of incorrect login attempts and the number of days between each pair of login trails. The results were statistically significant (p-value=0.001, correlation coefficient=0.305), indicating that there was a positive correlation between these two variables. Building on this finding, we further explored whether SỲNTHIMA was more helpful for participants who logged into their accounts after long time periods. We then found that the effect of SỲNTHIMA on reducing the number of incorrect logins was more apparent for Evernote than for Gmail owing to the amount of time that elapsed following the password creation phase, which may have resulted in more of the participants forgetting their passwords. Because SỲNTHIMA was disabled for the second login sessions for Evernote, we observed that the number of incorrect logins began to increase. We also expect that the login success rate for Gmail was higher than for Evernote because of the fact that SỲNTHIMA was enabled in the second login sessions for Gmail, which may have helped participants to recall their passwords. Therefore, this finding provides support for **H2** with respect to improving password recall in the long term.

When the hints were shown, our results also show that there was at least a 31% increase in the number of participants correctly entering their passwords on their first login attempt (see Table III and Figures 6 and 5a). For Gmail accounts, there was at least a 37% increase in the number of participants who correctly typed in their passwords at the first attempt once we had enabled SỲNTHIMA in the second login session. For Evernote, there was a 20% decrease in the number of participants who successfully logged in on their first attempt after we disabled SỲNTHIMA in the second login session. Based on the *associative-strength theory*, we also expect users' abilities to recall their passwords to improve as the frequency of encountering each password with its associated hint increases. Therefore, as entering passwords correctly on the first login attempt constitutes the most important measure of participants' ability to recall of their passwords [53], these results support **H2** with respect to improving password recall.

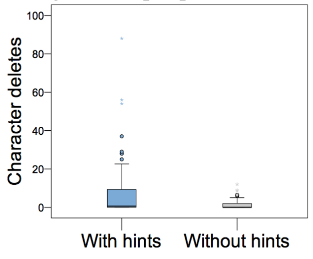
To test whether password hints are more effective for stronger passwords (**H3**), we considered the results of subtracting the number of incorrect login attempts with SỲNTHIMA enabled from the number of incorrect attempts with SỲNTHIMA disabled, as a metric to indicate the effectiveness of password hints for each of the two login trials for Gmail or Evernote. For example, when subtracting the number of incorrect login attempts for Gmail with SỲNTHIMA enabled



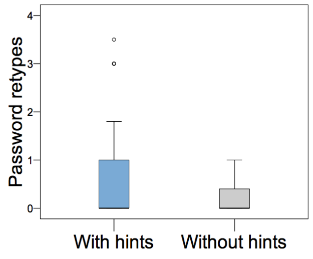
(a) Number of incorrect login attempts per trial



(b) Login times per trial



(c) Number of deleted characters per trial



(d) Number of password retypes per trial

Fig. 5: Distributions of numbers of incorrect login attempts, deleted characters, password retypes and login times for the 120 login trials

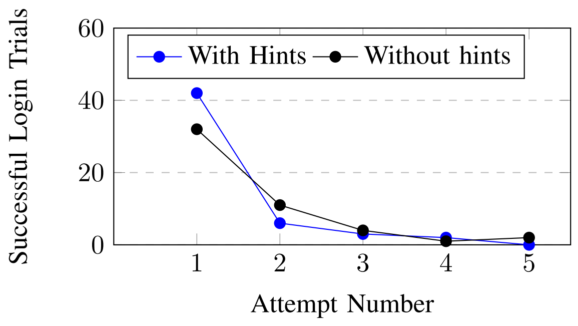


Fig. 6: Distribution of login trials according to the attempt number at which the login was successful

from the number of incorrect attempts while it was disabled, a negative result would indicate that the generated hint was not helpful for the corresponding participant, because the number of failed attempts increased when SỲNTHIMA was enabled. We employed this metric and a score indicating the strength of a given password (see Section VI) as two variables to run a Pearson's correlation test. We were surprised to find that the results were not statistically significant (p-value=0.305, correlation coefficient=0.067).

We further analyzed the collected data and discovered that most of the participants who had chosen strong passwords were less inclined to forget them (i.e., they successfully recalled their passwords with and without hints). However, this does not mean that those participants did not utilize the hints, as our observations also show that there were significant increases in the number of deleted characters and password retypes when SỲNTHIMA was enabled (see Section VII-C). Given this observation, and the fact that prior work has demonstrated a positive correlation between password strength and rates of password recall errors [2], we conclude that the data collected in our user study was not sufficiently representative of users who choose strong passwords and face difficulties in recalling them. As such, **H3** was not sufficiently tested in our study.

TABLE III: NUMBER OF PARTICIPANTS WHO SUCCESSFULLY LOGGED IN WITHIN THE FIVE ALLOWED ATTEMPTS

	With hints		Without hints	
	Gmail	Evernote	Gmail	Evernote
1st login attempt	22	20	16	16
2nd login attempt	2	4	6	5
3rd login attempt	2	1	4	0
4th login attempt	1	1	0	1
5th login attempt	0	0	1	1
Failed to login successfully	3	4	3	7

B. Time Taken for Login

For each login attempt, we measured the time taken by each participant between opening the login screen and clicking on the login form submission button. The time calculations included the time taken in making mistakes, deleting characters, adding characters, or retyping the entire password. The total time taken by each participant for each login trial was then calculated by considering the time taken for each login attempt. For instance, for a given login trial, if a participant correctly entered the password on the third login attempt, then the times taken in the first, second, and third attempts are all taken into account to calculate the login time for that trial.

For all login trials performed with SỲNTHIMA disabled, the average time taken by the participants to authenticate or use all of the five allowed attempts was approximately 39.9 seconds. On the other hand, the average time taken to log in when hints were presented was 42.4 seconds. Therefore, there was a 6% increase in the time our participants needed to authenticate when we enabled SỲNTHIMA (see Fig. 5b). However, these login times were calculated for all login attempts of each trial, starting from the opening of the login screen and ending with clicking on a login button, without including the time consumed in reloading a login page and starting to enter a new password after each failed login attempt. When observing our participants as they were entering their passwords, we estimated that an average participant took at least two seconds to begin a new login attempt after a failed one. After considering this time, we found that there was only a 3% increase in the average time needed to authenticate while SỲNTHIMA was enabled. On average, our participants took 43.5 seconds and 42.2 seconds to authenticate with and without password hints, respectively.

However, when we performed a further analysis of the average times our participants took to log into Gmail and Evernote in the first and second login sessions (see Table IV), we found that participants took longer to authenticate for Evernote in the second login session than in the first, even though SỲNTHIMA was enabled in the first session. Therefore, we expected that the increase in login times was due to the longer time elapsed since the password creation phase, which may have caused our participants to forget their passwords and spend more time trying to remember them by utilizing hints generated by SỲNTHIMA. To verify this from a statistical perspective, a Pearson's correlation test was performed with

TABLE IV: LOGIN TIMES PER LOGIN TRIAL (IN SECONDS)

	1st session		2nd session	
	Mean	Median	Mean	Median
Gmail	37.39	15.9	46.42	24.77
Evernote	38.43	17.99	42.45	21.6

the variables being the login times (in seconds) and the number of days that had passed since the last login session for each of the 120 login trials. As a result, we found that that there was a positive correlation between these two variables (p-value= 0.015, correlation coefficient=0.238). Given that there was no sufficient evidence to support **H4**, we conclude that further research is necessary to better understand the impact of the amount of time passed between consecutive login sessions on the overall time needed to authenticate.

The measures of skewness and kurtosis for login times when hints were enabled and when they were not, showed that our data was not normally distributed (see Table V). Therefore, we used a Wilcoxon non-parametric test to determine whether the observed differences in time between cases with password hints shown and those with hints not shown were statistically significant. For login trials that were not successful at their first attempts, we took into consideration the times consumed for all unsuccessful attempts as well as the successful ones. Upon setting the login times observed with SỲNTHIMA enabled and those when it was not as the variables, we obtained a p-value of 0.212. This indicates that our data reflected no significant differences between the time consumed to log in when hints were shown and that when they were not (see Table V for more descriptive statistics regarding the recorded login times).

Although utilizing hints might cause slight increases in the time taken per one login attempt, our results suggest that this time could have been spent on trying incorrect passwords if SỲNTHIMA was disabled. When utilizing hints generated by SỲNTHIMA for trying different passwords, our results suggest that the minimal delay caused by making corrections on entered passwords probably had helped the participants in narrowing down the set of possible passwords that they would have needed to consider, which eventually resulted in minimizing the number of incorrect login attempts. In traditional login scenarios, we note that the verification of login attempts by authentication servers takes a considerable amount of users' time, as users have to wait for authentication servers feedback after submitting each incorrect password guess. We also note that users might not always need to look at all hints generated by SỲNTHIMA if they do recall parts of their passwords, as the time they spend on utilizing password hints is completely under their control, and no restrictions are imposed by SỲNTHIMA.

C. Character Deletions and Password Retypes

We counted the number of times each participant clicked on the backspace button while typing in her password. Our results show that on average the participants deleted approximately two characters when SỲNTHIMA was disabled, and eight characters when SỲNTHIMA was enabled. For each login attempt, the number of times a participant deleted the whole password and started again was also counted. Our observations

TABLE V: LOGIN TIMES (IN SECONDS), CHARACTER DELETES AND PASSWORD RETYPES PER LOGIN TRIAL (AFTER EXCLUDING OUTLIERS)

		Mean	Median	SD	Skewness	Kurtosis
Login time	With hints	42.42	20.64	47.57	1.95	3.74
	Without hints	39.92	18.28	58.12	3.09	10.44
Character deletes	With hints	8.31	0.58	16.82	2.954	9.798
	Without hints	1.70	0	2.98	2.148	4.228
Password retypes	With hints	0.56	0	0.927	1.927	2.985
	Without hints	0.22	0	0.37	1.381	0.223

show that there was a significant increase in the number of password retypes when SỲNTHIMA was enabled. Our results show that participants who logged in successfully on their first attempt deleted at most two characters when SỲNTHIMA was disabled. On the other hand, the average numbers of deleted characters for those who logged in successfully to Gmail and Evernote on their first attempts with hints shown were eleven and seven, respectively. For the same cases, the numbers of password retypes were higher when hints were shown than when they were not (see Figures 5c and 5d).
To test for significant differences in the number of deleted characters and password retypes while participants attempted to log into Gmail and Evernote, we ran Wilcoxon tests to investigate whether the observed differences could be verified from a statistical viewpoint. For the number of times our participants deleted characters while they were entering their passwords, the obtained p-value was 0.003. There was also a statistical difference between the number of password retypes observed for login trials with password hints enabled and those where they were not (p-value=0.008) (see Table V for more detailed statistics). The significant differences in character deletions and password retypes between cases that included hints and those that did not therefore offer a potential explanation for the significant decrease in incorrect login attempts observed when the mechanism was enabled (see Section VII-A). This also shows that the hints presented by SỲNTHIMA had a positive effect on password retrieval, recall, and memorization [27]. We expect that this is because our participants began to think more carefully about the passwords they were entering and to utilize the generated hints, a factor which could eventually have led to decreasing their incorrect login attempts.

D. Self-reported Data

We asked our participants several questions in order to examine the perceived usability, security, and utility of SỲNTHIMA. Table VI summarizes the results of the six five-point Likert scale questions included in our post-test questionnaire (see Section VI and Table I). Furthermore, 93% of the participants indicated that they looked at the generated hints while they were trying to log in, and felt that these hints did not distract them from the task. When asked whether they would take advantage of SỲNTHIMA for other user accounts that they own, 63% of the participants said *yes*, 27% provided

TABLE VI: STATISTICS ON THE RESPONSES TO THE 6 LIKERT SCALE STATEMENTS INCLUDED IN THE POST-SESSION QUESTIONNAIRE

	A	**B**	**C**	**D**	**E**	**F**
Mean	3.36	3.8	3.7	3.33	4.23	3.73
Median	4	4	3.5	3	5	4

an answer of *maybe*, while only 10% said that they would not use SÝNTHIMA.

Overall, the means and medians presented in Table VI show that the perception of our participants regarding the extent to which SÝNTHIMA helped them to remember and confirm the correctness of their passwords was positive. It is not surprising that the answers to question **D** were neutral, considering the time required to think and utilize the generated hints before submitting the login forms. However, the results also suggest that our participants were satisfied with the usefulness and usability of SÝNTHIMA. It can therefore be expected that the time taken to utilize the generated hints per login attempt would result in a decreased number of incorrect login attempts. Overall, the results of our subjective participant evaluation suggest that the perception regarding the usability of SÝNTHIMA was positive.

The proposed mechanism presents a different contact name each time a user adds or deletes a character when typing their passwords. We observed that some users realized when they made recall errors, and got an idea of which character had been incorrectly recalled when they noticed changes in the order of the names that were presented. This observation was especially notable for the avoidance of recall errors caused by mistakes in recalling one or two characters. For instance, if a user became used to seeing names A, B, and C whenever she entered the final three characters of her password for a given website, then she would probably realize that the final character was wrong if the names A, B, and M were shown instead.

VIII. Limitations and Future Work

The conducted in-lab user study has some limitations. First, our user study tested the effectiveness of SÝNTHIMA on 30 participants only, which might not be a representative sample. The user study was also conducted in a controlled environment that might not necessarily reflect accurately the conditions of users' real-life usage of textual passwords. For example, we asked the participants to create new passwords that are not used previously. However, in real-life login scenarios, users are likely to use common password stems or reuse passwords between different accounts. Furthermore, each participant was asked to log into her Gmail and Evernote accounts under similar conditions which might not be the case when recalling passwords created for real accounts. Second, passwords recalled during the user study were encountered with their corresponding cues only once after the registration phase which might not be sufficient for developing strong associative ties between passwords and cues in users' memories. Despite these limitations, recruiting a larger testing pool for conducting a long-term experiment that involves multiple consecutive login sessions conducted over longer time periods and in non-lab environments is expected to reveal more promising results on the impact of introducing cues to textual passwords. On the basis of the *associative-strength theory*, this can be supported by the fact that encountering hints multiple times would strengthen the mental associative ties between passwords and hints, which would therefore allow users to utilize displayed hints more effectively.

Future directions for this work include conducting more thorough usability evaluation of SÝNTHIMA based on subjective and objective measures (e.g., using the framework proposed by Bonneau et al. [1]) in order to gain a deeper understanding of the usability implications of applying cued recall to textual passwords. Future research efforts could also include leveraging the picture-superiority effect, by modifying the implementation of SÝNTHIMA to generate visual password cues (i.e., by associating users' passwords with photos of people from their contact lists). To assist users whose address books include a large number of contact names, SÝNTHIMA could also be redesigned to use users' most commonly engaged or favorite contacts as hints for their passwords. We also expect the application of SÝNTHIMA to have positive effects on improving the accuracy and speed of recalling randomly generated passwords [38]. One factor that supports this argument is that SÝNTHIMA would enable the addition of semantic meaning to randomly generated passwords (i.e., by associating passwords with contact names). This is a process that has been recognized as being effective in increasing the likelihood of successfully recalling randomly generated strings [54].

IX. Concluding Remarks

This paper has presented SÝNTHIMA, a mechanism that applies cued recall to textual passwords. We have explored the effects of applying SÝNTHIMA on minimizing the number of invalid login attempts, and improving memory recall for textual passwords. After conducting a user study that compared the ability of users to recall their passwords under two conditions (with and without SÝNTHIMA enabled), our preliminary findings demonstrate that the application of SÝNTHIMA decreased the number of failed login attempts and improved the password recall rate. Because the participants were only given one chance to familiarize themselves with the hints linked to each of their accounts during our user study, we expect that users' ability to recall passwords will improve over time after they begin to utilize password hints each time they enter their login credentials. This argument is supported by the fact that all of the passwords examined in our user study were created for fake user accounts, which could be forgotten easily in comparison with those associated with high valued personal accounts. We also expect that the application of SÝNTHIMA would have important implications on minimizing users' need to resort to other verification methods to regain access to blocked accounts. The results of the subjective and objective evaluations conducted also suggest that introducing cues in textual passwords allowed users to carefully consider their passwords before submitting login forms. At the same time, no negative impact was observed on the usability of textual passwords. It is our hope that this work will spur further exploration regarding the design of effective cued recall-based textual password schemes.

Acknowledgment

The authors would like to thank the anonymous reviewers for their valuable comments which helped us improve the paper to reach its present form. This research work was supported in part by KACST.

References

[1] J. Bonneau, C. Herley, P. C. Van Oorschot, and F. Stajano, "The quest to replace passwords: A framework for comparative evaluation of web authentication schemes," in *Proceedings of the 2012 IEEE Symposium on Security and Privacy*, pp. 553–567, IEEE, 2012.

[2] M. L. Mazurek, S. Komanduri, T. Vidas, L. Bauer, N. Christin, L. F. Cranor, P. G. Kelley, R. Shay, and B. Ur, "Measuring password guessability for an entire university," in *Proceedings of the 2013 ACM SIGSAC conference on Computer and Communications Security*, pp. 173–186, ACM, 2013.

[3] C. Herley and P. Van Oorschot, "A research agenda acknowledging the persistence of passwords," *IEEE Security & Privacy*, vol. 10, no. 1, pp. 28–36, 2012.

[4] D. Florencio and C. Herley, "A large-scale study of web password habits," in *Proceedings of the 16th international conference on World Wide Web*, pp. 657–666, ACM, 2007.

[5] A. Forget, S. Chiasson, and R. Biddle, "Helping users create better passwords: is this the right approach?," in *Proceedings of the 3rd Symposium on Usable Privacy and Security*, pp. 151–152, ACM, 2007.

[6] R. Shay, S. Komanduri, P. G. Kelley, P. G. Leon, M. L. Mazurek, L. Bauer, N. Christin, and L. F. Cranor, "Encountering stronger password requirements: user attitudes and behaviors," in *Proceedings of the 6th Symposium on Usable Privacy and Security*, p. 2, ACM, 2010.

[7] M. Weir, S. Aggarwal, M. Collins, and H. Stern, "Testing metrics for password creation policies by attacking large sets of revealed passwords," in *Proceedings of the 17th ACM conference on Computer and Communications Security*, pp. 162–175, ACM, 2010.

[8] J. Bonneau, "The science of guessing: analyzing an anonymized corpus of 70 million passwords," in *2012 IEEE Symposium on Security and Privacy*, pp. 538–552, IEEE, 2012.

[9] M. Alsaleh, M. Mannan, and P. C. van Oorschot, "Revisiting defenses against large-scale online password guessing attacks," *IEEE Transactions on dependable and secure computing*, vol. 9, no. 1, pp. 128–141, 2012.

[10] M. Alsaleh and A. Alarifi, "Why phishing becomes a precursor to brutal cyber attacks? comparative evaluation framework of mitigation approaches," in *The 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, 2015.

[11] J. Fontana, "Another breach, another dollar: Is it time to kill the password?." http://zd.net/1QT593G. [Online; accessed: 10-May-2016].

[12] C. S. Alliance, "IDENTITY SOLUTIONS: Security Beyond the Perimeter." http://goo.gl/NkbMDX. [Online; accessed: 5-May-2016].

[13] B. Ur, P. G. Kelley, S. Komanduri, J. Lee, M. Maass, M. L. Mazurek, T. Passaro, R. Shay, T. Vidas, L. Bauer, *et al.*, "How does your password measure up? the effect of strength meters on password creation," in *Presented as part of the 21st USENIX Security Symposium (USENIX Security 12)*, pp. 65–80, 2012.

[14] D. Florêncio, C. Herley, and P. C. Van Oorschot, "Password portfolios and the finite-effort user: Sustainably managing large numbers of accounts," in *23rd USENIX Security Symposium (USENIX Security 14)*, pp. 575–590, 2014.

[15] G. Notoatmodjo, *Exploring the "Weakest Link": A Study of Personal Password Security*. PhD thesis, Citeseer, 2007.

[16] A. Adams and M. A. Sasse, "Users are not the enemy," *Communications of the ACM*, vol. 42, no. 12, pp. 40–46, 1999.

[17] M. Alsaleh, N. Alomar, and A. Alarifi, "Smartphone users: Understanding how security mechanisms are perceived and new persuasive methods," *PloS One*, vol. 12, no. 3, 2017.

[18] B. Lu and M. B. Twidale, "Managing multiple passwords and multiple logins: Mifa minimal-feedback hints for remote authentication," in *IFIP INTERACT 2003 Conference*, pp. 821–824, 2003.

[19] N. Alomar, M. Alsaleh, and A. Alarifi, "Social authentication applications, attacks, defense strategies and future research directions: A systematic review," *IEEE Communications Surveys & Tutorials*, 2017.

[20] R. Biddle, S. Chiasson, and P. C. Van Oorschot, "Graphical passwords: Learning from the first twelve years," *ACM Computing Surveys (CSUR)*, vol. 44, no. 4, p. 19, 2012.

[21] B. Schneier, "Two-factor authentication: too little, too late," *Commun. ACM*, vol. 48, no. 4, p. 136, 2005.

[22] C. Herley, P. C. van Oorschot, and A. S. Patrick, "Passwords: If we're so smart, why are we still using them?," in *Financial Cryptography and Data Security*, pp. 230–237, Springer, 2009.

[23] A. Alarifi, M. Alsaleh, and N. Alomar, "A model for evaluating the security and usability of e-banking platforms," *Computing*, pp. 1–17, 2017.

[24] S. L. Smith, "Authenticating users by word association," in *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 31, pp. 135–138, SAGE Publications, 1987.

[25] H. C. Ellis and R. R. Hunt, *Fundamentals of human memory and cognition*. William C. Brown, 1989.

[26] E. Tulving and D. M. Thomson, "Encoding specificity and retrieval processes in episodic memory.," *Psychological review*, vol. 80, no. 5, p. 352, 1973.

[27] E. Stobert and R. Biddle, "Memory retrieval and graphical passwords," in *Proceedings of the 9th Symposium on Usable Privacy and Security*, p. 15, ACM, 2013.

[28] K.-P. L. Vu, R. W. Proctor, A. Bhargav-Spantzel, B.-L. B. Tai, J. Cook, and E. E. Schultz, "Improving password security and memorability to protect personal and organizational information," *International Journal of Human-Computer Studies*, vol. 65, no. 8, pp. 744–757, 2007.

[29] J. Yan *et al.*, "Password memorability and security: Empirical results," *IEEE Security & privacy*, pp. 25–31, 2004.

[30] J. Bunnell, J. Podd, R. Henderson, R. Napier, and J. Kennedy-Moffat, "Cognitive, associative and conventional passwords: Recall and guessing rates," *Computers & Security*, vol. 16, no. 7, pp. 629–641, 1997.

[31] J. Zhang, X. Luo, S. Akkaladevi, and J. Ziegelmayer, "Improving multiple-password recall: an empirical study," *European Journal of Information Systems*, vol. 18, no. 2, pp. 165–176, 2009.

[32] S. Chiasson, A. Forget, E. Stobert, P. C. van Oorschot, and R. Biddle, "Multiple password interference in text passwords and click-based graphical passwords," in *Proceedings of the 16th ACM Conference on Computer and Communications Security*, pp. 500–511, ACM, 2009.

[33] D. Davis, F. Monrose, and M. K. Reiter, "On user choice in graphical password schemes.," in *Proceedings of the 13th conference on USENIX Security Symposium*, vol. 13, pp. 151–164, 2004.

[34] B. B. Zhu, J. Yan, D. Wei, and M. Yang, "Security analyses of click-based graphical passwords via image point memorability," in *Proceedings of the 21st ACM SIGSAC Conference on Computer and Communications Security*, pp. 1217–1231, ACM, 2014.

[35] S. Wiedenbeck, J. Waters, J.-C. Birget, A. Brodskiy, and N. Memon, "Passpoints: Design and longitudinal evaluation of a graphical password system," *International Journal of Human-Computer Studies*, vol. 63, no. 1, pp. 102–127, 2005.

[36] N. Wright, A. S. Patrick, and R. Biddle, "Do you see your password?: applying recognition to textual passwords," in *Proceedings of the 8th Symposium on Usable Privacy and Security*, p. 8, ACM, 2012.

[37] M. Just and D. Aspinall, "Personal choice and challenge questions: a security and usability assessment," in *Proceedings of the 5th Symposium on Usable Privacy and Security*, p. 8, ACM, 2009.

[38] J. Bonneau and S. Schechter, "Towards reliable storage of 56-bit secrets in human memory," in *23rd USENIX Security Symposium (USENIX Security 14)*, pp. 607–623, 2014.

[39] G. Selander and M. Näslund, "Apparatus and methods for obtaining a password hint," Nov. 8 2011. US Patent App. 14/356,561.

[40] M. K. Brown, H. A. Little, and M. G. Kirkup, "User-defined passwords having associated unique version data to assist user recall of the password," Sept. 22 2009. US Patent 7,594,120.

[41] M. E. Moy, "Computer security apparatus with password hints," June 13 1995. US Patent 5,425,102.

[42] S. Brostoff and M. A. Sasse, "Ten strikes and you're out: Increasing the number of login attempts can improve password usability," 2003.

[43] U. Manber, "A simple scheme to make passwords based on one-way functions much harder to crack," *Computers & Security*, vol. 15, no. 2, pp. 171–176, 1996.

[44] D. Florêncio and C. Herley, "Where do security policies come from?," in *Proceedings of the Sixth Symposium on Usable Privacy and Security*, p. 10, ACM, 2010.

[45] B. Grobauer, T. Walloschek, and E. Stocker, "Understanding cloud computing vulnerabilities," *IEEE Security & Privacy*, vol. 9, no. 2, pp. 50–57, 2011.
[46] D. E. Krutz, A. Meneely, and S. A. Malachowsky, "An insider threat activity in a software security course," in *Frontiers in Education Conference (FIE)*, pp. 1–6, IEEE, 2015.
[47] A. J. Menezes, P. C. Van Oorschot, and S. A. Vanstone, *Handbook of applied cryptography*. CRC press, 1996.
[48] S. Chiasson, P. C. van Oorschot, and R. Biddle, "A usability study and critique of two password managers.," in *Proceedings of the 15th conference on USENIX Security*, vol. 6, 2006.
[49] J. H. Huh, S. Oh, H. Kim, K. Beznosov, A. Mohan, and S. R. Rajagopalan, "Surpass: System-initiated user-replaceable passwords," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pp. 170–181, ACM, 2015.
[50] X. d. C. de Carnavalet and M. Mannan, "From very weak to very strong: Analyzing password-strength meters.," in *NDSS*, vol. 14, pp. 23–26, 2014.
[51] "Password Strength Checker: The Password Meter." http://www.passwordmeter.com/. [Online; accessed: 17-April-2016].
[52] L. Peterson and M. J. Peterson, "Short-term retention of individual verbal items.," *Journal of experimental psychology*, vol. 58, no. 3, p. 193, 1959.
[53] E. Stobert, A. Forget, S. Chiasson, P. C. van Oorschot, and R. Biddle, "Exploring usability effects of increasing security in click-based graphical passwords," in *Proceedings of the 26th Annual Computer Security Applications Conference*, pp. 79–88, ACM, 2010.
[54] M. M. King, "Rebus passwords," in *Proceedings of the Seventh Annual Computer Security Applications Conference, 1991.*, pp. 239–243, IEEE, 1991.

Noura Alomar received the Bachelor Degree of Computer and Information Sciences in Information Technology from King Saud University in 2011, and the Master of Science in Software Engineering from the School of Electronics and Computer Science, University of Southampton, Southampton, U.K., in 2014. She is currently a faculty member in the Software Engineering Department, College of Computer and Information Sciences, King Saud University, Riyadh, Saudi Arabia. Her current research focuses on software project management, user authentication, gamification, crowdsourcing technologies, human computing behavior and cross-cultural user-experience design.

Mansour Alsaleh received the undergraduate degree from King Saud University in 1999, the M.S. degree from the University of Ottawa in 2006, and the Ph.D. degree from Carleton University in 2011, all in computer science. He is currently the Deputy Director of the Joint Centers of Excellence Program (JCEP) at the King Abdulaziz City for Science and Technology (KACST) and Assistant Professor at KACST. Prior to pursuing his master studies, he spent four years in industry working in data security. His research interests lie primarily in the area of computer and network security and in the areas of big data analytics, data mining, and machine learning. His work has spanned a range of topics in network security, web security, authentication, e-privacy, identity federation frameworks and identity management. His current research interests include web security and leveraging machine learning and data mining to understand and practically address some selected problems in operational real-world enterprise environments.

Abdulrahman Alarifi received the Ph.D. in computer science from Syracuse University in 2007. He is currently an Associate Professor at King Abdulaziz City for Science and Technology in Saudi Arabia. He has more than nine years of experience in research in information security and machine learning. His research interests are in machine learning, wireless sensor network security, Internet security, intrusion detection, software security and cloud computing. He has published more than 50 research papers in these areas.