Secure Human-Computer Identification against Peeping Attacks (SecHCI): A Survey*

Shujun Li and Heung-Yeung Shum

January 2003

Abstract

It is an interesting problem how a human can prove its identity to a trustworthy (local or remote) computer with untrustworthy input devices and via an insecure channel controlled by adversaries. Any input devices and auxiliary devices are untrustworthy under the following assumptions: the adversaries can record humans' operations on the devices, and can access the devices to replay the recorded operations. Strictly, only the common brain intelligence is available for the human. In this paper, such an identification system is called SecHCI as the abbreviation – Secure Human-Computer Identification (or Interface). In the real world, SecHCI means the peeping attacks to widely-used fixed passwords: an adversary can observe your password via his own eyes or some hidden device (such as min-camera) when your input them on your keyboard or with your mouse.

Compared with human-computer identifications with the aid of trustworthy hardware devices, only a few contributions have devoted to the design and analysis of SecHCI. The most systematic works are made by N. J. Hopper & M. Blum recently: some formal definitions are given and the feasibility is shown by several SecHCI protocols with acceptable security (but usability is not very good because of their inherent limitations). In this paper, we give comprehensive investigations on SecHCI, from both theoretical and practical viewpoint, and with both system-oriented and user-centered methods. A user study is made to show problems of fixed passwords, the significance of peeping attack and some design principles of human-computer identifications. All currently known SecHCI protocols and some related works (such as visual/graphical passwords and CAPTCHAs) are surveyed in detail. In addition, we also give our opinions on future research and suggest a new prototype protocol as a possible solution to this problem.

*The corresponding author is Shujun Li, personal web site: http://www.hooklee.com. An early version has been published online in Elsevier Science's Computer Science Preprint Archive, vol. 2003, no. 1, pp. 5-57, 2003.

Contents

1 Introduction

1.1 Human-Computer Identification

The task of identification[1] is to help one party (called *prover* or *claimant*) to prove its entity to another party (called *verifier*), i.e., to help the *verifier* to distinguish the *prover* as the claimed identity from the malicious *impersonators*. Form the viewpoint of the *verifier*, the outcome of one identification procedure is one of the following two binary values about the *claimant*'s identity: *acceptance* or *rejection*. The above identification is *unilateral* and can be extended to *mutual* one: both parties should prove their identities to another party. The *mutual* feature becomes very important if the verifier can be impersonated by adversaries. Generally, a *unilateral* identification method can be easily modified to a *mutual* ones by twice bidirectional identifications.

Formally, there are several objectives of an identification system: 1) **Completeness** – An honest *claimant* A can prove its entity to the *verifier* B with overwhelming probability; 2) **Weak Security (Soundness)** – The probability is negligible that any other party C (*impersonator*) distinct from A, can cause the *verifier* B to accept A's identity; 3) **Untransferability**[2] – The verifier B cannot reuse an identification exchange with A to successfully impersonate A to a third *verifier* C; 4) **Strong Security** – The above points still remain true even if a malicious *impersonator* C observes a (polynomially) large number of previous identifications between A and B (even if C can participate the previous identification executions with either or both A and B); 5) **Timeliness** – The identification should be complete within acceptable (short enough) time interval. Generally, the first two objectives correspond to so-called *weak identification* (or called *weak authentication*), and the fourth one (sometimes together with the third one) corresponds to so-called *strong identification* (or *strong authentication*)[3].

Generally speaking, depending on which the security is based, there are three chief classes of identification methods: knowledge-based identification (what you know), token-based identification (what you have) and biometrics-based identification (who are you).

A frequently-used identification system is the login interface checking your password (knowledge-based) to determine whether or not you have the permission to use the computer, to access the files/printers on your LAN servers, and to connect your e-mail account, etc. Another kind of widely-used identification systems are automatic teller machines (ATM) deployed by commercial banks to provide convenient cash service for the customers with an legally issued banking cards (token-based), which is generally protected by 4-digit or 6-digit PINs (knowledge-based) to avoid possible theft, loss and forgery. In fact, card-based identification systems have been widely used and studied for many years [11, 12].

Also, there are a lot of other identification schemes either or both existing in cryptology literatures and security applications: one-time passwords [4, 13–15], visual cryptography based identification (a transparency as the token) [5, 6, 16], zero-knowledge based identification schemes [7–10], and plenty of biometrics based human-computer identification schemes [17–24]. Most above ones can enhance the security against dictionary attack of textual passwords, but they will reduce the usability by adding auxiliary devices and/or become sensitive to theft and loss. Please see Table 1 and 2 for a comprehensive comparison of different identification methods. More detailed description on identification can be found in [1, 25].

From 1990s, biometrics has attracted much attention as an entirely novel idea of human identification. Can biometrics replace other conventional cryptographic identification methods? We think the answer is false, because biometrics technology has some serious defects. In fact, most biometrics techniques are still in the stage of study, and not so ripe for final applications. Some primary defects about biometrics are described as follows. **1) *Some methods do not work for some individuals or situations***: This is the most crucial drawback of today's biometrics devices. Essentially speaking, there exists trade-off between FAR (False Acceptance Rate) and FRR (False Rejection Rate), and 100% FAR/FRR is absolutely impossible. Two neutral reports on the test of current biometrics products are [21, 26]. As a negative sound, ACLU (American Civil Liberties Union) has reported poor performance of face-recognition technology in practice [27]. **2) *Privacy concerns***: Almost all biometrics techniques are sensitive to privacy issues. A biometrics system for login can also be easily configured for surveillance. There are many people and organizations have shown their opposite opinions on using any biometrics technology [28] for surveillance. The extensive debates between the biometrics vendors and the liberty advocates will continue for a long time. **3) *Weak security***: The imperfect FAR/FRR make biometrics identification less secure than other human-computer identification methods. What's worse, biometrics-based login systems can be easily fooled by intelligent impersonators (even with rather simple facilities). A recently-reported example is the artificial "gummy" fingers made by Tsutomu Matsumoto et al. to frustrate fingerprint recognition systems [29]. Similarly, it is easy to fool face recognition systems by delicate face-painting operations.

[1] It is also called entity authentication and identity verification in literatures [1].

[2] Please note not all identification schemes satisfy this property. For some identification methods, it is required that the secret used by the *claimant* to prove itself identity should be known by the *verifier*.

[3] Informally, some researchers call token-based and biometrics-based identifications as *strong authentications* [2], since they are generally stronger than fixed password.

Table 1: Comparison of Identification Methods
Note: The following content is arranged based on the table in [3] with some modifications (denoted by "*").

Class	**Knowledge-Based** What you know	**Token-Based** What you have	**Biometrics-Based** Who you are
Methods	• **Weak Identification** – Password – PIN – Code plus "know phrase", e.g., Social Security Number plus mother's maiden name • **Strong Identification*** (see Table 2 for more details) – One-time password – Challenge-response protocol – Zero-knowledge based protocol	• Magnetic-striped card • Smart card • Dongle device (plugs into computer) • Proximity device (radio, infrared) • Transparency for visual cryptography* • Hand-held one-time password generator*	• **Physiological*** – Fingerprint/Handprint – Face recognition – Iris*/Retina scan – Vein – Ear Shape* – Body Odor* – DNA • **Behavioural*** – Handwriting – Voice/Speech – Keystroke Dynamics*
Advantages	• **Weak Identification** – Easy user enrollment – Already understood by users – Easily replaced • **Strong Identification*** (see also Table 2) – Immune to replay attack	• More secure than old passwords • Removal can kill a session (e.g., kiosk)	• Complex and difficult to impersonate • Minimal user effort • Always with you
Disadvantages	• **Weak Identification** – When implemented securely, hard to remember – Easy to guess Easy to steal – Easy to share • **Strong Identification*** (see Table 2)	• Must have it with you • Can be lost/stolen (thus tokens are generally protected by PINs*) • Physical distribution and replacement process	• Some methods do not work for some individuals or situations • Privacy concerns • Requires user's presence to enroll • Not secret • More expensive with extra hardware devices*

Table 2: Comparison of Strong Knowledge-Based Identification Methods

Class	**One-time password**	**Challenge-response protocol**	**Zero-knowledge based protocol**
Methods	• Shared password list • Sequentially updated password • One-time password sequence based on a one-way function [4] • One-time password generated by token • One-time password based on visual cryptography [5, 6]	• Protocol based on Symmetric-key encryption • Protocol based on (keyed) one-way functions • Protocol based on hand-held passcode generators • Protocol based on public-key decryption • Protocol based on digital signatures	• Fiat-Shamir protocol [7] • Feige-Fiat-Shamir protocol [8] • GQ protocol [9] • Shnorr protocol [10]
Advantages	• Acceptable performance to resist replay attack and observer attack • Simplest for implementation • Easily incorporated into conventional login system based on fixed password • Token based one-time password shares the merits of token • Visual cryptography based one-time password can resist observer attack with LVSVSS technique	• Better performance to resist replay attack than one-time password • Mutual identification is possible to defect faked *verifier* • Public-key based method is secure to untrustworthy *verifier*	• The best performance to resist replay attack • Secure to untrustworthy *verifier* • Have strict theoretical support • No danger from chosen-text attacks • No explicit use of encryption algorithms (to avoid policy and patent problems)
Disadvantages	• Hard to remember so that many users write down them to cause potential insecurity • Can only be used for limited times • Crashes if synchronization loses • *Faked verifier* can cheat the *claimant* of several one-time passwords for future use and damage the synchronization between the *claimant* and the real *verifier*	• Cryptographically secure PRNG is needed when random numbers are used as the nonces • Crashes if synchronization loses when sequence numbers or timestamps are used as the nonces	• Higher communications and/or computation overheads • Sometimes hard to ensure formal ZK property and/or soundness, and the practical efficiency simultaneously • Somewhat hard to explain the theory to the users

4) *More expensive with extra devices*: Almost all biometrics identification systems require the installations of new hardware devices, which will limit its applications and increase the implementation cost. Most users will not select biometrics systems unless their merits can exceed their defects.

Basically, I agree with the opinion that current biometrics devices for security applications are "more of the nature of toys than of serious security measures" [30][4]. Although there are a lot of efforts to promote the security applications of biometrics (such as BioAPI™ – a project of a standard programming interface for applications employing biometrics [31]), a long way is still ahead. But I agree that the family use insensitive to security and privacy will be the most promising direction of biometrics.

1.2 Fixed Password: Simple but Dangerous?

From computers to online WWW services, from UNIX to Windows® XP, fixed password is the most widely-used (and widely-abused with poor security considerations) identification in the digital world nowadays. It is really a wonder and a puzzle that such a simple mechanism has been widely accepted for almost 40 years from its invention in 1960s without any essential modification till 21st century [11, 32, 33]. It is a splendid but "puzzled" success. Maybe the reason is that almost people naturally hate complicated things (i.e., we are instinctively lazy) and trust the old saying "simpleness is beauty". Another reason of password's success may be the fact that many people underestimate the significance of the security issue in their cyber-life except that they have met real attacks from malicious adversaries.

The most important defect of textual passwords is the weak security against dictionary attack, since almost all users will select passwords from a very small subset of the total password space to memorize them easily [34]. To fight against such an attack, some countermeasures have been proposed, such as the compulsive policy to avoid the use of insecure passwords that can be found in one or more dictionaries, the policy to enforce users to change their passwords periodically, and even the policy to compel users to set pronunceable nonsense passwords [34–36]. But many such policies may cause other security problems, such as the one that users write down their passwords to avoid forgetting (a new way to leak the secrets) [35]. Actually, as pointed out by many researchers [3, 35, 37–42], there exists a real paradox between the security and usability (chiefly memorability) of fixed textual passwords. In Sec. 9.1 of [43], Bruce Schneier gave an interesting description of such embarrassment: *If you let a user select a password, he will select a bad one. If you force him to select a "good" password, he will write the good password on his post-it pad and post it on the margin of his screen. If you command him to change the password, he will change the password to the old one he changed in the last time.*

To enhance the memorability of textual password without the loss of security, some solutions have been proposed: passphrases [1], pass-algorithms [44], word association based passwords [36], rebus passwords [45], etc. The basic idea of those solutions is to enhance users' capability to recall the passwords by some context cues. Challenge-response mechanism is also suggested in [36, 44] to enhance the security of fixed passwords. Although these remedies can relieve this problem to some extend, they cannot be widely accepted since some inconvenience or limitations still exist.

Recently, the idea of using human vision capability to realize visual/graphical passwords has also been proposed as a novel solution to the paradox about textual passwords, which is based on the following cognitive fact: humans can recall pictures more easily and quickly than words (generally with less accuracy) [46]. Till now, most proposed graphical passwords can be categorized into three types: 1) ***selective pictures based passwords*** – Passface™ [47–51] (selecting your passfaces from decoy faces), Déjà Vu [52] (selecting your pass-images from decoy images); 2) ***point-and-click passwords*** – PassPic [53] (finding your pass-positions on a picture and clicking them), graphical "password windows" in Passlogix® v-GO™ SSO (Single Sign-On) system [54–56] (clicking your pass-objects in a picture); 3) ***drawing-based passwords*** – DAS graphical passwords [46] (drawing your pass-strokes on a grid). Theoretical and experimental analyses have shown that graphical passwords can provide better memorability and higher practical security than textual passwords. For more details about visual/graphical passwords, please see Sec. 5.1.

1.3 Peeping Attack: Another Real Danger!

Beside dictionary attacks, there still exists another real danger to jeopardize fixed passwords: peeping attack [57, 58], which is also called observer attack [46, 59, 60] or shoulder-surfing attacks [61]. In such an attack, your adversaries try to get your passwords with the help of your own negligence and "trivial" flaws of computer systems. Just like its name, its simplest form is just "peeping" of human eyes (i.e., "shoulder-surfing"). In this paper, we prefer to use the term "peeping attack" instead of "observer attack" and "shoulder-surfing attack" since we will extend the concept of "peeping": besides humans, electronic devices and hidden programs can also play the roles of peepers. In the following, we will give impressions of peeping attacks in both real world and

[4]Of course, this opinion is not suitable for other applications of biometrics technology.

theoretical world. Although most of use seem to neglect this attack, it really threaten our secrets from many directions.

1.3.1 In the Real World

Firstly, let us imagine how do you identify yourself to your computer everyday with your secret password: you type the password on the keyboard; the computer compares the input password with a pre-stored one, accepts you if the two passwords match and rejects you if the two cannot match. In many systems, the input password will be firstly hashed and then be compared with the pre-stored hashed value, which is used to avoid the possibility of administrators to get your passwords by reading the password file [1]. You may feel happy and comfortable since your computer has been "successfully" protected by your password for months or even years. You may feel pretentious since you think your password is so strange (random-like to all other people, even your wife/husband or your roommates) that nobody can guess it; but such a good situation may be false if some "eyes" monitor you.

Now let us transfer to another familiar scenario. When you sit before your computer and are ready to type your password, your friend Oliver (Onlooker) goes to stand behind your shoulders. He may prefer to observe the dance of your figures on the keyboard rather than watch your computer monitor, since he can see nothing on the screen but a lot of lousy asterisks. Under such a situation, how do you fell about your poor password and can you yet login your computer as calmly as usual? Often you will become nervous and may mistype your passwords for several times, and then give Oliver more chances to catch your password by his sharp eyes. Perhaps you will impatiently drive him away and then type your password.

If the danger of peeping only occurs in the above limited environment, it seems not so serious. Unfortunately, peeping attacks may arise in many different environments, such as in public computer centers full of free human eyes and the locations monitored by openly-deployed or hidden video cameras[5]. Can you boot out all "odious" people close to you in a public space and always find the cameras hidden somewhere in your room by your intimate friends? Another increasing danger is the malicious programs/torojan-horses/virus/worms deployed in your PC by your adversaries (please note that your adversaries may be your friends, your parents and even your husbands/wifes) to catch your secret passwords and then to catch your extreme secret. In a public computer center, such dangers are much more easy: if a network administrator wants to grab your password via which you can access your e-mail account at `hotmail.com`, what he can do is just to install a program monitoring the web browser and recording what you input in the login HTML page. Such unseen dangers make what you do on your computer entirely exposed to hidden eyes.

Maybe you can say "I never place any really important secrets in my computers, so I need not worry about such problems". But do you use banking cards and often withdraw your money via ATMs? Actually, because banking cards are generally protected by 4/6-digit PINs, which is also very sensitive to peeping attacks since it is possible for criminals to make fake cards. A lot of true stories about such attacks have been collected and analyzed by Ross J. Anderson in [59, 60] to show the insecurity problems with PINs of banking cards. Generally speaking, once an adversary successfully get the account number and the PIN of a customer's banking card, he can easily loot the customer's money with a counterfeit card. From [59, 60], we can enumerate three typical ways to collect the account numbers and the PINs: 1) Cunning criminals stand in ATM queues, observe your inputting PIN (the PINs can also be caught by hidden cameras), and pick up the discarded ATM ticket to get the account number; 2) A foxy maintenance engineer can covertly install a specially-designed device to collect a large number of account numbers and PINs; 3) The fastest growing way to get account numbers and PINs is to use bogus ATM terminals, for example, a group of criminals can set up fake vending machines to cheat honest customers. Have you been told by any bank staffs so many strange attacks to your banking cards? They are really occurring in our society.

In today's networking world, you can also withdraw your money through online virtual banks – many banks have supported online cashing services. The convenience provided by online banks makes many customers like to carry out electronic trades through Internet. However, behind the convenience to users, good chances come for your enemy: he need not to keep close watch on the real ATMs and be worry about how to make a counterfeit card; he can snugly sit on his home chair and monitor your computer with his hidden "eyes" to seize your account number and PIN and then loot your money by accessing the online cashing services with your PIN (of course, no card is needed).

The success of technically simple peeping attacks in the real world reveals a significant fact about practical cryptosystems: *most insecurity problems of cryptosystems were not cause by advanced cryptanalysis or other technical attacks, but by users' negligence, implementation errors and management failures* [59, 60]. The following virtual *Questions & Answers* can give convincing evidence.

- Question: *Have you been told differential attacks and/or linear attacks to break any block cipher?*

[5] A recent widely-reported sexual scandal about a Chinese woman ex-politician at Taiwan re-emphasizes the dangers from hidden cameras. Two related reports by BBC News & TIME Asia magazine are [62, 63].

- Answer: *Well... Seldom (never?) in newspaper, often at technical conferences and journals.*
- Question: *Then have you been told true stories about brute force attacks (the simplest attacks, most carried with distributed computing capabilities) to actual cryptosystems and denial-of-service attacks to WWW servers?*
- Answer: *More and more frequently, especially the distributed attacks from Internet.*
- Question: *What are the chief reasons of the recent successes of hackers and some worms?*
- Answer: *BUGs of softwares on UNIX/Linux/Microsoft Windows, network sniffers (like password peepers) and poor administrator passwords. Anyway, what hackers are always doing everyday are chiefly searching black holes and exchange their accident discoveries, not doing technical research.*

Finally, we should point out that, besides the simple attacks that even can be carried by kids, there are still a class of more technical peeping attacks caused by compromising emanations from computers. Following the definition of US government, ***compromising emanations*** are unintentional intelligence-bearing signals which, if intercepted and analyzed, disclose the classified information transmitted [64]. Research has showed that it is possible to recover readable information from the captured emanations at a distance [64–69]. To deal with this "signal leakage" issue, US government developed a series of standards which lay out how equipment should be suitably designed to avoid such leakage [70]. Generally speaking, standards, instrumentations, technical security countermeasures about compromising emanations are classified in the topic of TEMPEST – "Telecommunications Electronics Material Protected From Emanating Spurious Transmissions"[6]. To carry a TEMPEST peeping attack on a targeted computer, no physical access to this computer is needed, so such peeping attack is much more covert and dangerous than other ones. The only good news about such peeping attack is that governments of many countries have issued related laws to limit eavesdropping on computers and telecommunication channels with TEMPEST techniques [71, 72]. Some reports about TEMPEST risks in real world can be found in [73, 74]. Besides well-known electromagnetic TEMPEST, optical TEMPEST also has attracted some attention: in [68], M. Kuhn shown that it is possible to recover readable texts displayed on a CRT computer monitor even when there is no direct line-of-sight between the attacker and the monitor; in [67], Joe Loughry and David A. Umphress found that even LED indicators on some devices can leak information about the transmitted secret data. It is really true peepers everywhere [75]! See Fig. 1 for three usual ways via which your secrets on PC can be captured by adversaries.

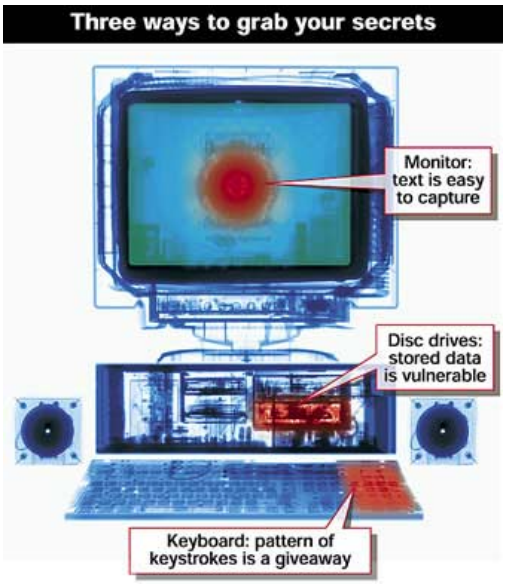


Figure 1: Three ways to grab your secret information on PC (extracted from [74])

[6]There are many different unofficial meanings of TEMPEST: "Transient Emanations Protected From Emanating Spurious Transmissions", "Transient Electromagnetic Pulse Emanation Standard", "Telecommunications Emission Security Standards", and several similar variations (including: "Tiny ElectroMagnetic Pests Emanating Secret Things"). In fact, now TEMPEST is nothing more than a fancy name for protecting against technical surveillance or eavesdropping of UNMODIFIED equipment [65].

Suggestions for Further Reading In [43], various dangers of currently-used cryptosystems are discussed and analyzed. Although the author did not specially mentioned peeping attacks, we can easily found its silhouette hidden behind the printed words (especially in the following chapters: Chapter 3 *Attacks*, Chapter 4 *Adversaries*, Chapter 9 *Identification and Authentication*, Chapter 10 *Networked-Computer Security* and Chapter 17 *The Human Factor*). We strongly suggest anyone, who is interested in security applications, reading this excellent book to obtain more significant points about non-technical attacks, such as peeping attack discussed here.

1.3.2 In the Theoretical World

Theoretically, the problem against (both *visible* and *unseen*) peeping attacks is the problem of how a human can prove its identity to a *trustworthy* computer with *untrustworthy* devices and via an *insecure* channel controlled by adversaries[7]. Here, the *untrustworthy* devices include input devices (the keyboard, the mouse, etc.) and any other auxiliary devices to help you to calculate your input (digital calculators, smart cards, etc.). All devices are *untrustworthy* because the adversaries can record how humans operate the devices and also can access the devices legally or illegally (or forge equivalent devices, recall the examples of banking cards in the last sub-subsection) to replay the recorded operations. Consequently, only the brain intelligence is available for the human to prove its identity and foil peeping attacks simultaneously.

According to the roles of onlookers, peeping attacks can be classified into two types: 1) ***passive peeping attacks (known response attacks)*** – adversaries can only passively observe the identification procedure of legal users; 2) ***active peeping attacks (chosen challenge attacks)*** – adversaries can impersonate legal verifiers to cheat the users with well-chosen challenges, and simultaneously observe the corresponding identification procedure. Active peeping attacks are meaningful when adversaries control the communication channels between the users and the legal verifiers. When Hook (Human) wants to login remote computers or online WWW servers, *active peeping attacks* may occur. Generally, two allied adversaries may be enough to carry out an *active peeping attack*; but Oliver himself can also manage to do so with the help of some auxiliary device (for example, a hidden mini-camera to monitor Hook or a network robot to cheat Hook). In this paper, the **Sec**ure **H**uman-**C**omputer **I**dentification against at least *passive peeping attacks* is called **SecHCI** in short to facilitate the description[8], we will give formal definitions of SecHCI in Sec. 3.

For a good SecHCI, besides the capability to resist *active* and *passive peeping attacks*, another feature is also desired: the human can detect the fake verifiers easily only by its own intelligence with considerable probability in one identification, which is called ***human sensitivity (or human consciousness) to active peeping attacks*** in this paper. *Human sensitivity to active peeping attacks* is useful to help users to find bad guys without exposing themselves and protect their secrets with active countermeasures.

According to humans' sensitivity (consciousness) to ongoing attacks, peeping attacks can also be divided into two classes: 1) ***open peeping attacks*** – ongoing peeping attacks that can be easily detected by humans with common carefulness, such as peeping attacks made by people standing by your side or openly-deployed cameras known to you; 2) ***hidden (unseen) peeping attacks*** – ongoing peeping attacks that that are hard to be detected by most humans, such as peeping attacks made by hidden cameras or malicious programs secretly deployed in your computers.

Because there exists trade-off between security and usability of SecHCI, it is generally difficult to design a perfect SecHCI with similar usability to fixed passwords. In many actual applications, the security against limited ($O(n)$, n is not too much) observations is enough, then it will be more easy to design practical SecHCIs and persuade users to transfer from fixed passwords to SecHCIs without too much repugnance. Furthermore, if we simplify the security to resist only open peeping attacks, the design of SecHCIs will become even more easily.

1.4 Solutions to Peeping Attack: State-of-the-Art & Problems

In almost all login systems, a simple countermeasure against peeping attacks is widely adopted: displaying "******" instead of the plain-password on the screen. It is the first wall against peeping and can provide limited security for some applications. Shielding plays the role of the second wall against peeping. When we type textual passwords on the keyboard, the hands and the quick movements of the fingers may shelter the input pass-characters from the hidden "eyes". When the users' login operations are made in a smaller space than keyboard, better shielding performance may be obtained, one example is DAS graphical passwords used on PDA-like devices [46]. Also, electromagnetic shielding is also a normal way to attenuate electromagnetic

[7] In [76], M. Blum et al. informally define this problem as follows: how a naked human inside a glass house can authenticate securely to a non-trusted terminal. "Naked" means that the human carries nothing: no smart cards, no laptops, no pencil or paper; and "Glass house" means that anybody can see what the human is doing, including everything that the human is typing.

[8] In [76], SecHCI is called by M. Blum et al. as HumanAut, HumanOIDs or PhoneOIDs, where PhoneOIDs are such protocols are carried by naked humans via phones. See [77, 78] for some details about the above terms. In this paper, we prefer to use the term SecHCI to cover more general cases.

radiation and then to resist eavesdropping from compromising emanations [79], and optical shielding can be used to resist peeping attacks based on optical TEMPEST techniques [67, 68]. But when multiple onlookers are surrounding you, it will become much more difficult to shield your inputs from all hateful "eyes". In addition, for hidden mini-cameras, shielding is entirely passive and cannot provide satisfactory security.

The similar idea to limit visible space to frustrate peeping attacks is also developed in visual cryptography, and a theoretically perfect solution is found: in [16], LVSVSS technique can be used to ensure that only the legal user itself sitting in a small visible space can see the secret image generated from two overlapped transparencies. However, theft/loss-sensitive private transparencies are still needed in LVSVSS and then violate the basic assumption of SecHCI. In fact, if trustworthy auxiliary devices are available for a human, many identification schemes will be OK to resist peeping attacks, such as most challenge-response identification protocols [1], zero-knowledge identification protocols [7–10] and secure key exchange protocols [80]. Another simple way to resist peeping attacks is to use one-time passwords [1, 4, 13–15]. But the inconvenience to memory a long list of passwords may "*provoke some initial outrage from users and security officers*" [44]. Auxiliary devices based one-time passwords may be right to reduce users' cognitive load, but again fall into the injunction of no use of any trustworthy devices. As a result, to fight against peeping attacks, we must find other practical solution to strengthen the weakest link in a cryptosystem – the human-computer interface [38]. It is the real significance of SecHCI.

With the use of challenge-response protocols and some specific algorithms (such as pass-algorithms [44], word associations [36] and Déjà Vu [52]), one-time passwords may provide partial security against peeping attacks. The basic idea is to use long passwords and to only expose partial information about the passwords in each challenge/response identification. But such an idea can only provide security against $O(n)$ observations with degraded usability. We want to seek a solution with higher security.

Compared with computer-computer identifications (or human-computer identifications with the aid of trustworthy hardware devices), only a few technical papers have devoted to the design and analysis of SecHCI [57, 58, 81–84]. T. Matsumoto and H. Imai proposed the first solution to SecHCI in EuroCrypt'91 [57], which is cryptanalyzed by C.-H. Wang et al. in EuroCrypt'95 [58]. C.-H. Wang et al. also presented an improved solution of Matsumoto-Imai protocol, but the usability is too poor to use it in practice. Soon T. Matsumoto proposed another solution in [81, 82]. But this solution is only secure against peeping attacks for a small number of identifications. In [83, 84], N. J. Hopper and M. Blum made promising advances on SecHCI: they gave some formal definitions and realized some protocols with acceptable security to show the feasibility of SecHCI. However, the usability of proposed human-computer protocols is not very good so that they are somewhat impractical as human-executable protocols, especially for Protocol 2 proposed in [84] that can provide resistance against *active peeping attacks*.

Besides the published efforts about SecHCI, there are some unpublished ones towards the solutions to this interesting but difficult problem [77, 78]. In [78], an image based SecHCI is implemented on Web to try out the feasibility. In fact, this system is a variant version of Protocol 1 in [84] with the parameter constrain $k = 2/n$ and non-essential change on the generation and display of challenges (please see Sec. 4.3.1 and 4.4 for details). In [77], many kinds of possible solutions (called PhoneOIDs by M. Blum – challenge-response protocols for use over the phone) to SecHCI are proposed and cryptanalyzed (some by M. Blum and some by his students). Generally speaking, all PhoneOIDs have bad usability since the passwords are too long; in addition, it has been found that all proposed PhoneOIDs are insecure [77, 85]. Till now, a really practical solution to SecHCI has not been found to settle the paradox between high security and acceptable usability.

In this paper, we will investigate comprehensive issues about SecHCI from both theoretical and practical viewpoints, survey all proposed SecHCI protocols and some related works (to the best knowledge of ours) and give our opinions on future research. In the next section, we firstly show a user study made by us to show the paradox between security and usability (chiefly memorability) of fixed passwords and the significance of peeping attack in users' viewpoint. In Sec. 3, we give some formal definitions about SecHCI (some ones are derived from [84]) and discuss a large number of practical attacks to SecHCI protocols. Sec. 4 contains chief content of this paper, and focuses on the comprehensive survey and analysis of currently-known SecHCI protocols. Sec. 5 overviews some related works about SecHCIs, including visual/graphical passwords (Sec. 5.1), CAPTCHAs (Sec. 5.2), etc. In Sec. 6, we give our opinions on future studies and the last section concludes this paper.

2 A User Study

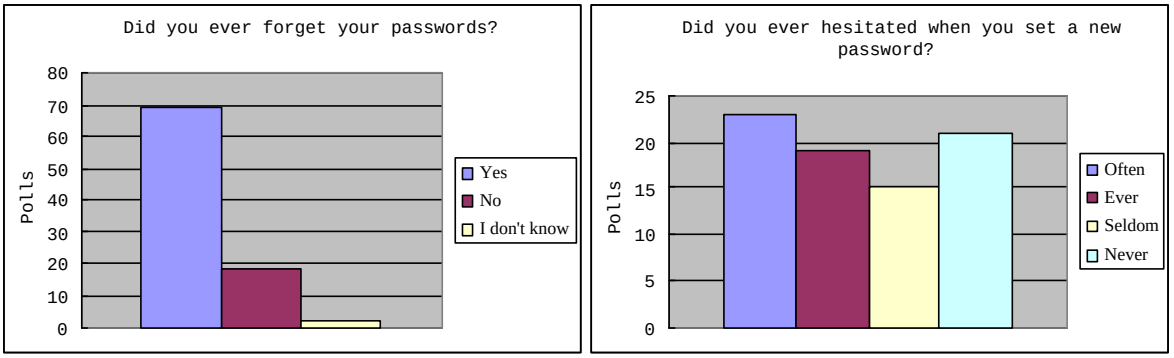
To investigate the opinions of common users on the security/usability of currently-used fixed passwords and the threat from peeping attacks, we establish a web poll site to collect such data. We hope such a study will be useful to show the significance peeping attacks in the real world, and to help the design of practical SecHCI protocols. Our tests are divided into two parts: fixed textual passwords, peeping attacks. Total 18 questions are raised, and about 100 volunteer people attended this investigation (but not all questions were answered by all volunteers). The collected data and the investigation report are shown in the follows subsections.

2.1 Fixed Textual Password

2.1.1 Security and Usability of Currently-Used Fixed Password

Question 1: Did you ever forget your passwords? This question is raised to qualitatively study the memorability of currently-used textual passwords. The result is shown in Fig. 2a. We can see that most people ever suffered from forgetting their passwords. It is the natural result of the poor memory capability of human brains and the lack of obvious meaning of some textual passwords (recall Sec. 1.2).

Question 2: Did you ever hesitated when you set a new password? This question is raised to reveal how people consider the security provided by fixed textual passwords. The result is shown in Fig. 2b. We can see most people ever hesitated when they set a new password and many people often suffer from such a simple problem. Generally speaking, people's hesitation on setting new passwords is threefold: 1) they have to chose passwords that are difficult enough to resist others' guessing (and dictionary attack); 2) they want to chose passwords different from their previous ones to get better security; 3) the number of current passwords of theirs cannot be too large to remember for them, otherwise they may forget some passwords even after several weeks.



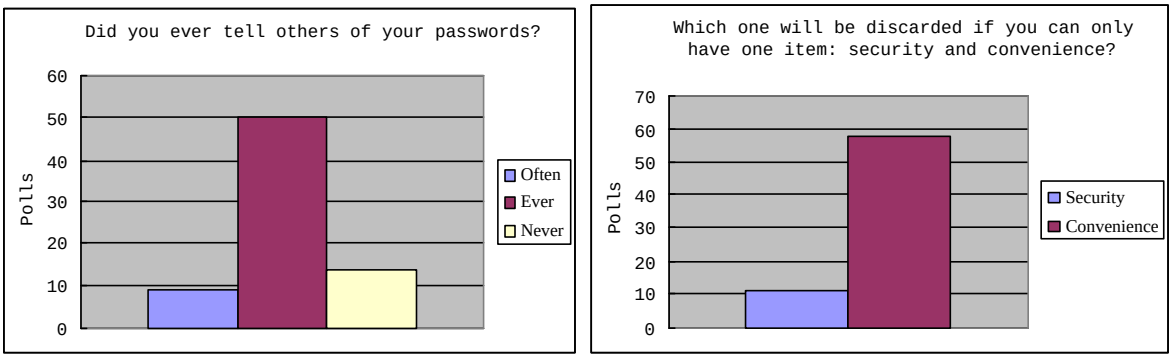
a) Did you ever forget your passwords?

b) Did you ever hesitated when you set a new password?

Figure 2: A user study: Question 1 and 2

Question 3: Did you ever tell others of your passwords? This question is raised to show people's subconsciousness about the balance of security and convenience (usability) of their passwords. The investigation result is shown in Fig. 3a. We can see most people ever "discard" security to access convenience (we are so lazy and negative to protect our own security!). The result of this question should be combined with Question 4 to reveal more comprehensive understanding on how people consider the significance of security.

Question 4: Which one will be discarded if you can only have one item: security and convenience? This question is raised to test people's active decision on the balance of security and convenience (usability) of their passwords. The investigation result is shown in Fig. 3b. In Question 3, we have found that most people would like to discard security to obtain convenience. In this test, we find most people think security is more important than convenience after careful consideration.



a) Did you ever tell others of your passwords?

b) Which one will be discarded if you can only have one item: security and convenience?

Figure 3: A user study: Question 3 and 4

Question 5: Do you have at least one passwords that are not known to others? This question is raised to show whether or not people carefully chose their passwords to avoid guessing by others. The result is given in Fig. 4. It is rather interesting that some people do not have really secret passwords. Although most people think they have such passwords, it is still possible that their passwords have been known to others (such stories have been found in our research institute).

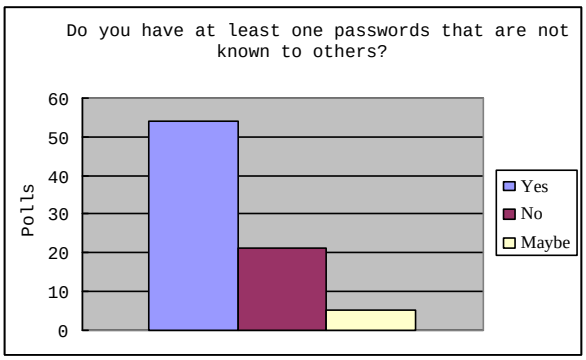


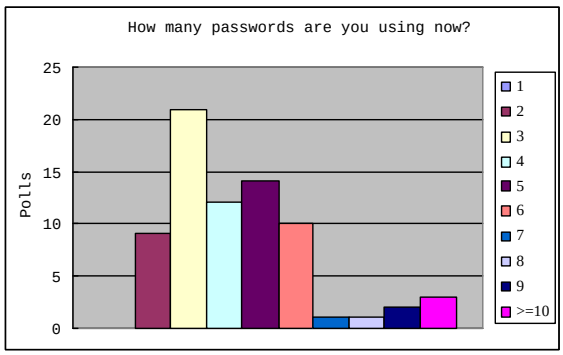
Figure 4: A user study: Question 5

Summary From the above five questions, we can find the following truth: generally, security is more important for users than usability, but our laziness and negligence often upset this principle.

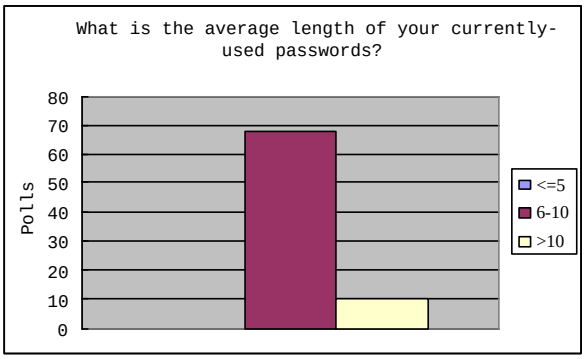
2.1.2 Some Bounds about Fixed Textual Password

Question 6: How many passwords are you using now? As we know, almost people will not use a unique password in all situations, which can be considered as the simplest way of us to protect our secrets. Question 6 is raised to test how people use this way to enhance their security. The result is given in Fig. 5a. We can see that most people will use less than 6 passwords. There are several people with more than 10 passwords, but we can expect that they will experience more suffers on forgetting some passwords.

Question 7: What is the average length of your currently-used passwords? Generally, the security of an identification system is uniquely determined by the password length. From [86, 87], we have known that 7 ± 2 is the magic number of humans' memory. This question and Question 8,9,10 are raised to confirm this cognitive result and direct the design of practical SecHCI protocols. The investigation result of this question is given in Fig. 5b. We can see that most people's passwords have the length of 6∼10.



a) How many passwords are you using now?

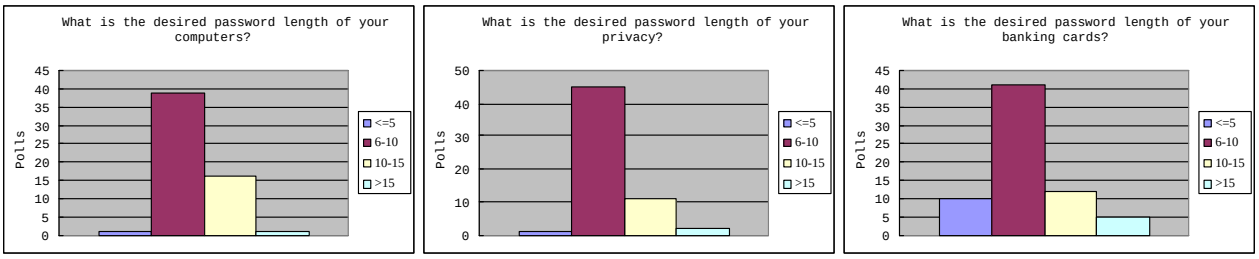


b) What is the average length of your currently-used passwords?

Figure 5: A user study: Question 6 and 7

Question 8,9,10: What is the desired password length of your computers/privacy/money? This three questions (together with Question 13,14,15) are designed to show the different security levels of different applications, and show the different attitudes of people on different security issues. The results of the three questions are given in Fig. 6. Carefully compare the three figures and consider the results of Question 13,14,15 in Sec. 2.2.1, we can see most people agree such a security rank: money $>$ privacy $\geq$ computers. Here, please note that banking cards are protected by more security mechanisms, such as the physically holdings of the cards

and the maximal number of sequent failure trials. Also, the results confirm the magic number 7 ± 2 on humans' memory capabilities.



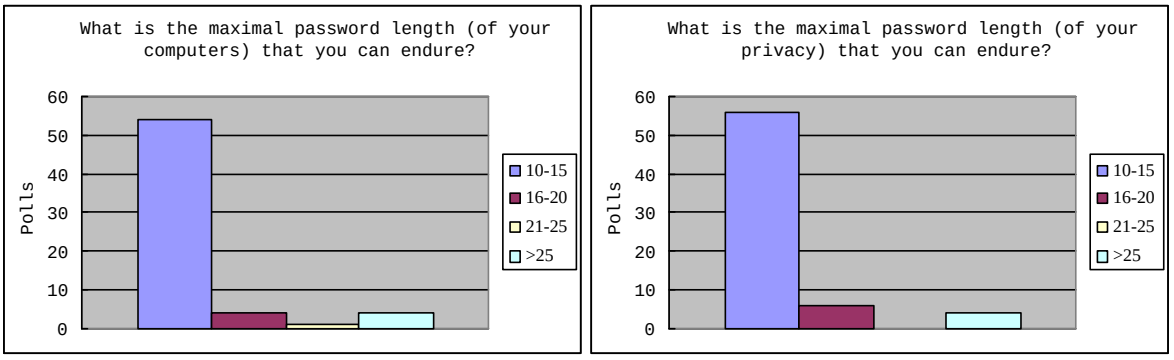
a) What is the desired password length of your computers?

b) What is the desired password length of your privacy?

c) What is the desired password length of your banking cards?

Figure 6: A user study: Question 8,9,10

Question 11 & 12: What is the maximal password length (of your computers/privacy) that you can endure? For some SecHCI protocols, to reach acceptable security, the password length must be large enough. However, we have known that the password length cannot be too large in consideration of acceptable memorability. Question 11 and 12 are raised to find the upper bound of the password length in a practical SecHCI protocol. The results are given in Fig. 7. We can see 15 is the upper bound of most people.



a) What is the maximal password length (of your computers) that you can endure?

b) What is the maximal password length (of your privacy) that you can endure?

Figure 7: A user study: Question 11 and 12

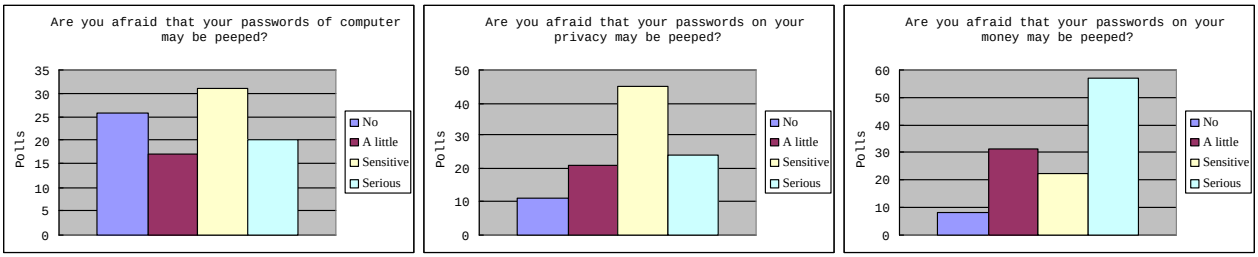
Summary From the above seven questions, we can find that the following design principle for human-computer identification systems: the password length should be about 7 ± 2, and should not be greater than 16. If the password length must be larger than 16, some specific cognitive algorithms should be introduced to help humans to choose, recall and freely change passwords at ease. Three such cognitive algorithms have been mentioned in this report: pass-algorithms [44], word-associations [36] and pass-rules (recall Sec. 4.4).

2.2 Peeping Attack

2.2.1 Do Most Users Think Peeping Attack is a Real Danger?

Question 13,14,15: Are you afraid that your passwords on your computer/privacy/money may be peeped? This three questions (together with Question 8,9,10) are designed to emphasize the different security levels of different applications, and show the significance of secure human-computer identification against peeping attacks (especially in the security applications about money). The results are given in Fig. 8. From the data we can again find the following (more obvious) security level: money $>$ privacy $>$ computer. For the security applications involving money, peeping attacks are concerned by most users. As a result, the significance of SecHCI is confirmed.

Question 16: What about your attitude on the suggestions from security experts and technical news? Because of the special role of peeping attack in security world, the warns from security experts and technical news may dramatically influence users' opinions on peeping attacks (consider the possible influence



a) Are you afraid that your passwords of computer may be peeped?

b) Are you afraid that your passwords on your privacy may be peeped?

c) Are you afraid that your passwords on your money may be peeped?

Figure 8: A user study: Question 13,14,15

of Chu Mei-feng's story [62, 63] on users' opinion on dangers from hidden cameras). Question 16 is raised to investigate how people will be influenced by security experts and technical news. The result is shown in Fig. 9. We can see most people will actively adjust their behaviors once hearing of warns from security experts and technical news.

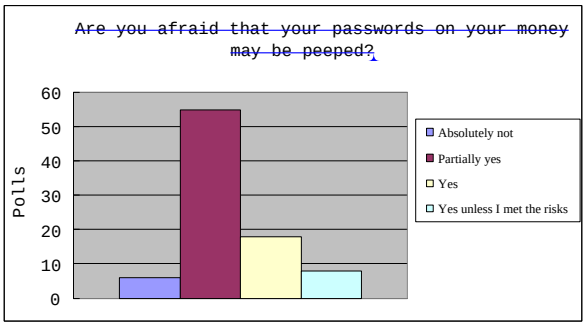
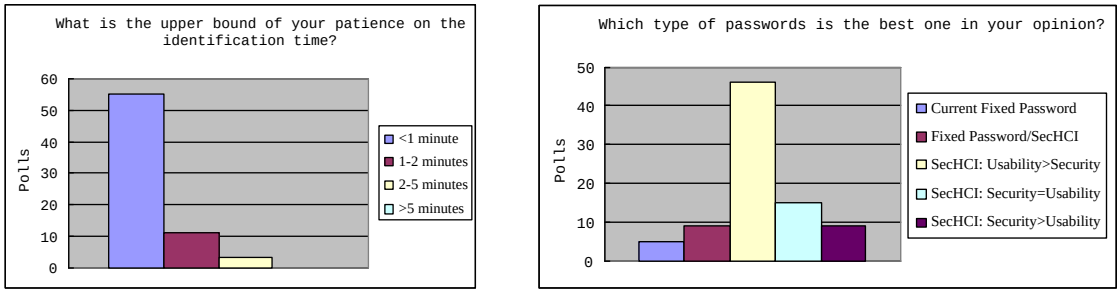


Figure 9: A user study: Question 16

Summary From the above four questions, we can confirm the significance of peeping attacks in the real world. The most sensitive applications is the ones about money, such as online e-trades and ATM uses.

2.2.2 Some Issues about the Design of SecHCI

Question 17: What is the upper bound of your patience on the identification time? In the design of SecHCI protocols, the identification procedure generally consumes much more times than fixed passwords. However, most people will become crazy if the identification time is too long. Question 17 is raised to investigate people's patience on this issue. The result is given in Fig. 10a. We can see 1 minute is the acceptable bound of most people.



a) What is the upper bound of your patience on the identification time?

b) Which type of passwords is the best one in your opinion?

Figure 10: A user study: Question 17 and 18

Question 18: Which type of passwords is the best one in your opinion? Generally speaking, it is rather difficult to fulfill both high security and good usability (please see the discussions in Sec. 4). If we relax

the requirement on security or usability on SecHCI, the design may be easier. Let us consider the following five human-computer identification systems: 1) currently-used fixed textual passwords; 2) SecHCIs with the quick entrance of fixed textual passwords; 3) pure SecHCIs with better usability but less security; 4) pure SecHCIs with balanced usability and security; 5) pure SecHCIs with better security but less usability. We believe all above five systems have their own applications in practice, an application of type-2 identification systems is as follows:

> *Your company is in a grand exhibition on consuming electronics and computer technology (may be COMDEX). Many audiences are around your computers listening to your interesting talk about the latest products of your company. Suddenly you find it is better if you can also demonstrate something on your private notebook computer, which has been on your desk and power off. Then you startup the notebook and try to input your password, at that time what do you command all your potential customers? Would you like to say "ladies and gentlemen, please turn away and go out 1 meter, and come back when I say OK"? I promise you prefer to give up your better idea about demonstration rather than offend your audiences and push yourself into a unnecessary embarrassment. Apparently, with the help of SecHCI, you may take it easy and do what you want. In this case, what is the real freedom? We can see SecHCI with quick entrance may bring both more freedom and security (against open peeping attacks).*

Question 18 is designed to investigate users' final selection on a good (both secure and convenient enough) identification system in practice. The result is given in Fig. 10b. It is shown that most people think type-3 identification systems are enough in practice, which implies that SecHCIs with $O(n)$ security are still useful.

3 General Model of SecHCI

3.1 Human-Computer Identification: Challenge-Response Protocol

To frustrate replay attack (one of peeping attacks) to fixed passwords, challenge-response protocols with time-variant parameters should be used. Here, the time-variant parameters make it (probabilistically) impossible for adversaries to replay previous observations to impersonate the legal users, and challenge-response protocol is required to support time-variant parameters. The time-variant parameters are sometimes called *nonces*, and include three chief types [1]: *random numbers*, *sequence numbers* and *time stamps*. A typical challenge-response identification protocol between two parties H (Human) and C (Computer) can be described as follows, which is a t-round 2-pass unilateral identification protocol: only H proves its identity to C.

$$
\begin{array}{lcll}
\text{H}\longrightarrow\text{C}: & \text{ID} & (S0) & \\
\text{C}\longrightarrow\text{H}: & \text{Challenge} - f_c(\text{Pass(ID)}) & (S1) & \Big\} t \text{ rounds} \\
\text{H}\longrightarrow\text{C}: & \text{Response} - f_r(f_c(\text{Pass(ID)})) & (S2) & \\
\text{C}\longrightarrow\text{H}: & \texttt{accept} \text{ or } \texttt{reject} & (S3) &
\end{array}
$$

Here, Pass(ID) denotes the shared password between H and C, $f_c(p)$ denotes a time-variant challenge associated with password p, and $f_r(c)$ denotes the response of H to a challenge c. In some challenge-response protocols, $f_c(p)$ may be just a (pseudo-)random function independent of p. Step $(S3)$ denotes the final decision of C to accept/reject the identity claimed by H.

If an onlooker observes $f_c(\text{Pass(ID)})$ and $f_r(f_c(\text{Pass(ID)}))$, he has to analyze $f_r(f_c(\text{Pass(ID)}))$ to try to find f_r or Pass(ID). A good challenge-response identification protocol should ensure such analysis difficult to provide enough security. Most protocols provide security based on currently-unsolved mathematical problems, which cannot be performed only by our brains without auxiliary devices. A SecHCI protocol should be not only a secure protocol but also a *human-only executable* challenge-response identification protocol.

In the following, we will give formal definitions of SecHCI. To describe the definitions more clearer, we follow the theoretical model used in [84] to define a human-computer identification protocol (HCIP) as a pair of (public, probabilistic) interactive protocol (H, C) with auxiliary inputs, and to denote the result of interaction between H and C with inputs x and y as $\langle H(x), C(y)\rangle$, and denote the transcript of bit exchanged during their interaction by $T(H(x), C(y))$. Here, the auxiliary inputs denote the passwords shared between H and C, and $\langle H(x), C(y)\rangle \in \{\texttt{accept}, \texttt{reject}\}$. The basic task of a HCIP is to `accept` H who shows his knowledge about an input z shared between H with C, and to `reject` others who does not know this input z. The above two sides of a HCIP is respectively called *completeness* and *soundness* in this paper[9]. If a HCIP can be performed by a human without any auxiliary devices, we call it SecHCI and such a feature is called *human-only executability*. In the next subsection,we will give the formal definitions of related concepts about HCIP and SecHCI.

[9]The two terms are initially used in zero-knowledge proof systems.

3.2 Basic Definitions about HCIP

Definition 1 (Completeness) *A HCIP is complete, if for any auxiliary input z*

$$Pr[\langle H(z), C(z)\rangle = \texttt{accept}] \geq 1 - P_c,$$

where P_c is a small negligible probability.

Completeness means that any user H can successfully prove itself identity to C with overwhelming probability, i.e., a HCIP is *complete* if it works for all honest users. Here, P_c is false rejection rate (FRR) and denotes the work reliability of a HCIP.

Definition 2 (Soundness) *A HCIP is sound, if for each input pair $x \neq y$,*

$$Pr[\langle H(x), C(y)\rangle = \texttt{accept}] \leq P_s,$$

where P_s is a small negligible probability.

Soundness means that any user H′ can only successfully cheat C with H's identity with negligible probability, i.e., a HCIP is *sound* if any dishonest user cannot hardly impersonate others. Here, P_s is false acceptance rate (FAR[10]) and denotes the security of a HCIP. Please note that this simple definition does not cover security against other specific attacks, including peeping attacks. The definitions of specific security against passive and active peeping attacks will be given in the next sub-subsection.

Definition 3 ((α, β, τ)-Human Executability) *A HCIP is (α, β, τ)-human executable, if any response $H(x)$ can be performed by a $(1 - \alpha)$ portion of the human population (who can use their brains to make responses) with the error probability β, and can be completed within τ seconds. Here, β, τ are mean values of all capable humans' behaviors.*

Good human executability means most humans can successfully prove themselves with high probability and within short time. Here, α, β, τ are mean values of all capable humans and denotes the usability of a HCIP. Please note β is essentially different from P_c: β denotes the probability that humans make unintentional wrong responses in each identification (which is determined by the computation complexity of making responses to specific challenges), but P_c denotes the probability that the verifier rejects a legal user with correct responses (which is determined by the inherent stability of the employed verifying algorithm[11]).

In this paper, human executability is also called human-only executability to emphasize the fact that no any auxiliary devices are available to execute the protocol. We would like to use HOE as the abbreviation of human-only executability. Apparently, the ideal parameters of HOE are $\alpha = \beta = 0$, $\tau = O(1)$. Practically speaking, a HCIP with *nearly perfect* HOE should satisfy $\alpha, \beta \leq 0.05$, $\tau = O(10)$. Similarly, *good* HOE corresponds to the parameters of $\alpha, \beta \leq 0.1$, $\tau = O(30)$; *feasible* HOE to $\alpha, \beta \leq 0.2$, $\tau = O(120)$ and *reluctant* HOE to $\alpha, \beta < 0.5$, $\tau = O(300)$. Please note that the exact values will be different for different applications in practice so that the above values are only used to show the rough level of HOE[12]. Almost all conventional challenge-response HCIPs are based on complicated mathematical computations to cause rather poor HOE – $(1, 1, \infty)$. In fact, the kernel task to design a HCIP against peeping attacks is to improve the HOE to an acceptable bound with carefully-designed challenge-response protocols.

3.3 Security against Attacks: Formal Definitions

Based on the above definitions, we can formally define the concepts about security of a HCIP mentioned in Sec. 1.3.2. Please note that our definitions are somewhat different from the ones given in [84].

Definition 4 ((p, k)-Security against Passive Peeping Attack (or Known Response Attack)) *A HCIP is (p, k)-secure against passive peeping attack, if for any computationally bounded adversary $\mathcal{A}$,*

$$Pr[\langle \mathcal{A}(T^k(H(z), C(z))), C(z)\rangle = \texttt{accept}] \leq p,$$

*where $T^k(H(z), C(z))$ is a **random** variant sampled from k **independent** transcripts $T(H(z), C(z))$. If p is independent of k, the HCIP is p-secure against passive peeping attack.*

[10] FAR and FRR are widely used [22] to measure the efficiency of biometric identification devices.
[11] As an example, please consider the possible failure of a face-recognition system when your face is corrected captured.
[12] In the last section, we have shown that 1 minute is the upper bound of most people's patience on the identification time.

Definition 5 ((p,k)-Security against Active Peeping Attack (or Chosen Challenge Attack)) *A HCIP is (p,k)-secure against active peeping attack, if for any computationally bounded adversary $\mathcal{A}$,*

$$Pr[\langle \mathcal{A}(T^k(H(z),C(z))),C(z)\rangle = \texttt{accept}] \leq p,$$

*where $T^k(H(z),C(z))$ is a **chosen** variant sampled from k transcripts $T(H(z),C(z))$. If p is independent of k, the HCIP is p-secure against active peeping attack.*

Only when an adversary can impersonate the legal verifier, it is possible to get a ***chosen*** variant $T^k(H(z),C(z))$. Apparently, a HCIP with (p,k)-security against active peeping attack is also (p,k)-secure against passive peeping attack. Thus, if a HCIP is (p,k)-secure against active peeping attack, we can call it (p,k)***-secure against peeping attack*** in short. In the following context, "security against peeping attack" always means "security against active peeping attack", i.e., "security against both peeping attack". Here, please note that k has special significance in the security against peeping attack, because k need not to be cryptographically large but ***practically*** large enough to prevent peeping attacks. Consider the following fact: you can only identify yourself $O(1000)$ times per year, $k = O(1000), p \leq 2^{-30}$ may be OK for a practically secure SecHCI.

To define the concept of human sensitivity (consciousness) to active peeping attacks, we follow the suggestion of [84] to add a third outcome to the interaction between H and C: $\langle H(x),C(y)\rangle = \perp$. When an adversary $\mathcal{A}$ impersonate the verifier C, if H detects the fake verifier, $\langle H(z),C(z,\mathcal{A})\rangle = \perp$, where $C(z,\mathcal{A})$ denotes the fake verifier imposed by $\mathcal{A}$ who wants to know the password z. In [84], the author use a 3-tuple value (p,q,k) to define H's capability against active adversaries, we prefer to cancel p and only use q to define *human sensitivity to active peeping attacks.*

Definition 6 ((q,k)-Human Sensitivity/Consciousness to Active Peeping Attacks) *A HCIP is (q,k)-human sensitive (conscious) to active peeping attacks, if for any computationally bounded adversary $\mathcal{A}$,*

$$Pr[\langle H(z),C(z,\mathcal{A}(T^k(H(z),C(z))))\rangle = \perp] \geq 1-q,$$

*where $T^k(H(z),C(z))$ is a **random** variant sampled from k **independent** transcripts $T(H(z),C(z))$. If q is independent of k, the HCIP is q-human sensitive to active peeping attacks.*

Apparently, (p,q,k)-detecting against active adversaries in [84] means (p,k)-security against active peeping attack plus (q,k)-human sensitivity (consciousness) to active peeping attacks.

3.4 SecHCI: A Formal Definition

Based on the definitions and concepts given in the last subsection, we can give a formal definition of SecHCI.

Definition 7 (SecHCI) *A SecHCI is a HCIP satisfying the following properties: completeness, soundness, and (α,β,τ)-human-only executability with **acceptable** parameters, and **practical** security passive (and active) peeping attacks. Please note that human sensitivity is not one of prerequisites.*

In practice, the values of the negligible probability P_c, P_s corresponding to completeness and soundness tightly depend on the demands of usability and security. Generally speaking, P_c reflects basic usability and $P_c \leq 0.01$ (the order of many biometrics identifications) may be acceptable in many applications and $P_c \leq 10^{-6}$ may be enough for most ones. P_s reflects basic security and should be much smaller than P_c in general. From strong cryptographical viewpoint, $P_s \leq 2^{-30}$ is required. More details about P_s will be discussed in the next subsection. From our user study given in the last section, we can see that a really acceptable SecHCI protocol should have the following HOE – $(0.1, 0.1, 60)$. The protocols given in [84] are on the order of $(0.9, 0.2, 300)$-human-only executable and so they are impractical.

3.5 Security against Attacks: Practical Aspects

In the last subsection, we give some theoretical definitions about security of SecHCI. In fact, to measure the actual security of a system, we must face various attacks in the real world. In this subsection, we will discuss currently-known attacks to SecHCI, among which some ones are basic attacks to break any human-computer identifications and some ones belong to the scope of peeping attacks. In addition, some "advanced" attacks are also introduced to avoid potential design flaws.

3.5.1 Basic Attacks

Random Response Attack (Soundness) When an impersonator randomly responses all challenges in once login, he should be rejected with overwhelming probability $1 - P_s$ (i.e., only can be accepted with negligible probability P_s). Apparently, capability against random response attack just means soundness. Generally, random response attack can only run online, $P_s \leq 2^{-30}$ will be acceptable for most applications. Of course, P_s need not be smaller than 2^{-K}, where K is the password entropy. For example, for banking card protected by 6 digits, $P_s \leq 2^{-20}$ is enough and $P_s \leq 2^{-16}$ may be acceptable.

Some simple countermeasures can be employed to impede the success of random response attack, such as the minimal interval between two login requests and locking the account if a maximal number of failure trial is reached (widely-used in banking card systems). But special problems should be taken into consideration when you decide to employ such countermeasures, especially the risk of Denial-of-Login Attack (see Sec. 3.5.3). In addition, some special mechanisms can be used to make automatic online attack much more difficult. For example, we can use randomly-generated pictures to realize the challenges, then Oliver's attack robot has to take some time to recognize the graphical challenges and then make random responses, which can be very hard if the involved patterns are complicated (as we know, many simple patterns for people is really very hard for computers, see also Sec. 5.2 for such a technique – CAPTCHA). If a SecHCI protocol can only be exhaustively carried by humans, the security can be relaxed to $P_s \geq 2^{-20}$.

Another problem should also be taken into consideration in the design of a SecHCI: once the user makes a wrong response to a challenge, how the system should do, reject him immediately or continue to cheat him? Since immediate rejection may tell the impersonator some useful information about his faults, we suggest cheating him until an upper security bound is reached (such a mechanism is also advised in [44]).

Brute Force Attack The simplest method to find the password is to exhaustively search it in the whole password space, which is generally called *brute force attack*. Such an attack will be available only if the adversaries observe $k \geq 1$ identification transcripts $T(H(z), C(z))$, so it belongs to peeping attack. We place it in this sub-subsection because it is a basic attack to textual passwords. Generally, the attacker cannot get the unique password if k is small, but it is possible for him to get a password subset smaller than the whole one. Then, he can search the smaller subset to guess the right password more quickly. According to [1], the attack complexity about $O(2^{60}) \sim O(2^{80})$ to find the unique password is required for offline brute force attack.

Dictionary Attack Dictionary attack is the most well-known attack to break fixed passwords. Essentially speaking, any human-computer identification system cannot avoid this problem if the secret keys can be freely selected by users (a desired feature to enhance usability). It is natural that humans can only select the passwords that are meaningful and easily to recall for them. Two ideas will be useful to enhance the security against dictionary attack: 1) visual/graphical passwords – it is much more difficult to construct a picture/image dictionary (of course, it is possible especially when the adversaries know the taste of users); 2) self-configurability – if users can define their own private configurations associate with their password, the security against dictionary attack will be improved (this idea initially proposed in [36]).

3.5.2 Peeping Attacks

Store-and-Replay Attack If an adversary $\mathcal{A}$ get k transcripts $T(H(z), C(z))$, he can store them and then replay them to increase the success probability of random response attack. Conventional password-based identification schemes cannot resist store-and-replay at all, since $\mathcal{A}$ can simply replay what H inputs in previous login to cheat C. To resist store-and-replay attack, time-variant parameters in challenge-response protocols are needed. In addition, the number of possible challenge/response pairs should be large enough to make it impossible for $\mathcal{A}$ to collect and store all challenge/response pairs. Considering the current storage technology and the time to collect the stored challenge/reponse pairs, $O(2^{40}) = O(1000\text{G})$ is OK for many practical applications, and $O(2^{50} = O(1000,000\text{G})$ is enough for almost conditions. In fact, in practical applications, the number can be much smaller, since a busy user can only login for $36,500,000 \approx 2^{25.1}$ in 100 years if he logins 100 times per day (so many!). In almost practical applications, we believe $O(2^{30})$ is enough. In ATM uses, $O(2^{20})$ is OK since generally we seldom access ATMs to withdraw money for 10 times per day.

Intelligent Off-Line Password Attack Intelligent (i.e., non-exhaustive) off-line password attack can be divided into two chief classes: known response attack (passive peeping attack) and chosen challenge attack (active peeping attack)[13]. Generally speaking, intelligent attack depends on the *ad hoc* design of a SecHCI. The following three types of attacks are frequently used in practice: 1) Differential attack – Similar to the one in block cipher, if there exist probability differences in a SecHCI, $\mathcal{A}$ can analyze the probability differences using

[13]Chosen-response attack is rare, since it is difficult for $\mathcal{A}$ to lead H to make responses as $\mathcal{A}$'s desire.

enough collected challenge/response pairs to obtain some useful information about the secret password (even can directly get the password if the SecHCI is not well designed). 2) Deduction-based attack – Any SecHCI must depend on some mathematical problem, $\mathcal{A}$ may be capable to deduce the password by solving the basis problem constructed by $T^k(H(z), C(z))$. One example is the Gaussian elimination attack to Hopper-Blum Protocol 1 in [83, 84]. Similar to offline brute force password attack, the attack complexity of $2^{60} \sim 2^{80}$ is required. Similar to store-and-replay attack, the number of known/chosen responses is $O(2^{30})$ for most practical applications. 3) Intersecting attack – This type of attack is similar to but rather different from differential attack. If some information about the password always (or always not) occurs in each challenge and/or response, it may be possible for an attacker to derive partial password by intersecting multiple challenge and/or response. This attack is based on such a fact that the 0/1-value occurrence probability (of *intersectable* information about the password), not the existence of diffenret probability. The known and chosen Q and A attack of Matsumoto Protocol in [82] are actually both intersecting attacks. The observer attack of Déjà Vu mentioned in [52] is another example.

Multi-Onlookers Peeping Attack As we have discussed in Sec. 1.4, shielding has been proposed as a simple but practical countermeasure against peeping attack. However, when more than one onlookers are near your side, it will become much more hard to shield what you input from all unfriendly eyes. In addition, shielding cannot work against hidden eyes. To resist multi-onlookers peeping attack, the security of SecHCI protocols should be ensured by subtle protocol design, not by the instructions for users.

3.5.3 "Advanced" Attacks

Partially-Known Password Attack Under some conditions, it will be possible for Oliver to partially guess Hook's password. In the world of conventional textual passwords, such an attack occurs somewhat frequently. For example, by guessing Hook's habit or from casual speech of Hook or by one partially successful peeping, Oliver may get to know the fact that the word "hook" is really in Hook's password "hook2000", then he can adjust his attack robot to find the password faster. For SecHCI, special considerations should be made on this kind of attack. Theoretically, we call a SecHCI (p, η)-***secure against partial-known password attack***, if

$$Pr[\langle \mathcal{A}(\eta \cdot K(z)), C(z) \rangle = \texttt{accept}] \leq p$$

for an attacker $\mathcal{A}$ with $\eta < 1$ of the knowledge $K(z)$ about the secret password z. Generally speaking, $\eta = 0.5$ can be used as a typical value to measure the security. The "meet-in-the-middle" attack of Hopper-Blum Protocol 2 in [84] is a typical partially-known password attack with $\eta = 0.5$.

Malicious Administrator Attack (Association Attack) Consider your passwords used for different banking cards, e-mail accounts and other online services, do you always use a few number of passwords to avoid confusion and forgetting? According to the user study in the last section, the answer is right for a large number of people. Once an administrator gets your password of his own service, he has your passwords of some other services. Of course, hashed password files reduce this danger from administrators, but how do you know the password files stored on the remote computers are really hashed correctly without leaking to the administrators since you have no permission to access the files? Perhaps you can assert this point since the service provider is a reputable company like Microsoft, but it is still very easy for a foxy administrator to add some hidden codes in the service programs (especially the Web programs, such as .aps and .php) to record your plain-password and send it to his own e-mail box for future attacks. In practice, such secret actions can be successfully carried without suspicion and detection by both the users and the bosses for a long time. If the administrator just uses the stolen passwords to see your private e-mails provided by his own company, such an attack belongs to normal peeping attacks, where the Web programs play the roles of untrustworthy auxiliary devices. But when the administrator uses the stolen passwords to guess your passwords on other services with higher probability, we call such attacks *malicious administrator attack* or *association attack*.

Although such an attack seems so strange that you may think it is rare in practice, but Bruce Schneier told us of one real story in Sec. 9.1 of [43]. To foil malicious administrator attack, we have to tell users to use low-correlative passwords for different services, which is almost impossible for most people since humans are easy to forget everything (even very important things). Recently, Single-Sign-On (SSO) mechanism [88, 89] is developed as an alternative solution to the problems with multiple passwords. One good implementation of SSO is Microsoft® .NET Passport [90]. However, although SSO may be fully successful and reach the ideal goal that each network citizen can hold only one password to venture the cyber-space all over the world, it still be desired for users to set the passwords without leaking any privacy information to the cyber-government.

Introducing self-configurability into SecHCI may provide a new idea to resist this kind of attack: the meaningful information about the secret password selected by H is configured and stored on the local computer, and the remote server only store the meaningless numbers about the password. For example, we can design

a SecHCI as follows: H freely selects n favorite images on his PC to represent the numbers from $1 \sim n$, the password is a meaningful combination of m images out of total n ones, but the one stored in server is only the meaningless numbers. Of course, with such a self-configuration feature, once H changes the order of the images contained in his password, synchronization between local computer and remote server is needed (in fact, it just means the change of the password stored in remote server although the secret meaning behind the password is untouched).

Denial-of-Login (DoL) Attack There exists a kind of notorious attack to online services – Denial-of-Service (DoS) attack [91], which aims at obstructing target servers from providing related services to legal users by exhaustively consuming resources, such as network bandwidth. Similarly, Oliver may carry out a kind of Denial-of-Login (DoL) attack to prevent Hook accessing online services (by making uninterrupted malicious login trials). To resist DoL attack, some security countermeasures against online attacks should be avoided, such as locking users' accounts once the number of continuous login failures reaches to a pre-defined number (the one used in today's banking card systems). The security of SecHCI should be ensured by the protocol itself, not by specially-designed login policy.

4 A Comprehensive Survey on Proposed SecHCI Protocols

4.1 Matsumoto-Imai Protocol [57, 58]

The first (to the best of our knowledge) SecHCI protocol was proposed by T. Matsumoto and H. Imai at EuroCrypt'91 conference [57]. After the proposal of Matsumoto-Imai protocol in 1991, C.-H. Wang et al. cryptanalyzed it with an active peeping attack (called "replay challenge attack") at EuroCrypt'95 [58]. C.-H. Wang et al. also proposed a modified protocol to resist the replay challenge attack, but the modified protocol is impractical as a SecHCI since its HOE is rather bad (C.-H. Wang et al. also acknowledged this point in their paper).

4.1.1 Mastumoto-Imai Protocol [57]

Firstly let us introduce the original Mastumoto-Imai protocol. Because Mastumoto and Imai used so many concepts and notations that the introduction of the protocol is made too intricate, here we use less notations and some entirely different concepts (especially the ones denoted by functions, such as q_j, f_j, a_j and $\mathrm{sort}(S)$) to obtain clearer description. This protocol is based on a whole alphabet Ω whose size is ω, and a question alphabet $\Gamma \subseteq \Omega$ whose size is γ. The shared password between H and C includes three parts: a window alphabet $\Lambda \subseteq \Gamma$ whose size is λ, an answer alphabet $\Delta \subseteq \Omega$ whose size is δ, and a secret word $W = \{W(1), \cdots, W(\lambda)\}$. The identification procedure is as follows:

H⟶C:	H's ID	$(H0)$
C:	$\texttt{counter} = 0$	$(C0)$
C⟶H:	A random question block $q_j = \langle q_j(1), \cdots, q_j(\gamma)\rangle$, where $\{q_j(1), \cdots, q_j(\gamma)\} = \Gamma$	$(C1)$
H⟶C:	An answer block $a_j = \langle a_j(1), \cdots, a_j(\gamma)\rangle$, where $a_j(i) \in \Delta$	$(H1)$
C:	Calculate the window $f_j = \langle f_j(1), \cdots, f_j(\lambda)\rangle$, where $f_j(i)$ represents the i^{th} element in q_j that is also in Λ, i.e., f_j satisfies the following three requirements: 1) $\forall i = 1 \sim \lambda$, $f_j(i) \in [1, \gamma]$ and $q_j(f_j(i)) \in \Lambda$; 2) $\forall i = 1 \sim \gamma$, $q_j(i) \in \Lambda \rightarrow i \in \{f_j(1), \cdots, f_j(\lambda)\}$; 3) $\forall k, k' = 1 \sim \lambda$, $f_j(k) \leq f_j(k')$	$(C2)$
C:	If $\forall i = 1 \sim \lambda$, $a_j(f_j(i)) = W(i)$ then $\texttt{counter}++$	$(C3)$
C⟶H:	If $\texttt{counter} \geq \alpha (\alpha \leq \beta)$ then outcome `accept`, otherwise outcome `reject`	$(C4)$

β rounds

In the above protocol, for legal users, $\forall i \in \{1, \cdots, \gamma\} - \{f_j(1), \cdots, f_j(\lambda)\}$, $a_j(i)$ should be selected uniformly from Δ at random. The answer elements determined by W are called "window-padding answer elements" and other elements are called "random-padding answer elements". To demonstrate how this protocol run, a simple example is given in [57]. When $\Omega = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 0\}, \omega = 10, \Gamma = \{1, 2, 3, 4, 5, 6, 7, 8\}, \gamma = 8$, $\lambda = 4, \delta = 4, \alpha = \beta = 1$, and the password is $\Lambda = \{1, 2, 4, 6\}, \Delta = \{1, 2, 3, 4\}, W = 3124$, the identification procedure can be denoted by Fig. 11 (Fig. 3 of [57]).

Since this protocol is suggested to resist peeping attack, it is natural that Δ should be excluded from the password since only one observed identification will expose it. Matsumoto and Imai gave the success probability of a known-Δ random attack:

$$p_\Delta = \sum_{j=\alpha}^{\beta} \binom{\beta}{j} p^j (1-p)^{\beta-j} \leq \binom{\beta}{\max(\alpha, \lfloor \beta/2 \rfloor)} p^\alpha, \text{ where } p = \frac{1}{\delta^\lambda \left(1 - \sum_{i=1}^{\delta-1} \binom{\delta}{i} \left(\frac{i}{\delta}\right)^\gamma\right)}. \quad (1)$$

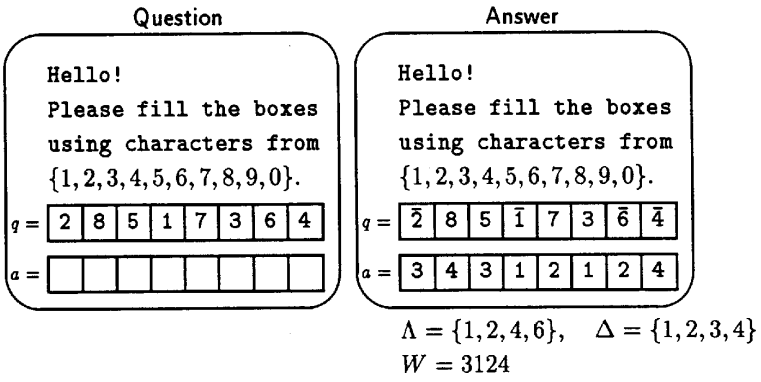


Figure 11: A simple example of Matsumoto-Imai Protocol

They also pointed out that the complexity of getting the password by (passive) peeping attack is at most $\binom{\gamma}{\lambda}$ trials if an adversary can observe some identifications.

4.1.2 Cryptanalysis of Mastumoto-Imai Protocol [58]

However, as C.-H. Wang et al. pointed out, there exist other two problems in Matsumoto-Imai protocol. Firstly, the complexity against passive peeping attack of Matsumoto-Imai protocol is less than $\binom{\gamma}{\lambda}$. Assume $\Delta = \{e_1, e_2, \cdots, e_\delta\}$, they gave the new calculated complexity as follows

$$\sum_{\substack{s_1+s_2+\cdots+s_\delta=\lambda \\ 1\le s_i\le \min(\delta,t_i)}} \binom{t_1}{s_1}\binom{t_2}{s_2}\cdots\binom{t_\lambda}{s_\lambda} < \binom{\gamma}{\lambda}, \tag{2}$$

where s_i is the occurrence number of e_i in a_j, and t_i is the occurrence number of e_i in q_j. Secondly, there exists a powerful active peeping attack – "replay challenge attack", and C.-H. Wang gave the following result:

Theorem *An intruder can find the window* Λ *in* $\sum_{k=0}^{\gamma-\lambda} \frac{\binom{\gamma-\lambda}{k}\binom{\gamma-k}{\lambda}(\delta^n-1)^k}{\delta^{n(\gamma-\lambda)}}$ *expected trials if the same question is replayed* n *times.*

From the above theorem, we can easily find the expected number of trials increase exponentially with n. For example, when $\gamma = 36, \lambda = 18, \delta = 2, \beta = 1$, the number is about 8165 when $n = 3$ and only 5 when $n = 7$. Also, the passive peeping attack and the replay challenge attack can be combined to carry a more powerful attack. As a result, Matsumoto-Imai protocol is not secure enough.

To improve the security of Matsumoto-Imai protocol, C.-H. Wang et al. suggested making all elements in a_j probabilistic, i.e., making the following fact holds: $\forall i = 1 \sim \lambda$, $a_j(f_j(i))$ is not always be $W(i)$. To do so, C.-H. Wang et al. added a constrain $\gamma = 2\lambda$ and introduced a new function to control how to probabilistically assign $W(1) \sim W(\lambda)$ to λ values of $a_j(f_j(1)) \sim a_j(f_j(\lambda))$. Unfortunately, when λ is not too small, the required deduction is too complicated for most humans. To finish the answer, it is possible that many people have to write down some intermediate results, which is undesired in a SecHCI protocol since the whole identification procedure can be observed by adversaries.

4.1.3 Our Notes

In [57, 58], the authors didn't give the required number of observed identifications in passive peeping attack. But it is rather important to measure the security against passive peeping attack (recall Sec. 3.3 and 3.5.2). Here, we can estimate this number: given a randomly selected window alphabet Λ, assume the probability of that two responses agree with Λ is always lower than $\frac{1}{\zeta^\lambda}$, then the probability of that all k responses agree with Λ is $\frac{1}{\zeta^{\lambda(k-1)}}$, thus the required number of observed identification is $O\left(\left\lceil \left(1 + \left\lceil \ln\binom{\gamma}{\lambda} \Big/ (\lambda \ln \zeta)\right\rceil\right) \Big/ \beta \right\rceil\right)$. Apparently, from the probability of $a_j(i)$, we can deduce one possible value of ζ: $\frac{1}{\zeta} = \max(\frac{\lambda}{\gamma}, 1 - \frac{\lambda}{\gamma}) \Rightarrow \zeta = \min(\frac{\gamma}{\lambda}, \frac{\gamma}{\gamma-\lambda})$. For Example 1 in [57], $\gamma = 36, \lambda = 18, \beta = 1$, $\zeta = 2$, this number is $O\left(1 + \left\lceil \ln\binom{36}{18} \Big/ (18 \ln 2)\right\rceil\right) = O(3)$; for Example 2, $\gamma = 50, \lambda = 10, \beta = 1$, $\zeta = 5/4 = 1.25$, so this number is $O\left(1 + \left\lceil \ln\binom{50}{10} \Big/ (10 \ln 1.25)\right\rceil\right) = O(12)$. We can see the number is very small, and it increases logarithmically as γ, λ increase. To enhance the security, we must increase $\binom{\gamma}{\lambda}$ to over $O(2^{60})$, such as $\gamma = 64, \lambda = 32$. However, too large γ, λ will make the usability worse.

Beside the modified version of C.-H. Wang et al. given in [58], there still exists some other possible modifications. The basic idea used by C.-H. Wang et al. is to change all "window-padding answer elements"

$a_j(f_j(1)) \sim a_j(f_j(\lambda))$ by "random-padding answer elements". Actually, we can use other algorithms to realize this task. For example, when $\gamma = 2\lambda$, we can add the i^{th} "random-padding answer element" to the i^{th} "window-padding answer element" and use a secret (or public) function $g : \Gamma \to \Delta$ to obtain the value of $a_j(f_j(i))$. When g is a linear function, it will be relative easy for humans to make right responses. Also, we can use the λ "random-padding answer elements" to re-order the λ "window-padding answer elements". Of course, when γ, λ is not too small, the usability cannot be enhanced essentially.

4.2 Matsumoto Protocols [81, 82]

After the proposal and its cryptanalysis of Matsumoto-Imai protocol, T. Matsumoto proposed another kind of SecHCI protocols in [81, 82][14]. In [82], the author presented a basic protocol and two extended protocols, and show the possibilities to use the two extended protocols to construct some visual identification SecHCI protocols. Here we will respectively introduce the three protocols and some interesting examples, and point out the problems of the proposed protocols.

4.2.1 Protocol 0

Let s be a prime power and u, v be positive integers. Also, let $\boldsymbol{F_s}$ and $\boldsymbol{F_s^v}$ respectively represent the finite filed of order s and the vector space $\{[x_1, \cdots, x_v] | x_1, \cdots, x_v \in F_s\}$. The shared password between H and C is $\boldsymbol{K} = [\boldsymbol{k}_1, \cdots, \boldsymbol{k}_u]$, where $\boldsymbol{k}_i^{\mathrm{T}} \in \boldsymbol{F_s^v}$. The identification procedure is as follows:

H⟶C:	H's ID	$(H0)$
C⟶H:	A random vector $\boldsymbol{q}_i \in \boldsymbol{F_s^v} - \{\boldsymbol{0}\}$	$(C1)$
H⟶C:	An answer $a_i \in \boldsymbol{F_s}$	$(H1)$
C⟶H:	If $a_i \neq \boldsymbol{q}_i \cdot \boldsymbol{k}_i$, then outcome `reject`	$(C2)$
C⟶H:	Outcome `accept`	$(C3)$

$\Big\} i = 1 \sim u$

Apparently, the above protocol can be described with another equivalent format: assume $\boldsymbol{Q} = [\boldsymbol{q}_1^{\mathrm{T}}, \boldsymbol{q}_2^{\mathrm{T}}, \cdots, \boldsymbol{q}_u^{\mathrm{T}}]$, and $\boldsymbol{A} = \boldsymbol{Q}^{\mathrm{T}}\boldsymbol{K} = [A(i,j)]_{i=1,j=1}^{i=u,j=u}$, then the right response is $\boldsymbol{a} = [A_{1,1}, A_{2,2}, \cdots, A_{u,u}] \in \boldsymbol{F_s^u}$.

In [82], aiming at three different attacks – "blind attack", "known Q and A attack" and "chosen Q and A attack", T. Matsumoto gave the success probabilities of random answers P_r. The "blind attack" is challenge-only attack, and the left two belong to peeping attack ("chosen Q and A attack" is an active peeping attack). The following is the related results about the three attacks:

Proposition 1 *For the blind attack,* $P_r = \frac{1}{s^u}$.

Proposition 2 *For the known Q and A attack,* $P_r = \left(\frac{1}{s} + \left(1 - \frac{1}{s}\right) \cdot \frac{1}{s^v - 1} \cdot \sum_{l=1}^{v} (s^l - 1) R(n,l)\right)^u$, *where* $R(n,l)$ *is calculated as follows: assume* $D = \{(n,l) | n \geq 1, 1 \leq l \leq v, l \leq n\}$, *1) when* $l = 0$ *or* $l > \min(n, v)$, $R(n,l) = 0$; *2) when* $(l,l) \in D$, $R(n,l) = \prod_{m=0}^{l-1} \left(1 - \frac{s^m - 1}{s^v - 1}\right)$; *3) when* $(n,l) \in D$ *and* $n \neq l$, $R(n,l) = \left(1 - \frac{s^{l-1} - 1}{s^v - 1}\right) \cdot R(n-1, l-1) + \frac{s^l - 1}{s^v - 1} \cdot R(n-1, l)$.

Proposition 3 *For the chosen Q and A attack,* $P_r = \left(\frac{1}{s} + \left(1 - \frac{1}{s}\right) \cdot \frac{s^n - 1}{s^v - 1}\right)^u$.

4.2.2 Protocol 1 and 2

T. Matsumoto claimed that the vector product $\boldsymbol{q}_i \cdot \boldsymbol{k}_i$ in Protocol 0 seems hard for many people, so it will be better to employ human vision capability to improve the usability of Protocol 0. Protocol 1 and 2 are designed based on the above hope.

Protocol 1 In this protocol, a set Ω is introduced to benefit users, whose size is s^v, i.e., the size of $\boldsymbol{F_s^v}$. The shared password is changed to u elements in Ω: $K = [k_1, k_2, \cdots, k_u]$, and the identification procedure is as follows:

H⟶C:	H's ID	$(H0)$
C⟶H:	A set of all elements in Ω and $Q_i = \{(\omega, q_i(\omega)) \| \omega \in \Omega\}$, where $q_i(\omega) = \boldsymbol{q}_i \cdot \psi_i(\omega) \in \boldsymbol{F_s}$ and ψ_i is a bijection from Ω to $\boldsymbol{F_s^v}$	$(C1)$
H⟶C:	An answer $a_i \in \boldsymbol{F_s}$	$(H1)$
C⟶H:	If $a_i \neq q_i(k_i)$, then outcome `reject`	$(C2)$
C⟶H:	Outcome `accept`	$(C3)$

$\Big\} i = 1 \sim u$

[14] We have not obtained a copy of [81] (we ever requested one via e-mail, but the authors did not response), but we can guess the SecHCI protocols in that paper should be similar to the ones in [82].

Protocol 1 has the similar security to Protocol 0. In Matsumoto's paper, two examples are given to show the feasibility of Protocol 1. Here, we introduce Example 2. Please see Fig. 12, where $s = 3, v = 4, u = 9$, $\boldsymbol{F_s} = \{1, 2, 3\}$, Ω is the set of $s^v = 81$ sampled names of railway stations, and Q_i is the set of pairs of station names and figures (or colors) appearing in small circles put on the locations of the stations. Generally speaking, such a SecHCI protocol has good usability.

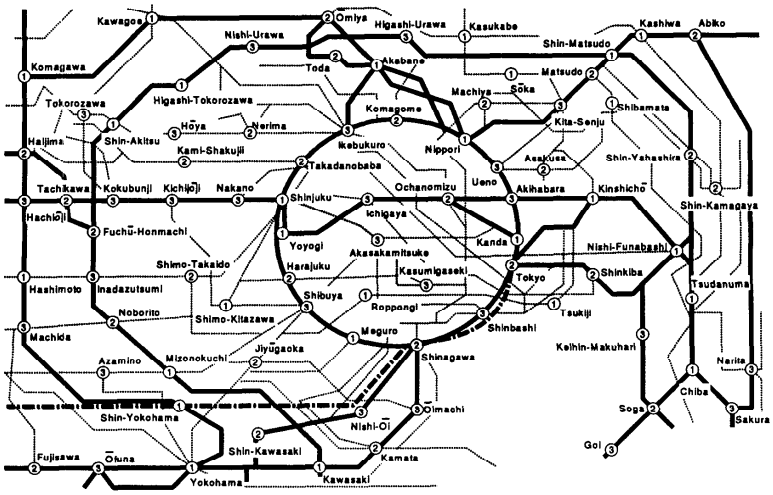


Figure 12: An example of Matsumoto Protocol 1 – Map Scheme

Protocol 2 Investigate Protocol 1, we can easily find u should be small enough to obtain good usability and s, v cannot be too large since the screen can only display limited objects. Protocol 2 is suggested by T. Matsumoto to balance the trade-off between security and usability. The basic idea is: given a positive integer $m \geq v$, divide v into m parts $v = \sum_{f=1}^{m} v_f, v_f > 0$; similarly, divide $\boldsymbol{q}_i$ and $\boldsymbol{k}_i^{\mathrm{T}}$ into m parts: $\boldsymbol{q}_i = [\boldsymbol{q}_{i1}, \cdots, \boldsymbol{q}_{im}]$ and $\boldsymbol{k}_i^{\mathrm{T}} = [\boldsymbol{k}_{i1}^{\mathrm{T}}, \cdots, \boldsymbol{k}_{im}^{\mathrm{T}}]$. By this way, the number of displayed object is reduced from s^v to $\sum_{f=1}^{m} s^{v_f}$, which is generally much smaller than s^v. Here, we omit the description of this protocol, and give Example 3 of [82] in Fig. 13.

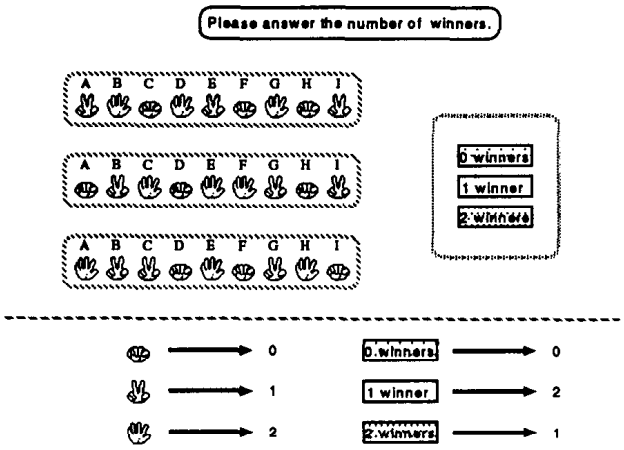


Figure 13: An example of Matsumoto Protocol 2 – Three Players Game Scheme

4.2.3 Our Notes

What about the security of the three protocols? Since Protocol 1 and 2 are only variants of Protocol 0. We can focus the security analysis on Protocol 0. It is obvious that Protocol 0 cannot resist passive peeping attacks with $O(v)$ observed identifications, since $\boldsymbol{k}_i$ can be uniquely solved by adversaries with v q_i/a_i pairs if the v q_i can generate a full-rank matrix. Experiments show that for n randomly generated vectors in $\boldsymbol{F}_{\boldsymbol{s}}^{v}$, they are independent (i.e., they can form a full-rank matrix) with high probability. In Fig. 14, we give the experimental result of the frequency for $s = 2 \sim 10$ and $v = 2 \sim 10$. We can see the frequency goes to 1 when $s \geq 10$ and $v \geq 4$, which confirm the result that $O(v)$ identifications are enough to uniquely solve the password.

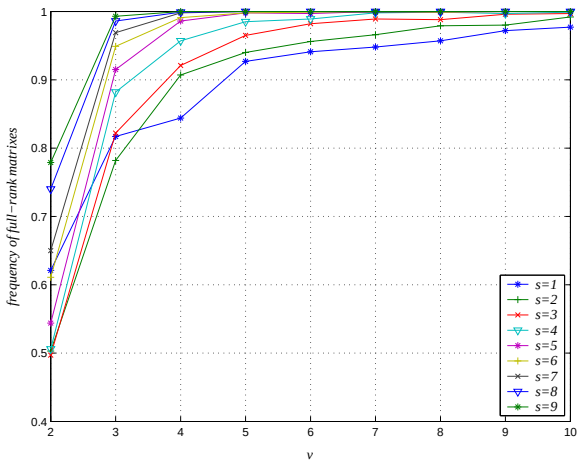


Figure 14: Occurrence frequency of full-rank matrixes in Matsumoto Protocol 0

In fact, although T. Matsumoto didn't consider the possibility of solving the password by $O(v)$ identifications, the success probability of known Q and A attack also exposes some information about the insecurity of Protocol 0. See Fig. 3.5 of [82], the fact that $P_r \approx 1$ for $n \geq 9$ implies that the complexity of Protocol 0 against passive peeping attack is about $O(v)$. As a result, we think Matsumoto protocols are weak SecHCI protocols and can only be used in some limited applications, such as the secure human-computer communications of one-time messages. Compared with Matsumoto-Imai protocol proposed in [57], Matsumoto protocols do not have stronger security than Matsumoto-Imai protocol.

4.3 Hopper-Blum Protocols [83, 84]

In [83, 84], N. J. Hopper and M. Blum proposed several SecHCI protocols. Because [84] is a more comprehensive version of [83], in the following context, we will only focus on the latter paper. In [84], the authors proposed two different protocols to respectively resist passive peeping attacks and active peeping attacks.

4.3.1 Protocol 1

Protocol 1 is based on the problem of *learning parity with noise* (LPN), which seems to be a NP-Hard problem. In this protocol, the secret password is a (0,1)-vector $\boldsymbol{x} \in \{0,1\}^n$ satisfying $|\boldsymbol{x}| = k$, The identification procedure can be described as follows:

H⟶C:	H's ID	$(H0)$	
C:	$i = 0$	$(C0)$	
C⟶H:	A random (0,1)-vector $\boldsymbol{c} \in_R \{0,1\}^n$	$(C1)$	m rounds
H⟶C:	With probability $1-\eta$, $r = \boldsymbol{c} \cdot \boldsymbol{x}$; and with probability η, $r = 1 - \boldsymbol{c} \cdot \boldsymbol{x}$	$(H1)$	
C:	If $r = \boldsymbol{c} \cdot \boldsymbol{x}$ then $i++$	$(C2)$	
C⟶H:	If $i \geq (1-\eta)m$ then outcome `accept`; otherwise outcome `reject`	$(C3)$	

In Theorem 1 and 2, N. J. Hopper and M. Blum claimed that the masquerading probability of $\mathcal{A}$ is $\left(\frac{1}{2}\right)^m \sum_{i=(1-\eta)m}^{m} \binom{m}{i} \leq e^{-c_0 m}$ $(c_0 \geq \frac{2}{3})$, and Protocol 1 is $(e^{-\frac{2}{3}m}, poly(n))$-secure against passive peeping attack. Also, the best known attack's complexity is claimed to be $\binom{n}{\lceil k/2 \rceil}$. They mentioned that the challenge vector c can be selected from $\{0,1,\cdots,9\}^n$, not $\{0,1\}^n$ to provide enough security with smaller m. A computer program is implemented with the parameter set of $n = 200, k = 15, m = 7, \eta = \frac{1}{7}$ to estimate the human-only executability of Protocol 1. The chief defect of Protocol 1 is that it cannot resist active peeping attacks, since $\mathcal{A}$ can record $n/m(1-\eta)^2$ successful identifications and replay them to find $(\boldsymbol{c}, r)$ pairs responded with the rule of $r = \boldsymbol{c} \cdot \boldsymbol{x}$.

Although the basic idea and the chief results about Protocol 1 are right, there still exist some errors and problems. Here, we will point out these problems and give more discussions on the security and usability of Protocol 1 in the following.

Problem 1 *The deduction of Theorem 1 and the result of Theorem 2 have nontrivial errors.* In Theorem 1, N. J. Hopper and M. Blum use "a Chernoff bound" to get $\left(\frac{1}{2}\right)^m \sum_{i=(1-\eta)m}^{m} \binom{m}{i} \leq e^{-c_0 m}$ $(c_0 \geq \frac{2}{3})$. But the deduction is wrong since there is not such a Chernoff bound [92] with which we can get $c_0 \geq \frac{2}{3}$. The

most like Chernoff bound is $Pr[X \geq \eta(1+\delta)] \leq e^{-\delta^2\eta/3}$, with which we can only get $Pr[i \geq (1-\eta)m] = \left(\frac{1}{2}\right)^m \sum_{i=(1-\eta)m}^{m} \binom{m}{i} \leq e^{-c_0 m}$, where $0 < c_0 = (1-2\eta)^2/6 < \frac{1}{6}$. As a natural result, the security against passive peeping attack proved in Theorem 2 is also false. We have contacted N. J. Hopper to discuss this error and he acknowledged that they really made a mistake when employing "a Chernoff bound" [85].

Fortunately, when η is not very close to $\frac{1}{2}$, it is still true that the masquerading probability decreases exponentially with the number of rounds, m. When $\eta = \frac{1}{2}$, the masquerading probability will be fixed at $\frac{1}{2}$ and independent of m. Generally speaking, $\eta = \frac{1}{m}$ may be OK, since the masquerading probability will be $\frac{m+1}{2^m}$.

To enhance the masquerading probability with smaller m, [84] suggested generating $\boldsymbol{c} =< c_1, c_2, \cdots, c_n >$ from $\{0, 1, \cdots, 9\}^n$ and calculate r with mod 10 arithmetic: $r = \sum_{i=1}^{n} c_i \cdot x_i \bmod 10 = \sum_{j=1}^{k} c_{i_j} \bmod 10$, where $i_1 \sim i_k$ is the k indexes of $x_{i_j} = 1$ in $\boldsymbol{x}$. With such a modification, the masquerading probability becomes $Pr[i \geq (1-\eta)m] = \sum_{i=(1-\eta)m}^{m} \binom{m}{i} \left(\frac{1}{10}\right)^i \left(\frac{9}{10}\right)^{m-i}$. When $\eta = \frac{1}{m}$, $Pr[i \geq (1-\eta)m] = \frac{9m+1}{10^m}$.

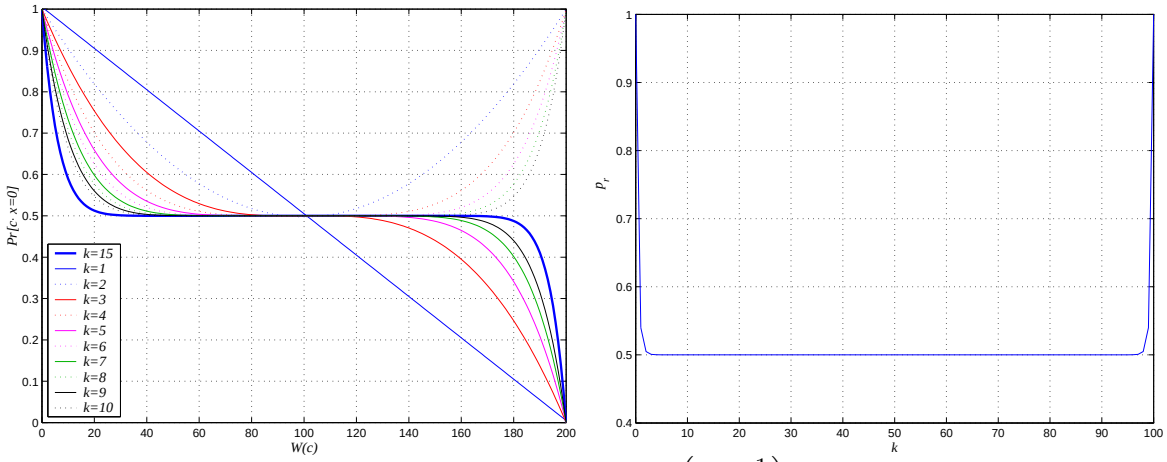
Problem 2 *The masquerading probability of each challenge can be absolutely greater than $\frac{1}{2}$ ($\frac{1}{10}$ for the decimal version), more detailed discussions should be made on the negative influence of such inequality.* It is true that $Pr[\boldsymbol{c} \cdot \boldsymbol{x} = 0] = Pr[\boldsymbol{c} \cdot \boldsymbol{x} = 1] = \frac{1}{2}$ for randomly-generated challenge (0,1)-vector $\boldsymbol{c}$, which is because

$$\begin{aligned} &(1-1)^k = C(k,0) - C(k,1) + C(k,2) - C(k,3) + \cdots \pm C(k,k) \\ &\Rightarrow C(k,0) + C(k,2) + \cdots = C(k,1) + C(k,3) + \cdots = \frac{2}{2^n} \\ &\Rightarrow \frac{C(k,0)\cdot 2^{n-k} + C(k,2)\cdot 2^{n-k} + \cdots}{2^n} = \frac{1}{2} \\ &\Rightarrow Pr[\boldsymbol{c} \cdot \boldsymbol{x} = 0] = Pr[\boldsymbol{c} \cdot \boldsymbol{x} = 1] = \frac{1}{2}. \end{aligned} \tag{3}$$

However, for any challenge (0,1)-vector $\boldsymbol{c}$ with specific weight $W(\boldsymbol{c})$, $Pr[\boldsymbol{c} \cdot \boldsymbol{x} = 0] = Pr[\boldsymbol{c} \cdot \boldsymbol{x} = 1]$ may be not always true and the two probabilities are uniquely determined by $W(\boldsymbol{c})$. Since adversaries can see $W(\boldsymbol{c})$, they can response $\boldsymbol{c}$ with higher probability than $\frac{1}{2}$ with the knowledge of $Pr[\boldsymbol{c} \cdot \boldsymbol{x} = 0]$ and $Pr[\boldsymbol{c} \cdot \boldsymbol{x} = 1]$ determined by $W(\boldsymbol{c})$. Theoretically, $Pr[\boldsymbol{c} \cdot \boldsymbol{x} = 0]$ and $Pr[\boldsymbol{c} \cdot \boldsymbol{x} = 1]$ can be calculated as follows:

$$Pr[\boldsymbol{c} \cdot \boldsymbol{x} = 0] = \frac{\sum_{i=0}^{\lfloor \min(W(\boldsymbol{c}),k)/2 \rfloor} \binom{k}{2i}\binom{n-k}{W(\boldsymbol{c})-2i}}{\binom{n}{W(\boldsymbol{c})}}, \text{ and } Pr[\boldsymbol{c} \cdot \boldsymbol{x} = 1] = 1 - Pr[\boldsymbol{c} \cdot \boldsymbol{x} = 0]. \tag{4}$$

In Fig. 15a, we give the values of $Pr[\boldsymbol{c} \cdot \boldsymbol{x} = 0]$ with respect to $W(\boldsymbol{c})$ when $n = 100, k = 1 \sim 10$. We can see when $W(\boldsymbol{c}) < 20$ or $W(\boldsymbol{c}) > 80$, there exist enough difference between $Pr[\boldsymbol{c} \cdot \boldsymbol{x} = 0]$ and $\frac{1}{2}$. When $W(\boldsymbol{c}) = 20 \sim 80$, $Pr[\boldsymbol{c} \cdot \boldsymbol{x} = 0] \approx \frac{1}{2}$. Generally speaking, there are only a limited number values of $W(\boldsymbol{c})$ satisfying $Pr[\boldsymbol{c} \cdot \boldsymbol{x} = 0] = \frac{1}{2}$. For example, when $n = 100, k = 1$, $Pr[\boldsymbol{c} \cdot \boldsymbol{x} = 0] = \frac{1}{2}$ only when $W(\boldsymbol{c}) = 50$; when when $n = 100, k = 1$, $Pr[\boldsymbol{c} \cdot \boldsymbol{x} = 0] = \frac{1}{2}$ only when $W(\boldsymbol{c}) = 55$ and 65; when $n = 100, k = 3 \sim 20$, $Pr[\boldsymbol{c} \cdot \boldsymbol{x} = 0] \neq \frac{1}{2}$ holds for all values of $W(\boldsymbol{c})$.



a) $Pr[\boldsymbol{c} \cdot \boldsymbol{x} = 0]$ with respect to $W(\boldsymbol{c})$ when $n = 200, k = 1 \sim 10, 15$

b) $\log_2\left(p_r - \frac{1}{2}\right)$ with respect to k when $n = 100$

Figure 15: Problem 2 of Hopper-Blum Protocol 1

From the above fact, $\mathcal{A}$ can calculate the $n+1$ probabilities of $Pr[\boldsymbol{c}\cdot\boldsymbol{x}=0]$ for $W(\boldsymbol{c})=0\sim n$, and response each challenge as the following rule: if $Pr[\boldsymbol{c}\cdot\boldsymbol{x}=0]>\frac{1}{2}$, then selects 0 with probability $1-\eta$ and selects 1 with probability η; otherwise selects 1 with probability $1-\eta$ and selects 0 with probability η. By this way, the masquerading probability of $\mathcal{A}$ is $p_r^m\sum_{i=(1-\eta)m}^{m}\binom{m}{i}$, where $p_r\geq\frac{1}{2}$. Of course, considering the non-uniform distribution of $W(\boldsymbol{c})$, if the challenge vector $\boldsymbol{c}$ is generated randomly, adversaries can only get a little benefit from the fact of $Pr[\boldsymbol{c}\cdot\boldsymbol{x}=0]\neq\frac{1}{2}$. We can get the masquerading probability p_r of $\mathcal{A}$ for randomly-generated challenges:

$$p_r=\frac{\sum_{W(\boldsymbol{C})=0}^{n}\max\left(F(W(\boldsymbol{c}),k),\binom{n}{W(\boldsymbol{c})}-F(W(\boldsymbol{c}),k)\right)}{2^n},\tag{5}$$

where $F(W(\boldsymbol{c}),k)=\sum_{i=0}^{\lfloor\min(W(\boldsymbol{c}),k)/2\rfloor}\binom{k}{2i}\binom{n-k}{W(\boldsymbol{c})-2i}$. In Fig. 15b, we give the values of $\log_2\left(p_r-\frac{1}{2}\right)$ with respect to k when $n=100$. We can see $p_r\approx0.5$ when $k=4\sim n-4$ (when $k=8$, $p_r\approx0.50000056613505$).

In addition, there exists another replay attack (active peeping attack), which is based on the probability difference between $Pr[\boldsymbol{c}\cdot\boldsymbol{x}=0]$ and $\frac{1}{2}$ (similar to the MLE attack discussed in Sec. 3.2 of [84]). If for challenge vectors with the same weight $Pr[\boldsymbol{c}\cdot\boldsymbol{x}=0]\neq\frac{1}{2}$, $\mathcal{A}$ can replay such vectors to H for multiple times to find the probability difference $Pr[\boldsymbol{c}\cdot\boldsymbol{x}=0]-\frac{1}{2}$ with the estimated probability $Pr[\boldsymbol{c}\cdot\boldsymbol{x}=0]$. In fact, such a difference is the reflection of the statistical distance between a non-uniform distribution A and the uniform distribution U defined on a same set $E=\{0,1\}$: $\Delta(A,U)=\frac{1}{2}\sum_{e\in E}|Pr_A[e]-Pr_U[e]|=\frac{1}{2}\left(\left|Pr[\boldsymbol{c}\cdot\boldsymbol{x}=0]-\frac{1}{2}\right|+\left|Pr[\boldsymbol{c}\cdot\boldsymbol{x}=1]-\frac{1}{2}\right|\right)$. Generally speaking, to confirm the above statistical distance $\Delta(A,U)$, averagely $O\left(\frac{1}{\Delta(A,U)}\right)$ samples are needed. In Fig. 16, we give the experimental results of $\log_2\left(\frac{1}{\Delta(A,U)}\right)$ with respect to $W(\boldsymbol{c})$ and k when $n=200$. We can see that when $W(\boldsymbol{c})$ and k is small, the number of required samples is not too large and can be used by $\mathcal{A}$.

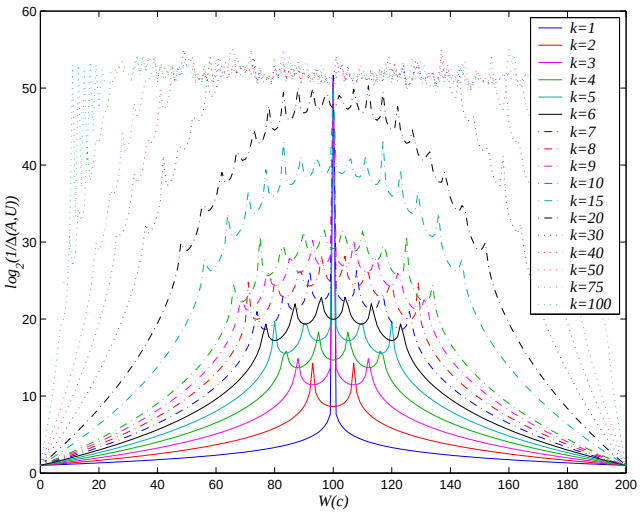


Figure 16: $\log_2\left(\frac{1}{\Delta(A,U)}\right)$ with respect to $W(\boldsymbol{c})$ and k when $n=200$

Carefully observe the above figure, we can find two interesting phenomena: 1) $\frac{1}{\Delta(A,U)}$ has k local maxima, and the distance between the smallest one and the greatest one has positive relationship with k, which holds for all values of k shown in Fig. 16; 2) $\frac{1}{\Delta(A,U)}$ has a obvious upper bound (about 2^{54}), does it imply some essential feature of $\frac{1}{\Delta(A,U)}$ and is it the reflection of the lower bound of $\log_2(p_r-\frac{1}{2})$ (see Fig. 15b)? Although it is less meaningful to investigate the reasons of the two phenomena, the results themselves are really interesting.

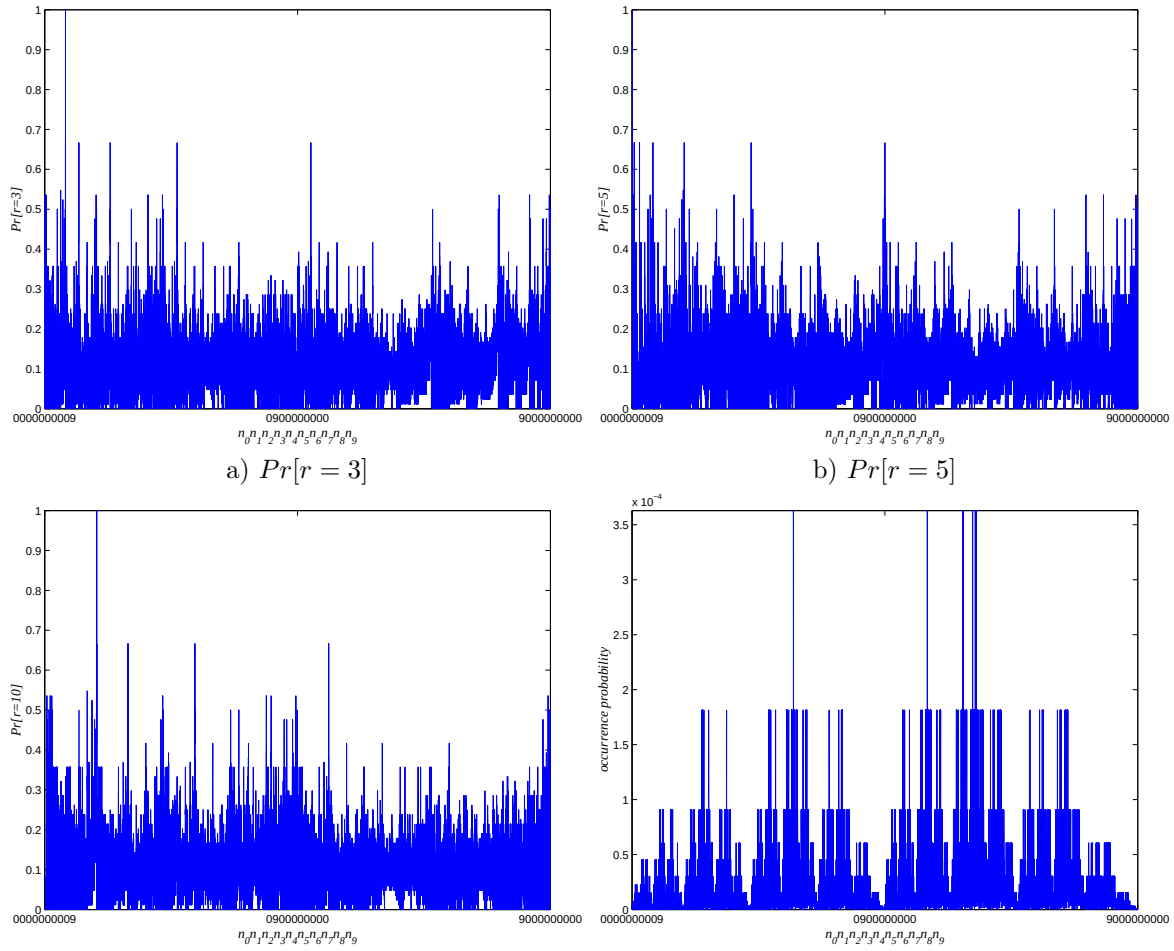
Problem 2 on Decimal Version of Protocol 1 Problem 2 still exists when $\{0,1\}^n$ is replaced by $\{0,1,\cdots,9\}^n$ to generate $\boldsymbol{c}$. In such a variant of Protocol 1, for random challenge vector $\boldsymbol{c}$, the masquerading probability is $\frac{1}{10}$ since $Pr[r=i]=\frac{1}{10}(i=0\sim9)$; while for a vector $\boldsymbol{c}$ with specific numbers of $0,1,\cdots,9$, $Pr[r=i]=\frac{1}{10}(i=0\sim9)$ will not always true and adversaries can response with higher masquerading probability than $\frac{1}{10}$. How can $\mathcal{A}$ calculate the masquerading probability? He can do so by firstly calculating the 10 probabilities $Pr[r=0],Pr[r=1],\cdots,Pr[r=9]$. The 10 probabilities can be defined as follows:

$$Pr[r=i]=\frac{\sum_{k_0+k_1+\cdots+k_9=k}B(i,k_0,k_1,\cdots,k_9)\binom{k}{k_0,k_1,\cdots,k_9}\binom{n-k}{n_0-k_0,n_1-k_1,\cdots,n_9-k_9}}{\binom{n}{n_0,n_1,\cdots,n_9}},\tag{6}$$

where n_i, k_i respectively represent the number of the digit i in $\boldsymbol{c}$ and in the k indexes of $\boldsymbol{x}$, and

$$B(i, k_0, k_1, \cdots, k_9) = \begin{cases} 1, & \sum_{j=0}^{9} k_j \cdot j = i \\ 0, & \sum_{j=0}^{9} k_j \cdot j \neq i \end{cases} . \tag{7}$$

$\mathcal{A}$ determines its response as follows: $\underset{i=0\sim 9}{\text{argmax}}(Pr[r=i])$. Of course, when n is large enough, it is impossible to calculate $Pr[r=i]$ for all values of $n_0, n_1, \cdots, n_9$, but $\mathcal{A}$ can instantly calculate them to make the response with higher success probability once he encounters a specific $\boldsymbol{c}$ (i.e., get the values of $n_0, n_1, \cdots, n_9$). The upper bound of the computation complexity is $\binom{k+9}{9} \times (4n+2)$ integer multiplications, where $\binom{k+9}{9}$ is the number of all possible values of $n_0, n_1, \cdots, n_9$ [15] and $4n+2$ stands for the complexity of 33 factorials. Since k should not too large to provide acceptable usability, $\binom{k+9}{9} \times (4n+2)$ will be not too large and online computation is possible. For the implemented system in [84] with the parameter $n = 200, k = 15$, the complexity is about $O(2^{30})$ (even when we increase n to 1000, the complexity will be increase not much large $O(2^{32.3})$ since it is proportional to n), which is acceptable for many stand-alone powerful computers and distributed computing systems (but for PCs with relative limited computation capabilities, 2^{30} complexity will still consume rather large time). In a PC with Pentium®III Xeon® 733 MHz CPU, we develop a Matlab® program to test the actual speed, $Pr[r=0] \sim Pr[r=9]$ can be calculated within half an hour for most values of $n_0, n_1, \cdots, n_9$. If other programming languages with better computing performance are used on more powerful PCs, it should be possible to calculate them within several minutes.



a) $Pr[r=3]$ b) $Pr[r=5]$

c) $Pr[r=10]$ d) The occurrence probability of $n_0n_1n_2n_3n_4n_5n_6n_7n_8n_9$ in randomly-generated $\boldsymbol{c}$

Figure 17: $Pr[r=3], Pr[r=5], Pr[r=10]$ with respect to $n_0n_1n_2n_3n_4n_5n_6n_7n_8n_9$ when $n=9, k=3$

To show the non-uniformity of $Pr[r=i]$ for different values of $n_0, n_1, \cdots, n_9$ more clear, with the parameters $n=9, k=3$, we exhaustively calculate all possible values of $Pr[r=i]$ and give the experimental results in Fig.

[15] This number is obtained as follows: from $\sum_{j=0}^{9} k_j = k$ and $k_j \in \{0, 1, \cdots, 9\}$, we can deduce $\sum_{j=0}^{9}(k_j+1) = k+10$. Assume $k'_j = k_j + 1$, we have $\sum_{j=0}^{9} k'_j = k+10$ and $k_j \in \{1, 2, \cdots, 10\}$. Consider the above problem with another idea: find 9 different separators in $\{2, \cdots, k+10\}$ to divide $\{1, 2, \cdots, k+10\}$ into 10 parts. Apparently, the number of all possibilities is $\binom{k+10-1}{10-1} = \binom{k+9}{9}$.

17. We can see that when n is small the values of $Pr[r=0] \sim Pr[r=9]$ are far different from $\frac{1}{10}$ with non-negligible probability. Fortunately, when n is large enough, $Pr[r=i] \approx \frac{1}{10}$ for almost all values of $n_0, n_1, \cdots, n_9$, and the values of $n_0, n_1, \cdots, n_9$ satisfying $Pr[r=i] \approx \frac{1}{10}$ occurs with much higher probability than the ones satisfying $Pr[r=i] \not\approx \frac{1}{10}$. For the the implemented system in [84], we calculate $Pr[r=1] \sim Pr[r=10]$ for some values of $(n_0, n_1, \cdots, n_9)$ and find out that $Pr[r=i] \approx 0.1$ holds for almost all possible values $(Pr[r=i] - 0.1 \lessapprox O(10^{-5}))$. As a summary, if $\mathcal{A}$ uses the above knowledge to enhance his guesses, we can get the following result similar to the above-mentioned one of (0,1)-version of Protocol 1: when the challenge vector is generated randomly, the masquerading probability p_r will be absolutely larger than $\frac{1}{10}$, but $|p_r - \frac{1}{10}|$ will decrease exponentially as n decreases, i.e., it will be close enough to $\frac{1}{10}$ if n is large enough ($n \geq 100$ may be OK).

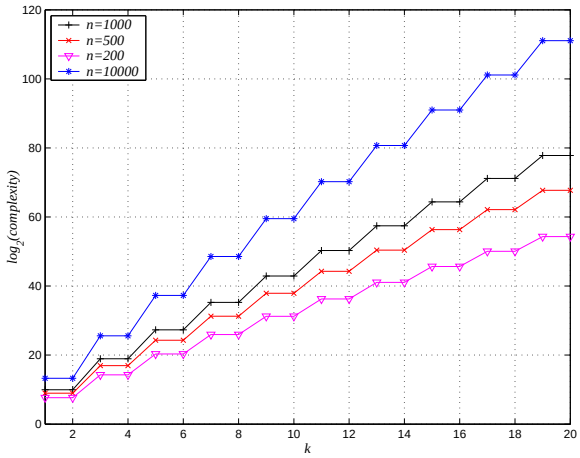


Figure 18: The best known attack's complexity with respect to k for $n = 200, 500, 1000, 10000$

Problem 3 To provide acceptable security against different kinds of attacks, all related values (n, m, k) should be large enough (for decimal version of Protocol 1, $m = O(8)$ may be OK to provide enough security against random response attack). Based on the assumption that the best known attack's complexity is $\binom{n}{\lceil k/2 \rceil}$, N. J. Hopper and M. Blum stated that $n = 1000, k = 19$ can provide $O(2^{78})$ security. In fact, the security is more dependent on k than on n, see Fig. 18 for the change of the complexity $\binom{n}{\lceil k/2 \rceil}$ with respect to k and n. That is to say, k should be large enough to provide enough security. However, too large k makes negative influence on the usability of Protocol 1 in the following two sides: 1) too large k makes humans difficult to remember the password, it has been known that $k = 8$ is the magic number of our memory [86, 87]; 2) too large k makes the computation load heavy (km decimal additions are need for users in Protocol 1). If we cannot use too large k, then we have to increase n to obtain enough security, for $k = 8$ (the most desired value from the consideration about the above memory magic), we can find even $n = 10000$ can only provide the security of $O(2^{49})$ (see Fig. 18). What's worse, too large n will make it inconvenient to display the challenge on the screen (in a monitor with the display resolution of 800×600, we can only show $80 \times 60 = 4800$ digits with 10×10 size) and further lower the usability of the identification system in practice.

Following the above analysis, the implemented system of decimal version of Protocol 1 in [84] has too low security for strict security applications: $\binom{n}{\lceil k/2 \rceil} = \binom{200}{8} \approx 2^{45.65}$. Then let us change to the following questions: for those applications without demands of high security (such as banking cards protected by only 4 or 6 digits), is Protocol 1 an acceptable solution? Can we use $k = 8$ to get acceptable security? The answer relies on the number of identifications that should be observed by $\mathcal{A}$. If the number is too small, then the complexity $2^{45.65}$ is not secure enough since the attack computation can be carried off-line.

Now let us investigate the average number of required observed identifications when the "meet in the middle" attack is effectively employed. Generally speaking, the "meet in the middle" attack can be meaningful only when the right password is unique to make the observed identifications, otherwise $\mathcal{A}$ has to exhaustively find all candidate passwords and wait for further identifications to judge which one is the right one (and the search complexity will be similar to the one of brute force attack). Of course, if the number of candidates is small enough, $\mathcal{A}$ can use them one by one to executing the identification system, and then determine the right one (please note that many applications limit the number of repeated wrong trials). Then the kernel of this question becomes: *what is the number of observed identifications to ensure the uniqueness of the right password to make all observed responses?*

For each observed $(\boldsymbol{c}, r)$ pair, there are averagely a half of all possible passwords can make the right response since $r \in \{0, 1\}$. If $\{0, 1, \cdots, 9\}^n$ is used to generate the challenge vector $\boldsymbol{c}$, averagely $\frac{1}{10}$ of all passwords can

do the same thing [16]. Following such an idea, with l observed identifications, averagely $\frac{1}{2^{lm}}$ of all passwords can make all observed responses (for decimal version of Protocol 1, this probability will be $\frac{1}{10^{lm}}$). Assume the focused probability is P_u, then the average number of passwords left is $N_u = \binom{n}{k}P_u$, which is $\binom{n}{k}/2^{lm}$ for the (0,1)-version of Protocol 1 and is $\binom{n}{k}/10^{lm}$ for the decimal version. Letting $N_u \leq 1$, we can get a lower bound of the number of observed identifications that can ensure the uniqueness of the password: $l \geq \frac{\log_2 \binom{n}{k}}{m}$ for the (0,1)-version of Protocol 1 and $l \geq \frac{\log_{10} \binom{n}{k}}{m}$ for the decimal version. For the implemented system in [84], $n = 200, k = 15, m = 7$, we can derive $l \geq 11$ for the (0,1)-version of Protocol 1 and $l \geq 4$ for the decimal version. That is to say, $O(11)$ observed identifications for the (0,1)-version, and $O(4)$ for decimal version are (probabilistically) enough to carry out the "meet in the middle" attack effectively with the attack complexity $O\left(\binom{n}{\lceil k/2 \rceil}\right)$.

Considering the above attack is an offline attack and can be carried effectively on PCs with limited computation capabilities[17], Protocol 1 is not suitable at least for applications concerning sensitive privacy and money, such as applications involved banking cards.

4.3.2 Protocol 2

Protocol 2 is based on another harder problem: given m pairs $(\boldsymbol{v}_1, u_1), \cdots, (\boldsymbol{v}_m, u_m)$, where $\boldsymbol{v}_i \in \{0, 1, \cdots, 9\}^n$, $u_i \in \{0, 1, \cdots, 9\}$, find a set $z = \langle (x_1, y_1), (x_2, y_2), \cdots, (x_k, y_k) \rangle \in (\{1, \cdots, n\} \times \{1, \cdots, n\})^k$ such that $u_i = f(\boldsymbol{v}_i, z)$ for all $i = 1, 2, \cdots, m$, where $f(\boldsymbol{v}_i, z) = \sum_{i=1}^{k} \min(\boldsymbol{v}_i[x_i], \boldsymbol{v}_i[y_i]) \bmod 10$. This problem is called *sum of k mins*, which can be uniquely solved when $m \geq \binom{n}{2}$ and leads to a sparse subset sum problem when $m < \binom{n}{2}$. When $m \geq \binom{n}{2}$, the solving complexity is $O\left(\binom{n}{2} \times \binom{n}{2}^3\right) = O\left(\binom{n}{2}^4\right) = O\left(\frac{n^8}{16}\right)$; and when $m < \binom{n}{2}$ the complexity of the best algorithm is $\binom{n(n-1)/2}{k/2}$, which is greater than $\binom{n}{k}$ when $k > 3$. In [84], the authors also discussed the complexity of a MLE (maximum-likelihood estimation) approach to solve this problem, as a result, they pointed out that k should be properly selected in order to ensure the number of required samples greater than $\binom{n}{2}$.

In this protocol, the secret password is two secrets $p_1, p_2 \in (\{1, \cdots, n\} \times \{1, \cdots, n\})^k$ and a secret digit $d \in \{0, 1, \cdots, 9\}$, the identification procedure can be described as follows:

H⟶C:	H's ID	$(H0)$	
C:	$i = 0$	$(C0)$	
C⟶H:	A random vector $\boldsymbol{c} \in \{0, 1, \cdots, 9\}^n$ satisfying $f(\boldsymbol{c}, p_1) = d$	$(C1)$	m rounds
H:	If $\boldsymbol{c}$ is not close to linear then report a network infiltration	$(H1)$	
H:	If $f(\boldsymbol{c}, p_1) \neq d$ then report a network infiltration	$(H2)$	
H⟶C:	Response with $r = f(\boldsymbol{c}, p_2)$	$(H3)$	
C⟶H:	If $r = f(\boldsymbol{c}, p_2)$ then $i{+}{+}$, otherwise outcome `reject`	$(C2)$	
C⟶H:	If $i = m$ then outcome `accept`	$(C3)$	

In Step $(C1)$ of the above protocol, the challenge vector $\boldsymbol{c}$ is randomly generated with some error-correction algorithm as follows[18]: $n = 100wh$, divide the $100wh$ elements of $\boldsymbol{c}$ into $w \times h$ square $\boldsymbol{S}_{1,1} \sim \boldsymbol{S}_{w,h}$ (all are 10×10 matrixes), for each square $\boldsymbol{S}_{i,j}$ choose 3 digits $(a_{i,j}, b_{i,j}, c_{i,j})$ uniformly at random, and then generate $\boldsymbol{S}_{i,j}$ by a linear equation $\boldsymbol{S}_{i,j}(x, y) = a_{i,j}x + b_{i,j}y + c_{i,j} \bmod 10$. Step $(H1)$ is realized by the following method: for each square $\boldsymbol{S}_{i,j}$, randomly choose a point $\boldsymbol{s} = (s_x, s_y) \in \{0, 1, \cdots, 9\}^2$ and an offset $\boldsymbol{r} = (r_x, r_y) \in \{0, 1, \cdots, 9\}^2$, then test $\boldsymbol{S}_{i,j}(s_x, s_y) = \boldsymbol{S}_{i,j}(s_x + r_x, s_y + r_y) - \boldsymbol{S}_{i,j}(r_x, r_y) + \boldsymbol{S}_{i,j}(0, 0)$. If only several points do not satisfy $\boldsymbol{S}_{i,j}(s_x, s_y) = \boldsymbol{S}_{i,j}(s_x + r_x, s_y + r_y) - \boldsymbol{S}_{i,j}(r_x, r_y) + \boldsymbol{S}_{i,j}(0, 0)$ (i.e., "$\boldsymbol{c}$ is close to linear"), we can use $\boldsymbol{S}_{i,j}(s_x + r_x, s_y + r_y) - \boldsymbol{S}_{i,j}(r_x, r_y) + \boldsymbol{S}_{i,j}(0, 0)$ to "self-correct" $\boldsymbol{S}_{i,j}(s_x, s_y)$.

The use of error-correction algorithm is to avoid slight changes of previous challenges, which can be replayed by $\mathcal{A}$ to reveal p_2 with about $O(n)$ well-chosen challenges (an active peeping attack based on intersecting). Of course, the above countermeasure also dramatically reduces the number of all possible challenge vectors from $10^n = 10^{100wh}$ to 10^{3wh-1} ($a_{i,j} = b_{i,j} = c_{i,j} = 0$ is excluded), and the distribution of all digits in the n elements of $\boldsymbol{c}$ is uniform, i.e., $\forall i \in \{0, 1, \cdots, 9\}$, the number of i contained in each legal challenge $\boldsymbol{c}$ is $\frac{1}{10} \cdot 100wh = 10wh$.

As N. J. Hopper and M. Blum mentioned in [84], the chief defect of this protocol is that the usability is much (should be twice?) worse than Protocol 1: $8km$ (should be $2km$?) base 10 sums and $2km$ mins should be calculated by H, with the parameter $k = 12, m = 6$ (with confidence 10^{-6}), the computation load is 576

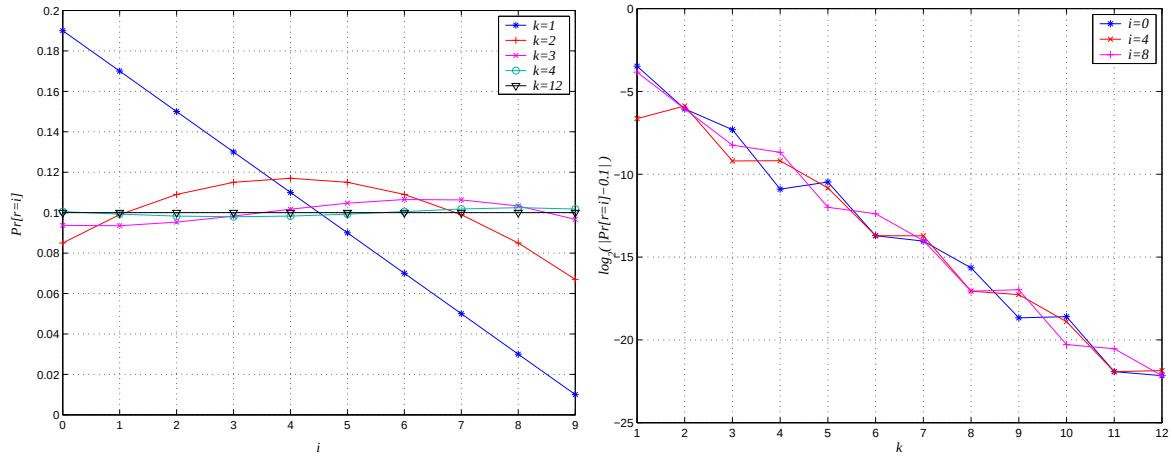
[16] Naturally, it is the price of the use of $\{0, 1, \cdots, 9\}^n$, which is used to get acceptable masquerading probability with small m.

[17] The computation complexity of $O(2^{45.65})$ will consume only 1.5 days for an advanced PC with the speed of $2^{30} = 1$G effective operations per second, and only 38 days for a old PC with the speed of $2^{24} = 16$M operations per second.

[18] Please note that we change many notations to make the explanation more clearer and more compatible with the description of this protocol.

(should be 144?) base 10 sums and 144 mins. In addition, some load should be made to test the linearity of $w \times h$ squares and to self-correct the modified elements in challenge vectors.

Similar to Protocol 1, the non-uniformity problem of $Pr[r = i](i = 0 \sim 9)$ (Problem 2) also exists in this protocol. In fact, the MLE approach to break the password is based on such non-uniformity. In Fig. 19, we give the values of $Pr[r = i](i = 0 \sim 9)$ for $k = 1, 2, 3, 4, 12$. Apparently, as k increases, the distribution of i is exponentially goes close to the uniform one (see Table 1 of [84]).



a) $Pr[r = i](i = 0 \sim 9)$ when $k = 1, 2, 3, 4, 12$ b) $\log_2(|Pr[r = i] - 0.1|), i = 0, 4, 8$ with respect to $k = 1 \sim 12$

Figure 19: $Pr[r = i](i = 0 \sim 9)$ of Protocol 2

4.4 HumanOIDs at Aladdin Center of CMU

At Aladdin Center [93] of Carnegie Mellon University, there is a HumanAut project [78] whose task is practical implementations of SecHCI protocols[19]. This project informally define their research as HumanOIDs [76]: "*HumanOIDs is a protocol that allows a naked human inside a glass house to authenticate securely to a non-trusted terminal. 'Naked' means that the human carries nothing: no smart cards, no laptops, no pencil or paper. 'Glass house' means that anybody can see what the human is doing, including everything that the human is typing.*" We can see the concept of HumanOID is equivalent to SecHCI defined in this paper.

4.4.1 HumanAut: An Image-Based Identification System

An image based prototype system has been placed on the web site to investigate its usability. In this system, there are n digital images, and your password is a set of $k = n/2$ images. The password images are "good" and other ones are "bad". The identification procedure is as follows:

H⟶C:	H's ID	$(H0)$	
C:	$i = 0$	$(C0)$	
C⟶H:	An images set $\{I_1, \cdots, I_m\}$, and a digits list $\{d_1, \cdots, d_m\}$, where $m = n/2$	$(C1)$	$t = 5$ rounds
H⟶C:	$r \in \{0, 1, \cdots, 9\}$	$(H1)$	
C:	If $r = \sum_{i=s_1}^{s_k} d_i \bmod 10$ then $i++$, where $s_1 \sim s_k$ are the indexes of "good" images in $\{I_1, \cdots, I_m\}$	$(C2)$	
C⟶H:	If $i = 4$ then outcome `accept`, otherwise outcome `reject`	$(C3)$	

A screenshot of the above system is shown in Fig. 20. In fact, this system can be considered as a variant of the decimal version of Protocol 1, except that the challenge vectors are generated with a specific rule $n_0 \geq n/2$ to constrain the number of images simultaneously displayed on the web page.

This system has an obvious defect that the length of the password is too long, when n is large, it is very difficult for humans to recall their passwords. See Fig. 21, can you remember only 6 "good" images immediately and recall them after one week? This defect is not discussed by the proposers. Here, we will try to find an acceptable (also practical and really interesting!) solution to this defect.

[19]The authors of [83, 84], M. Blum and N. J. Hopper, are the chief members of this project.

Figure 20: The HumanAut Project

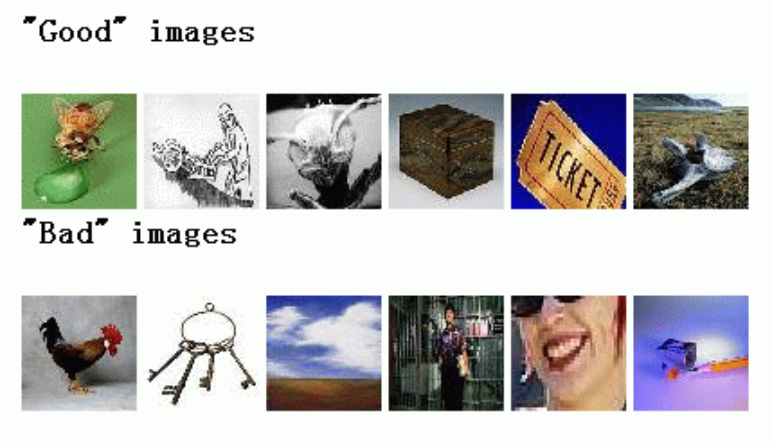


Figure 21: Examples of "good" images and "bad" images

4.4.2 Pass-Rule: An Approach to Remember Lengthy Password

Since it is very difficult for almost humans to accurately remember a lot of pass-images, why not try to find one or more focused features to represent all pass-images? Of course, to do so, the focused features must not be included in the "bad" images, thus, the identification system must support the function that users can freely change the images before they set new passwords or change their old passwords.

Applying this simple idea to the above HumanAut system, one can use a set of features to divide all n images into 2 different types, then the password will be the a set of features, which is called *pass-rule* and denoted by $\mathcal{P}_{\mathcal{F}} = \{\mathcal{F}_1, \cdots, \mathcal{F}_{k'}\}$ in this paper, where $\mathcal{F}_1 \sim \mathcal{F}_{k'}$ represent $k' \leq k$ chosen features and called *pass-features* in this paper. Apparently, our pass-rules can be considered as a visual/graphical version of the pass-algorithms in [44] and the word association in [36]. This idea seems very interesting, but it will make the password space collapse to a much smaller pass-rule space. So the kernel of the security changes to the design of pass-rules. Because the pass-rule is just used to enhance the memorability of the password (i.e., users do not need to tell computers anything about the pass-rule), if the number of all possible pass-rules are hard enough, it will be impossible for both adversaries and their powerful computers to guess the rules. Generally speaking, they should be both hard for humans and computers. Fortunately, because of the ambiguity and the huge number of all possible pass-rules, it is practically impossible for humans to guess them one by one. Thus, we can weaken the condition of "hard enough" to "hard enough for computers" and "hard enough to be distinguished by humans". Apparently, "hard enough for computers" just means the pass-rules should be the kernels of CAPTCHAs, "hard enough to be distinguished by humans" means the chosen pass-features should not be found by other humans easily with their brains and eyes.

Now let us give some details on the selection of pass-features. A class of features can be defined based on the contents of images, there are unlimited features included in this class. A simple example is: "good" images are cartoon pictures and "bad" images are real photos. However, if the chosen pass-feature is too simple (such as "cartoons vs. photos"), then the pass-feature may be not "hard enough to be distinguished by humans", since an adversary can distinguish it by naked eyes or try some possible guesses to determine it. This problem will become worse as n become greater, since too many $(n/2)$ "good" images can expose the pass-rule if all employed pass-features are too simple (you can easily find about a half of images are cartoons). To ensure the system security, we must reduce the possibility of such intelligent guesses. A possible solution is the abstract content, such as the drawing style of pictures, the type of different stories behind images (cosmic or tragedy, realistic or romantic, peaceful or dramatic, etc.), the relationship between the content of the images and you

(for example, if all images are photos of movie stars, then "good" images can be the images of the stars you dislike), the function of the focused article in images (for example, if all images are small icons of different articles, then "good" icons are the ones of the articles that are about meal, both foods and tools), etc.

Beside content-based pass-features, we can also use (independent of content) visual properties of images as pass-features, such as the density of edges (the level of details), the use of different colors (such as the chief colors in images) and strokes (for drawing pictures), the background color (light, or content) of one specific part of each image (for example, the mean light level or the existence of texts in the left upper corner), the relationship between different parts of images (for example, approximately balanced lights vs. high contrast lights, strongly different colors vs. approximate colors in four corners), the generation type of images (artificial vs. realistic, "made by many people" vs. "made by one person" vs. "made by computers"), etc. We can see that there also exist a large number of potential pass-features.

Actually, the final source of pass-features is your particular imagination and your private secrets, such as your own feelings on colors, shapes, visual representations of different articles, and even your special interests/hatreds on any object. To realize the target of "hard enough to be distinguished by humans", such secrets of your own may be the real kernel. I believe everybody can discover many pass-features, which are actually known by himself and very hard to expose without clues from himself. Consider the following fact: if you do not tell others your negative feeling on a politician, how can they find your secret by themselves? As a good example, see the images given in Fig. 22, even after I tell you which are "good" images (which actually seldom occurs in the real world), can you find out my pass-feature? You can see that, similar to your wallpaper, desktop scheme, and ICQ/WinAmp skin, it is possible that your password can become the next funny "game" on your PC.

Figure 22: Can you find out the pass-feature hidden in the above 12 images?

At last, please note that more than one different pass-features can be simultaneously used in your pass-rule, which will make the size of pass-rule space much larger and the cryptanalysis from both humans and computer (ro)bots much more difficult. In addition, reducing the length of the password can also be helpful to enhance the impossibility of the exposition of pass-rules[20] and improve the usability, $k = n/4$ may be OK.

More Discussions on HumanAut Of course, there are still some other problems in HumanAut, a relative serious one is about the consuming time of one identification. m should be large enough to provide high security against random response attack. If m is too small, $\mathcal{A}$ can use the following intersecting attack to obtain all "good" images: given n_o observed challenges, $\mathcal{A}$ exhaustively searches all possible combinations of "good" images, and tries to find the combinations agreeing with all responses. Consider the agreeing probability of n_o random responses is $\frac{1}{10^{n_o}}$ and one identification contains t challenge/response pairs, it is possible for $\mathcal{A}$ to find the password within $O(n)$ observed identifications. The above intersecting attack's complexity is about $O(2^m \times n)$, then it is obvious that $m \geq 60$ is required to provide the security of $O(2^{60})$. Averagely there are $m/2$ "good" images in each challenge, then $tm/2$ base 10 sums are required for one identification. Consider the masquerading probability of each challenge is $\frac{1}{10}$, to obtain high security against random response attack, 4-out-of-5 challenges are not enough (for ATM uses, $t = 5$ is still OK). If the required security is that the masquerading probability is smaller than 2^{-30}, then $t \geq 10$ is required. Assume $m = 60, t = 10$, averagely 300 base 10 sums are required for one identification, which is rather heavy for most people (the consuming time will be $O(300)$ seconds) and may make many users flee. To avoid this flaw, some new challenge/response rules should be introduced.

Basically, for applications (ATM uses) with low security needs, I think HumanAut will be useful, with the use of self-configurability, pass-rules and some other modifications. To employ it in general applications, some

[20] Generally speaking, the more frequently the pass-rule occurs, the more easily one can find it. Following such an idea, when "good" images are overwhelmed by "bad" images, it is more difficult for adversaries to find the pass-feature.

more studies should be made in the future. In addition, as we have mentioned before, self-configurability is also useful to resist malicious administrator attacks (see Sec. 3.5.3) and provide extra fun to users.

4.4.3 PhoneOIDs: SecHCI Protocols via Phone

In [77], M. Blum defined PhoneOIDs as "challenge-response protocols for use over the phone". We can see PhoneOIDs are a sub-class of SecHCI protocols. The additional requirement on SecHCI protocols is that PhoneOIDs should be practical over phone, i.e., the two parties of PhoneOIDs (the prover and the verifier) are both unaided people (but adversaries are not). Generally speaking, PhoneOIDs are the most difficult protocols in SecHCI protocols, since in general SecHCI protocols the verifier (computer) is permitted to have powerful computation capability.

Many different PhoneOIDs have been proposed in [77], but it has been known that all of them are not secure enough. Because there are only brief introductions of several PhoneOIDs, we cannot make comprehensive discussions on them. Here, we will only give the description of two selected protocols PhoneOID #80 and #82 to demonstrate the basic ideas used in Blum's PhoneOIDs, where H_v and H_p respectively represent the verifier and the prover.

PhoneOID #80 In this PhoneOID protocol, the password shared between H_v and H_p includes three functions: two transformation functions $f : \{A, \cdots, Z\} \to \{0, \cdots, 9\}$ and $g : \{0, \cdots, 9\} \to \{0, \cdots, 9\}$, one permutation function $f_p : \{A, \cdots, Z\}^k \to \{A, \cdots, Z\}^k$.

$H_p \longrightarrow H_v$:	H_p's ID	$(H_p 0)$
$H_v \longrightarrow H_p$:	A k-character English word $\langle a_1, \cdots, a_k\rangle \in \{A, \cdots, Z\}^k$	$(H_v 1)$
$H_p \longrightarrow H_v$:	A k-digit number $\langle x_1, \cdots, x_k\rangle \in \{0, \cdots, 9\}^k$	$(H_p 1)$
$H_v \longrightarrow H_p$:	Calculate the reordering word $\langle b_1, \cdots, b_k\rangle = f_p(\langle a_1, \cdots, a_k\rangle)$, if $x_1 = g(f(a_1) + f(a_2))$ and $\forall i = 2 \sim k$ $x_i = g(x_{k-1} + f(b_i))$ then outcome `accept`, otherwise outcome `reject`	$(H_v 2)$

PhoneOID #82 In this PhoneOID protocol, the password shared between H_v and H_p also includes three parts: one transformation function $f : \{A, \cdots, Z\} \to \{0, \cdots, 9\}$, one permutation function $g : \{0, \cdots, 9\} \to \{0, \cdots, 9\}$, and a K-digit number $\langle d_1, \cdots, d_K\rangle$. Please note the difference between k and K.

$H_p \longrightarrow H_v$:	H_p's ID	$(H_p 0)$
$H_v \longrightarrow H_p$:	A k-character English word $\langle a_1, \cdots, a_k\rangle \in \{A, \cdots, Z\}^k$	$(H_v 1)$
$H_p \longrightarrow H_v$:	A k-digit number $\langle x_1, \cdots, x_k\rangle \in \{0, \cdots, 9\}^k$	$(H_p 1)$
$H_v \longrightarrow H_p$:	Define $x_{i,1} = g((d_i + f(a_1)) \bmod 10)$ and $x_{i,j} = g((x_{i,j-1} + f(a_j)) \bmod 10)$, then do: if $\forall 1 = 2 \sim K$, $x_i = x_{i,k}$ outcome `accept`, otherwise outcome `reject`	$(H_v 2)$

From [77, 85], we have known none of PhoneOIDs are secure. Here, we would like to point out another defect of the above PhoneOIDs protocols: the size of the password is too large to remember for most people. In PhoneOID #80, the two transformation functions need us to remember 36 digits and the permutation functions needs us to remember a special rule; in PhoneOID #82, the password needs us to remember $36 + K$ digits. Apparently, it is rather hard for most people to remember so long passwords without any special memory training [86, 87].

5 Other Related Works

In this section,we will introduce some related works about SecHCI, including visual/graphical passwords, CAPTCHAs and some research about Human Interactive Proofs (HIPs), which are all helpful to provide promising ideas to the design and/or implementation of SecHCI protocols.

5.1 Visual/Graphical Passwords

We have mentioned in Sec. 1.2 that the idea of using human vision capability to realize visual/graphical passwords has attracted much attention recently. Many different solutions have been proposed, chiefly based on the following cognitive fact: humans can recall pictures more easily and quickly than words (generally with less accuracy) [46]. Such visual passwords can be considered as the most natural and simplest cases of HIPs. It is really meaningful to study how can we simplify the security world by our powerful (of course sometimes clumsy) intelligence. Maybe our cyber-life will be changed by those visual passwords tomorrow. For the design of SecHCI protocol, the ideas used in different visual/graphical passwords are useful to provide better usability with special implementations.

In Sec. 1.2, we have mentioned there are three chief classes of visual/graphical passwords: selective pictures based passwords, point-and-click passwords and drawing-based passwords. Here, we give some details about how those proposed visual/graphical passwords work.

5.1.1 Class I – Selective Pictures Based Passwords

The basic idea of this class is to input password by selecting m pictures in total $n > m$ displayed pictures. Two typical systems are Passface^{TM} [47–51] and Déjà Vu [52]. Here, we briefly introduce how they work.

Passface^{TM} [47–51] The identification procedure of Passface system contains the following phases: **I**) Users start by getting to know five (or more) faces - their passfaces. These are assigned by the system at random from a large library of anonymous faces. **II**) To authenticate a user, the system challenges them with a 3×3 grid of faces containing one passface and 8 decoy faces positioned randomly within the grid. **III**) The user responds by indicating the position of their passface in the grid. **IV**) This challenge-response process is repeated with each of the user's remaining pass-faces - each time presented in a grid with 8 more decoy faces. **V**) The user is authenticated once all passfaces have been recognized successfully. A screenshot of Passface^{TM} is given in Fig. 23, which is captured from [47].

Figure 23: Passface^{TM}: Selecting your faces

Déjà Vu [52] Déjà Vu has three phases: portfolio creation, training, and authentication. **I)** ***Portfolio Creation Phase*** – To set up a Déjà Vu image portfolio, the user selects a specific number of images from a larger set of images presented by a server. To avoid the potential defect induced from the guess of users' habits, Rachna Dhamija and Adrian Perrig suggested using Random Art [112] to generate random abstract images. Random arts depends only on the initial seed, so it is not necessary to store the images pixel-by-pixel, since the image can be computed quickly from the seed. All images are hand-selected to ensure consistent quality. **II)** ***Training Phase*** – After the portfolio selection phase, a short training phase is used to improve the memorability of the portfolio images. **III)** ***Authentication Phase*** – A trusted server stores all portfolio images for each user. Since each image is derived directly from the seed, the server only needs to store the seed. For each authentication challenge, the server creates a challenge set, which consists of portfolio and decoy images. If the user correctly identifies all portfolio images, she is authenticated. A screenshot of Déjà Vu is given in Fig. 24.

5.1.2 Class II – Point-and-Click Passwords

This class of passwords employ a simple idea: one click some secret positions in a specific image with (or without) a secret order. Two typical systems are PassPic [53] and the graphical "password windows" in Passlogix® v-GO^{TM} SSO (Single Sign-On) system [54–56]. In such visual passwords, fuzzy commitment mechanism (see the following sub-subsection) is generally needed to ensure the feasibility of users' inaccurate memory and clicking on the secret positions.

PassPic [53] A screenshot is given in Fig. 25. Please note that the black circles denote the range of fuzzy commitment.

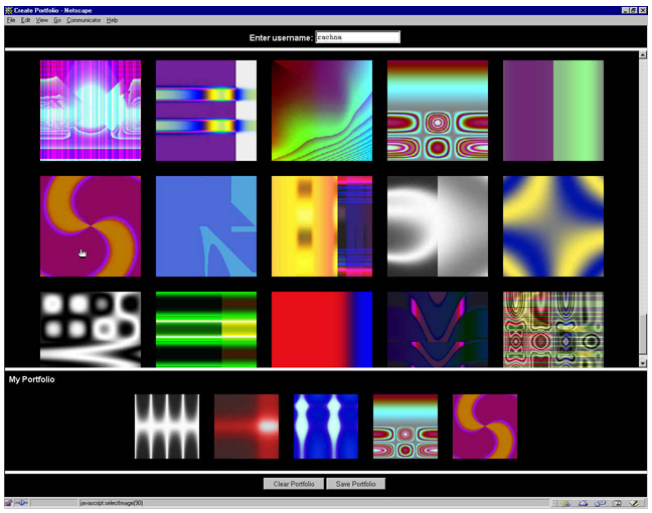


Figure 24: Déjà Vu: Selecting your portfolio images

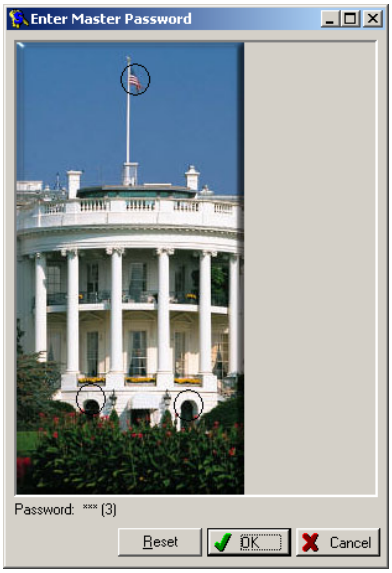


Figure 25: PassPic

Graphical "Password Windows" in Passlogix® v-GO™ SSO [54–56] Many similar but somewhat different ideas are used to realize the graphical "password windows", such as Cocktail, Lounge, Hand of Cards, Hiding Valuables, Make a Meal, Time, Phone, Stock, Trade, Periodic Table and Keyboard. A screenshot of Cocktail scheme is shown in Fig. 26, where the passwords are different combinations of drinks.

Figure 26: Graphical "password windows" in Passlogix® v-GO™ SSO: Cocktail

5.1.3 Class III – Drawing-Based Passwords

A typical system is DAS graphical passwords proposed by Ian Jermyn et al. at USENIX Security Symposium 99 (as both the best paper and the best student paper) [46]. The basic idea is to draw pass-strokes on a $m \times n$ grid. Theoretical and experimental analyses have shown that graphical passwords can provide better memorability and higher practical security than textual passwords. Some pass-strokes are shown in Fig. 27. Although the idea used in DAS password is rather simple, it is very promising as a solution to the password on PAD-like devices and also on common computers.

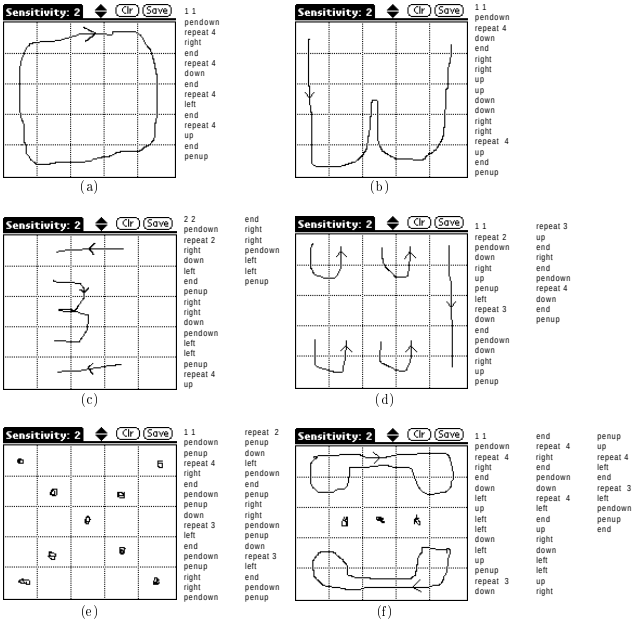


Figure 27: Drawing-a-Secret (DAS) Graphical Passwords

5.2 CAPTCHA (or Reverse Turing Test)

CAPTCHA stands for "Completely Automated Public Turing Test to Tell Computers and Humans Apart", which is named by M. Blum [77, 94, 95]. "Public" means that means that commented source code is made available to anyone who wants it, which is widely known as the Kerckhoffs' principle in cryptology [96]. "Completely Automated" means programmable, i.e., the Turing test should be carried and verified by a computer, not by humans (it is the reason that some researchers call CAPTCHA *reverse Turing test* [97]). Following the above definition, a program that can generate and grade tests that distinguish humans from computers, but whose code or data are private, is not a CAPTCHA.

The chief use of CAPTCHAs is to fight against different kinds of malicious (ro)bots from the viewpoints of users and/or service providers. Possible applications include: online polls, online registration, filtering junk mails (by mail servers). A possible use of CAPTCHA in the design of SecHCI protocol is: a CAPTCHA can be used to relax the security against random response attack, which has been mentioned in Sec. 3.5.1. An important problem about CAPTCHA is stealing cycles from humans, which has been discussed in [94].

CAPTCHA is a really novel concept emerging at the junction between cryptography and humanity. The basic idea of CAPTCHA is firstly introduced by M. Naor in Sep. 1996 [98]. In his paper, Naor pointed out that many AI problems can be used as candidates to fulfill such a task considering the absence of intelligence, including *gender recognition, facial expression understanding, finding body parts, deciding nudity, naive drawing understanding, handwriting understanding, speech recognition, filling in words (NLP problem), disambiguation (NLP problem).* In late 1997, Andrei Broder, the preceding chief scientist of Alta Vista has described a scheme to prevent robots of automatically submitting URLs. The related patent is United States Patent 6,195,698 [99]. In Sep. 2000, Udi Manber, the current chief scientist of Yahoo! Inc., visited UC Berkeley for seeking solutions to the "chat room problem", which is about the malicious chatting robots. Soon M. Blum, Luis von Ahn and John Langford at CMU developed an initial version of Gimpy [100] for Yahoo! and founded the CAPTCHA Project [95], which is now supported by Aladdin Center of CMU [93] and has been used by Yahoo!

In the following context, we will give a comprehensive survey on the state-of-the-art of current CAPTCHA technology. Most works on this topic has been presented at the first workshop on Human Interactive Proofs (HIPs) in Feb. 2002 and can be found at the web site of Aladdin Center of CMU [93].

5.2.1 Gimpy

Gimpy [100] is a very reliable system developed by the CAPTCHA Project. "It was originally built for (and in collaboration with) Yahoo! to keep bots out of their chat rooms, to prevent scripts from obtaining an excessive number of their e-mail addresses, and to prevent computer programs from publishing classified ads. Gimpy is based on the human ability to read extremely distorted and corrupted text, and the inability of current computer programs to do the same. Gimpy works by choosing a certain number of words from a dictionary, and then displaying them corrupted and distorted in an image; after that Gimpy asks the user to type the words displayed in that image. While human users have no problem typing the words displayed, current bots are simply unable to do the same." Now there are currently several versions of Gimpy, and two ones can be accessed in the web site of Gmipy: Gmipy – http://www.captcha.net/cgi-bin/gimpy, and EZ-Gmipy – http://www.captcha.net/cgi-bin/ez-gimpy. Also, the implementations of the two versions are downloadable respectively at http://www.captcha.net/gimpy_install.tar and http://www.captcha.net/ez-gimpy_install.tar. See Fig. 28 for some screenshots of the two versions. Please visit http://edit.yahoo.com/config/eval_register?.src=www&.done=http://www.yahoo.com for the implementation at Yahoo!

a) Gimpy b) EZ-Gimpy (used at Yahoo!)

Figure 28: Gimpy and EZ-Gimpy

In our opinions, the basic idea used by Gimpy aims at the AI problem of distorted character recognition in complex background, which has not been solved by AI community till now. Thus, it is a very good candidate for practical CAPTCHAs. In fact, there are still some others text recognition based CAPTCHAs developed by other parties, including the one used at AltaVista [101] and Pessimal Print based CAPTCHA [97].

5.2.2 Another Gimpy-Like CAPTCHA@AltaVista

One Gimpy-like CAPTCHA (maybe the first one) has been developed in late 1997 to prevent bulk URL submission by robots [101]. This system was taken in use at AltaVista in 2001 and "reduced the number of spam add-URL by over 95%". Now, it has been in use at http://addurl.altavista.com/sites/addurl/newurl.

The conceptual work of this system was done at Compaq SRC by Andrei Broder, Martin Abadi, Krishna Bharat and Mark Lillibridge. The implementation used by AltaVista was done by Laurent Chavez. The related IP rights US patent 6,195,698 [99] is owned by Compaq. Basically, this system only use the distortion of the characters without the generation of complex background, but it has similar performance to Gimpy. See Fig. 29 for a screenshot of this system at ALtaVista.

Figure 29: Gimpy-Like CAPTCHA used at AltaVista

5.2.3 Pessimal Print

Following Gimpy realized by M. Blum in 2000, A. L. Coates (now he is a collaborators of the CAPTCHA Project) et al. also discussed how to use pessimal print to construct CAPTCHAs in [97, 102]. What is pessimal print? That is low-quality images of machine-printed text synthesized pseudo-randomly over certain ranges of words, typefaces and image degradations (see Fig. 30). With some experiments, the authors shown that judicious choices of these ranges can ensure that the following fact: the generated images are legible to humans but illegible to several of the best present-day OCR machines. Based on this result, pessimal print can be used to implement some CAPTCHAs.

Figure 30: Pessimal Print

5.2.4 Visual Pattern Based CAPTCHA: Bongo

Bongo [103] is a CAPTCHA that asks the user to solve a visual pattern recognition problem. In particular, Bongo displays two series of blocks, the left and the right series. The blocks in the left series differ from those in the right, and the user must find the characteristic that sets the two series apart. A possible left and right series are shown in Fig. 31 (these two series are different because everything in the left is drawn within the upper half part, while everything in the right is drawn within the bottom half part). After seeing the two series of blocks, the user is presented with four single blocks and is asked to determine whether each block belongs to the right series or to the left. The current (beta) version of Bongo is available at http://gs9.sp.cs.cmu.edu/cgi-bin/bongo.

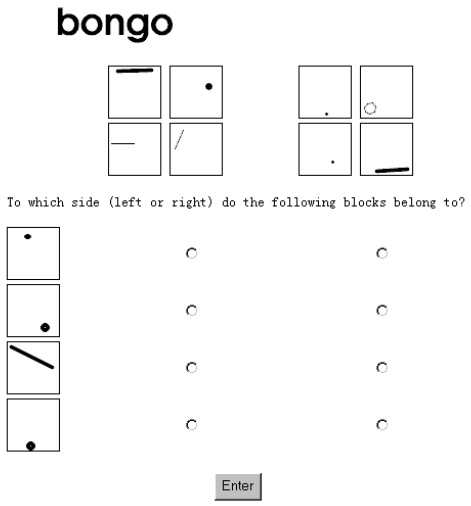


Figure 31: Bongo

5.2.5 Image Based CAPTCHA: PIX

PIX [104] is a program that has a large database of labeled images. All of these images are pictures of concrete objects (a horse, a table, a house, a flower, etc). The program picks an object at random, finds 6 random images of that object from its database, **distorts them at random**, presents them to the user and then asks the question "what are these pictures of?" (See Fig. 32). Current computer programs should not be able to answer this question. It is very important to randomly distort the images before their display to users, otherwise PIX will be not a CAPTCHA since the image database (and the image labels) should be public [94]. Here, please note that the basic idea used in PIX is a kind of pass-feature we have mentioned in Sec. 4.4.

5.2.6 Sound Oriented CAPTCHAs: Sounds and Byan

Sounds [105] is a sound version of Gimpy and also developed by the CAPTCHA Project at CMU. The program picks a word or a sequence of numbers at random, renders the word or the numbers into a sound clip and distorts the clip. It then presents the distorted sound clip to its user and asks the user to type in the contents of the sound clip. Because of unclear reason, there is no more online details about Sounds.

Byan [106, 107] is similar to Sounds and developed by Nancy Chan (now she is a collaborators of the CAPTCHA Project) at City University of Hong Kong. It is based on the abilities of humans to pay attention to a particular sound source. In each Byan test, a user is given a sound file. In the sound file, there are 2 voices: one of them expresses 6-digit in the chosen language, and the other one expresses some words in another language. In order to pass the test, you have to enter what 6-digit you heard. In latest version of Byan, new feature was

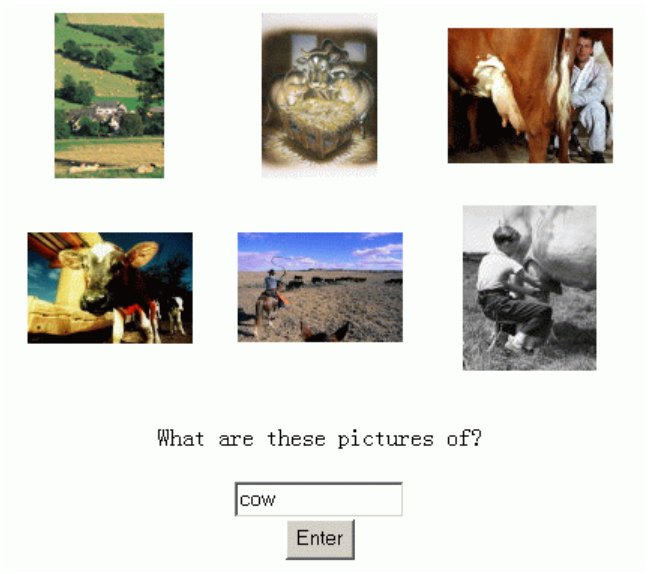


Figure 32: PIX

added: In order to make the tests easier to human, 1-digit wrong is accepted. For example, if the exact answer is "123456", then "023456", "193456" etc. will be count as "PASS", but "123467" will still be count as "FAIL". All the speeches are generated immediately by sending request to the online demo version of the tts sysnthesizer of Bell-Labs. Its fourth version Byan-IV can be accessed at http://144.214.74.131/cgi-bin/scaptcha4.cgi.

5.2.7 CAPTCHAs Based on Spoken Language Interfaces

In [108], Daniel Lopresti et al. at Bell Labs presented their works on employing speech as mutual interfaces (both for humans and computers) to design CAPTCHAs. Daniel Lopresti et al. presented three methods to design speech based CAPTCHAs: 1) making the dialog difficult for machines (high level: syntactic); 2) making the speech signal difficult for machines (low level: recognizing phonemes); 3) they processing the speech to create aural illusions that affect humans but not computers. The most intriguing method is the third one, since it provides an entirely different (converse) way to design CAPTCHAs. The first two methods are similar to Sounds and Byan we just discussed, but provide more details on how to distort the speech to make it difficult for computers. An example of the third method is based on the sensitivity of human to the distance between $f1$ (the first formant) and $f0$ (pitch, or fundamental frequency) of vowel (computers has less sensitivity).

5.2.8 Text-Only CAPTCHAs

As Philip B. Godfrey pointed out in [109], there are some reasons to support the use of text-only CAPTCHAs: 1) text is a new (and the simplest) media; 2) blind people cannot pass vision based CAPTCHAs; 3) a text-only interface would be more convenient in certain situations, such as a (text-only) dumb terminal connected with a UNIX system. Unfortunately, compared with visual and aural CAPTCHAs, it seems much difficult to design a good text-only CAPTCHA. What is the reason? This puzzle has been investigated by M. Blum and Lius von Ahn in [77, 110] and Philip B. Godfrey in [109].

In [77, 110], M. Blum and Lius von Ahn proved the impossibility of an English-text-only CAPTCHA based on a general model and six assumptions. The following content (until the end of the proof) is extracted from [77] with trivial modifications.

The Model: A text-only CAPTCHA is a randomizing algorithm with access to a large public data base (the world wide web as perceived through GOOGLE) that carries on a conversation with an opponent – either a bot that has access to GOOGLE or a human. The conversation proceeds in stages, starting with stage 1. In each stage, the CAPTCHA presents a CHALLENGE and the opponent gives a RESPONSE. At that point, the end of a stage, the CAPTCHA either ACCEPTS (believes its opponent to be human), REJECTS (believes its opponent to be a bot), or continues on to the next stage. The CAPTCHA is required to make its final decision (ACCEPT or REJECT) in a small number of stages.

More Details on the Model: The CAPTCHA initially sets stage $k = 1$. In STAGE k: 1) CAPTCHA generates a random #, random(k), and then uses it[21] to generate public challenge(k) and private state(k).

[21] If $k > 1$, it also uses its history($k-1$) of conversation and its state($k-1$) – EXCLUSIVE of all random $\#s$ generated in previous stages.

2) It awaits/gets public response(k). 3) It then uses its current history(k)= $\langle$challenge(1),response(1);$\cdots$; challenge(k), response(k)$\rangle$ and current state(k) to evaluate: ACCEPT, REJECT, or continue.

Assumptions:

1. RANDOM NUMBERS are used, if at all, only to create a public challenge (a continuation of the conversation) and some small amount of private state information. The CAPTCHA never uses its random numbers, if any, to decide between ACCEPT, REJECT, or continue (the conversation).

2. CONSISTENCY (HISTORY IRRESPECTIVE OF STATE DETERMINES ACCEPTANCE or REJECTION): At the end of stage k, the decision to ACCEPT, REJECT, or continue is completely determined by history(k). The only purpose of state(k) is to help the CAPTCHA to decide EFFICIENTLY – whether to ACCEPT, REJECT, or continue.

3. CONTINUE $\Rightarrow$ A PATH TO ACCEPT EXISTS: If, at the end of a stage, the CAPTCHA neither ACCEPTS nor REJECTS, then in the next stage the CAPTCHA is guaranteed to present a challenge (continuation of the conversation) for which there exists a non-rejectable response – one that causes the CAPTCHA to ACCEPT or continue.

4. CHALLENGE POSSIBLE $\Rightarrow$ THAT (SAME) CHALLENGE PROBABLE: Given history($k-1$) and challenge(k), it is efficiently possible for a bot to find a random # and a state($k-1$) that causes its own private virtual copy of the CAPTCHA to generate the same challenge(k), and therefore to evaluate response(k).

5. NONREJECTABLE RESPONSE POSSIBLE $\Rightarrow$ SOME (POSSIBLY DIFFERENT) NONREJECTABLE RESPONSE IS PROBABLE: If a response is not rejected by the CAPTCHA, then a random response has nontrivial probability (greater than 1% say) to not be rejected.

6. ALL BRANCHES ARE FINITE: Every branch is a FINITE sequence consisting of ROOT, challenge(1), response(1), (continue), $\cdots$, challenge(k), response(k), ACCEPT or REJECT. It follows by Knig's Lemma (compactness) that the entire game tree is finite.

Theorem *In the above model and under the above assumptions, an English-text CAPTCHA is impossible.* *Proof*: Assume to the contrary that a CAPTCHA is possible. We construct a bot that is accepted by the CAPTCHA, thus falsifying our assumption to the contrary.

The bot works as follows:

STAGE 1. When presented with challenge(1), it searches for a random number that causes its own private virtual copy of the (public) CAPTCHA to generate (the same) challenge(1). This is possible by assumption 4. This challenge(1) has a nonrejectable response, by assumption 3. The bot can find a nonrejectable response(1) by running its virtual CAPTCHA starting from the state it entered after generating challenge(1) – on randomly chosen responses, to look for a nonrejectable response. For each rejecting response, it reinitializes the CAPTCHA back to where it had just generated challenge(1) and runs it again on another random response. It does this again and again until it finds a nonrejectable response(1). It is sure to find such nonrejectable response(1) by assumption 5. The bot then supplies this response(1) to the (actual) CAPTCHA.

STAGE k: The actual CAPTCHA now generates challenge(k). The bot looks for this challenge(k) and a nonrejectable response(k).

This continues until the CAPTCHA ACCEPTS or REJECTS, which must happen since, by assumption 6, the tree is finite, and by assumption 3, "continue" is not a leaf. By assumption 3, the branch cannot lead to REJECT, so the branch must end in ACCEPT.

Somewheres above we need to make the point that if the actual CAPTCHA and the virtual CAPTCHA have the same history up to but not including the final ACCEPT or REJECT, then either both ACCEPT or both REJECT. This follows from assumption 2. ■

A Note on Text-Only CAPTCHAs How does the OCR-based CAPTCHA circumvent the above Theorem? The CAPTCHA chooses one of a very large set of random numbers to select the challenge (a distorted image) and the state information (the actual word). The opponent cannot guess the random number (efficiently). It therefore cannot simulate the generation (of a copy) of the actual challenge.

The Blum's Theorem is based on the possibility of a bot to efficiently construct a virtual CAPTCHA, then by which to get a virtual copy of the public challenge and a nonrejetctable response. We can see that Assumption 4 is the kernel of the design of practical CAPTCHAs. To design a good CAPTCHA, a basic task is to ensure the practical impossibility for a bot to find a same challenge (and the corresponding response) generated by this CAPTCHA. A natural way is to increase the number of possible challenges to a lower bound, $O(2^{30})$ should be enough considering the matching of images and audio is much slower than the one of texts. For most proposed

visual/aural CAPTCHAs, this requirement is generally fulfilled by the distortion mechanism (including noise adding and background complicating).

In [109], Philip B. Godfrey proposed a "Find the Bogus Word" text-only CAPTCHA and analyzed its two defects. One defect is about the possibility of the use of statistical model to predict the bogus word, another is so-called the "Source Text Problem", which is on how to prevent the capabilities of a bot to generate the source text given a challenge. Apparently, it is the problem of how to disable Assumption 4 in Blum's Theorem. As a promising candidate, Godfrey introduced a "Total Trigram Transformation" (just like the distortion in visual/aural CAPTCHAs) to replace each word in the challenge sentences with a new word. Based on such a transformation, a possible text-only CAPTCHA works as follows: Each challenge contains two sentences, one is the GOOD sentence that is directly extracted from a corpus, another is the BAD sentence that is generated from with a trigram model trained on the corpus. Transform both sentences, present two transformed sentenced to the user and let him to tell which one is transformed from the GOOD sentence. The hope is that enough of the structure of the sentence will be preserved that a human will be able to decide which sentence "flows" better, and that the sentence is obfuscated enough that a bot cannot pass by searching the corpus for the original sentence. It is claimed that preliminary tests show humans can pass better than guessing. However, in our opinion, because humans have less endurance on the distortions of texts than the ones of images and sounds and there exist less distortion algorithms, it seems much hard to design a text-only CAPTCHA with similar or better performance than visual/aural CAPTCHAs.

5.2.9 Chinese CAPTCHAs?

In [111], Ke Yang discussed some issues for Chinese CAPTCHA. The issues include: 1) Chinese is denser – there are many short sentences of complex meanings and many idioms, so that it may be possible to develop some CAPTCHAs based on such density; 2) Chinese Calligraphy allows more deformation than English, which may be useful in the design of Chinese CAPTCHAs; 3) Issues lying between simplified and traditional Chinese. Till now, no Chinese CAPTCHAs has been reported.

5.2.10 More Image-Based CAPTCHAs?

In [77], the topic of "CAPTCHA and the image recognition problem" was discussed and the feasibility of some more CAPTCHAs based on IMAGE SEARCH problem was studied. The following content is extracted from [77] with some modifications.

The IMAGE SEARCH problem: Given a slightly distorted fragment of an image, find the original or a close match in the image data base. It is likely that image search is a hard AI problem. It would be valuable to be able to find an image in a database, given some slightly distorted portion of the picture. The fundamental problem in such CAPTCHAs is: How can one transform a picture so that the result is virtually distinguishable to a human but hard for a computer to look up?

Some CAPTCHAs based on the IMAGE SEARCH problem: A great many CAPTCHAs are included in this class:

1. Challenge: What is common to these 5 pictures? Response: Fork. Luis von Ahn suggests a CAPTCHA that picks a "picturable noun" from BASIC ENGLISH and searches on the web for 5 images indexed by that noun. It is obvious that PIX is such a CAPTCHA.

2. Challenge: What's wrong with this picture? Response: The car is in the tree. Take a piece of one picture and blend it into another. Then distort the whole so that it can't be looked up. Or make a picture transparent and overlay a small piece of it on another picture before distorting.

3. Challenge: Who is this? Response: George Washington. There are lots of pictures of well-known people and places.

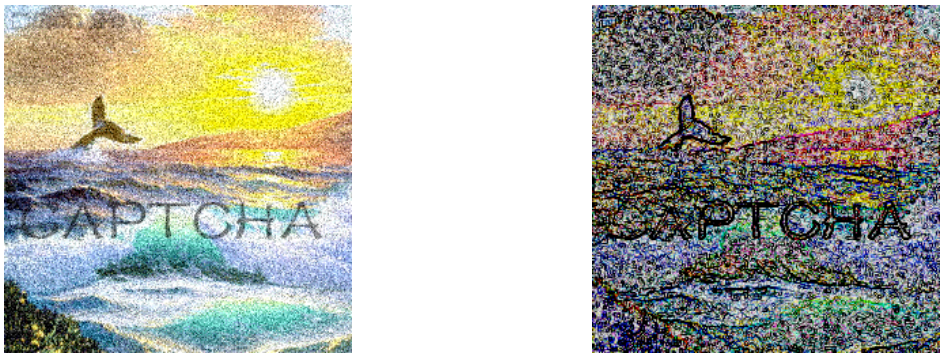
4. Challenge: What point in this picture corresponds to the given point in this other picture? Response: Point to the point. Given a picture and a slightly distorted variant of the picture, find points in the distorted picture that map from given points in the original picture. Or given two distinct views of a picture, find the point in one that corresponds to a given point in the other. The pictures could be the two images of a 3-D picture, or they could be two slightly different views of a face.

5.2.11 Our Notes on CAPTCHAs

Recall the suggestions in [98], besides the AI problems used in currently-known CAPTCHAs, many other problems are also available, including gender recognition, facial expression understanding, finding body parts, deciding nudity, naive drawing understanding, etc. Of course, as we have learned from Blum's Theorem on the impossibility of text-only CAPTCHAs, the real kernel of CAPTCHAs is how to transform the random

pre-challenges (images, sounds, texts, or other media) to distorted challenges, i.e., to disable Assumption 4 – Challenge Possibility.

Investigate all proposed image-based CAPTCHAs, the chief distortion algorithms are geometry deformation, noise adding and background complicating. In fact, there are still some other efficient distortion algorithms, such as image morphing, image mosaic, image fusion, etc. (actually, there exist so many different image filters to realize cool effects that are hard enough for computers to recover the original images). Also, we can combine many different image processing techniques to further enhance the performance of Gimpy. In Fig. 33, we give an example image processed by geometry distortion, image fusion and wind filter. From the edge image, we can see it is rather difficult to recognize "CAPTCHA" with current text recognition technology.



a) A processed image b) The edges of the left image

Figure 33: Images processed by geometry distortion, noise adding and image fusion

Following the above ideas, we can give several general models of image-processing based CAPTCHA:

1. Extract m images $\{I_1, \cdots, I_m\}(m \geq 2)$ from a public image database (or from web), process the m images to get a transformed image $I' = f(I_1, \cdots, I_m)$, where f is a compound IP (image processing) function containing one or more IP sub-functions. Transform W into an image I_W (with transparent background) and fuse this image with I' with a transparency factor α to get the final image I_C. In addition, an extra IP function f' can be used to process I'_W. The challenge is I_C, and the right response is W. Here, the involved IP sub-functions can be noise adding, geometry distortion, image fusion, etc. When using image fusion, please note that the color of I_W and the background where I_W is embedded should have enough color difference to make the embedded word legible to humans' eyes. The random mechanism and the parameters used in all IP filters makes it difficult for a bot to obtain the challenges. Such a model can be considered as a generalized description of Gimpy (in Gimpy, I_W is embedded into I' with $\alpha = 1$). An interesting implementation of this model is: using online map tools (or a virtual map generation software) to generate a map image with some place marks, embedded the word W into this map, it is generally hard to recognize W from this "chaotic" map image (there are many rivers, cities, national boundaries, coastlines and interferential place marks).

2. Given a randomly selected keyword W, extract m images $\{I_1, \cdots, I_m\}(m \geq 2)$ from a public image database (or from web), process the m images to get a transformed image $\{I'_1, \cdots, I'_m\}$, where $I'_i = f(I_i)$ and f is a compound IP function containing one or more IP sub-functions. Here, f can also involve another random image, for example, if f is image fusion, then I_i is fused with a random generated or selected images to generate I'_i. The challenge is $\{I'_1, \cdots, I'_m\}$, and the response is W. Generally speaking, f can be some simple IP functions, such as gray transformation plus resizing (or geometry transformation and noise adding). Such a model can be considered as a generalized description of PIX.

3. Construct an image database of n different objects, where all images should be typical so that most humans can easily get to know the corresponding objects related with the images. All images are labeled with the names of related objects. The challenge is an image I_C, which is generated as follows: randomly select an image I from the database and process it with a IP function to get $I' = f(I)$. The right response is the object name of I. To reduce the success probability of random guess, m images can be simultaneously displayed, then the guess probability will be $\frac{1}{n^m}$. When $n = 100, m = 6$, the probability is about 2^{-40} and enough for most applications. An example of this model is the "Who is this?" CAPTCHA mentioned in [77]. Of course, we can also use other object databases, such as database of foods, kitchenwares, etc.

Two more trivial notes: 1) In addition, to generate random images, Random Art [112] and Fractals Art can be helpful to provide more recourse of meaningless images with rich edges (which are rather useful to foil text recognition programs). 2) ASCII Art may be also useful to deign text-based CAPTCHAs. In fact, ASCII Art can be considered as special cases of image mosaic technique.

5.3 More Topics on Human Interactive Proofs (HIPs)

Essentially speaking, SecHCIs can be classified into a more general topic: Human Interactive Proofs (HIPs), which is a newly-developed research area lying between cryptography and human-computer interface [76]. SecHCI, CAPTCHA and research on visual/graphical passwords are all sub-topics of HIPs. Of course, there are some other sub-topics on HIPs. The most difficult problem of HIPs is the lack of strict models for human capabilities and interactions with computers. The interposition of ambiguous human behaviors makes the systematic design and theoretical analysis of a HIP system much more difficult than common cryptographic protocols. Fortunately, in recent years, some novel and interesting studies about HIPs have started. We believe more and more research will come in the future and some fruits may benefit the design and implementation of SecHCI protocols. Following this idea, in this sub-subsection, we would like to give a brief overview of some more topics on HIPs (besides SecHCIs, visual/graphical passwords and CAPTCHAs).

5.3.1 Formal Studies on Security and Complexity of HIPs

Apparently, it is meaningful to formally define the general model of HIPs, because it will help designers to systematically find security defects and strictly prove desired cryptographical properties. Some works have been done by N. Hopper and M. Blum in [84, 113] and by us in Sec. 3 of this paper. T. Berson also mentioned its significance in [114].

In [113], N. Hopper raised the problem of formally describing HIPs. Based on the previous formal definitions given in [84], he gave a partially formal definition of HIP:

Definition *A pair of interactive programs (P, V) with auxiliary inputs is an Interactive proof of Related Secrets for the relation R if:*

- *For all* $(x, y) \in R$, $Pr[\langle P(x), V(y)\rangle = \texttt{accept}] > 1 - \epsilon$.
- *For all* $(x, y) \notin R$, $Pr[\langle P(x), V(y)\rangle = \texttt{accept}] < \epsilon$.
- *For any* $\mathcal{P}$, *let* $p = Pr[\langle \mathcal{P}, V(y)\rangle = \texttt{accept}]$. *Then if* $p > \epsilon$, *there exists an* x *such that* $(x, y) \in R$ *and* $\mathcal{P}$ *and* $P(x)$ *are computationally indistinguishable.*

Based on the above definition, it is claimed that an Interactive Proof of Related Secrets is a Human Interactive Proof if the prover is (α, β, t)-human executable for some reasonable threshold of (α, β, t). If a HIP satisfies the requirement of (p, k)-security against peeping attacks, then it is a Secure HIP. After defining Secure HIPs, N. Hopper also gave the partially formal definitions of HumanOIDs (i.e., SecHCIs in this paper) and CAPTCHAs. Here, the definition of HIP is still not rigorous, because it is difficult to formally define "human executability". With the formal definition of HIPs, it become possible to study the complexity problems with HIPs. Such studies has not yet carried out (some complexity problems about CAPTCHAs was given in [113]).

5.3.2 Computer Vision and HIPs

It is possible to use some computer vision techniques to design HIPs, especially CAPTCHAs. Here, we give some useful ideas about this intersecting area. The first point is about biometrics, such as face recognition, which is used to realize new generation human-computer identification systems (see below for some details). The second point is using eye-tracking technique to design new passwords. For example, we can divide the whole screen into $m \times n$ parts, and let users to gaze different parts with pre-defined secret orders. Of course, eye-tracking can be also used as a novel input interface of normal passwords (both textual and graphical). Apparently, eye-tracking passwords are helpful to resist (at least passive) peeping attacks since it is generally difficult for adversaries to monitor eye movements of legal users with hidden cameras. Another point is about the use of shape recognition in the design and performance analysis of pattern recognition based CAPTCHAs (such as Gimpy), an example is shape matching technique based on shape contexts as a promising tool to break some CAPTCHAs [115, 116]. Are there more points?

5.3.3 Biometrics

Considering the involvement of humans' interactions in the biometrics login systems, biometrics can be considered as a topic about HIPs. As we know, there are two classes of biometrics: behavioral biometrics and physical/physiological biometrics. For physical/physiological biometrics, such as face recognition and fingerprint recognition, the humans interactions with computers are very simple and only used to provide inputs to machines. While behavioral biometrics require much more active human interactions, and such biometrics can be used to realize so-called *torture-robust* identification systems [117]. Torture-robust identification systems are such systems with which people cannot disclose the passwords even if under duress. How can behavioral biometrics do that? It is because human interactions are more of reflexes than reflections (i.e., it is difficult

for humans to intentionally impersonate others). As a natural result, Adrian Perrig and Dawn Song call such systems *reflex instead of reflection* in [117]. A well-known behavioral biometric is keystroke dynamics. A comprehensive report about keystroke dynamics can be found in [118] and a commercial product based on such keystroke dynamics has been in market [119].

5.3.4 Visual Cryptography

The concept of "visual cryptography" is firstly introduced by M. Naor and A. Shamir in Crypto'94 [120]. What is visual cryptography? It is a cryptosystem in which the encrypted information can be decrypted by the human visual system. The basic model consists of a print page of ciphertext (which can be sent by fax or mail) and a printed transparency (which serves as a secret key). The original plaintext is revealed by placing the transparency with the key over the page with the ciphertext, even though each one of them is indistinguishable from random noise. The above model can also be considered as a 2-out-of-2 secret sharing scheme [96], and can be extended to k-out-of-n visual secret sharing (VSS) schemes. For more details about visual cryptography, please see D. Stinson's introduction article [121]. Similar idea is also used by Yvo G. Desmedt et al. to propose a new information hiding scheme [122], in which humans can reconstruct a secret image hidden in two cover images (both images are not random) with the help of a 3-D viewer.

5.3.5 Human-Error-Tolerant Passwords (or Fuzzy Commitment)

As we know, people always forget something (even they are very important) and confuse different (even independent) things. It frequently occurs that we can only recall partial information about something. When you encounter such problems with your passwords, what will happen? The current human identification systems cannot help us to solve this problem. A widely-used remedy is the password recovery system on the Web: when you forget your passwords, you can request the system e-mail the password to your e-mail box (if you don't forget the password of the e-mail box), or you can answer some private questions (predefined by yourself) to convince the system your are the password-holder. However, sometimes you even cannot exactly answer the private questions you have set some months before by yourself (consider the answer to "Who am I?" should be "me!", but you only remember "me"). Some researchers have tried to solve this problem by introducing some fuzzy mechanisms into the current security architecture [123–126]. Although different terms are used, in our opinions, *human-error-tolerant passwords* [124] or *fuzzy commitment* [125] seems to be more appropriate. We can also consider such solutions as the passwords with the function of self error-correction. Some point-and-click graphical passwords [53, 54] are examples of human-error-tolerant passwords (you need not click the exact secret positions but click the approximate positions). Besides enhancing the current passwords, in [123], another interesting application of *fuzzy commitment* is so-called *movie lovers' problem*, which is about the following situation: Alice want to publicly publish her contact information to find some friends with ***similar*** tastes in movies.

5.3.6 Other Sides?

In the first workshop on HIPs, there are some other talks that may also provide some intriguing ideas in related topics. Unfortunately, because I have not found the documents of all talks presented at HIP 2002, I cannot judge whether or not these ideas exist and how valuable there ideas are. Here, we only list some ideas that may be useful.

- Moni Naor: Humans, Computers and Cryptography;
- Stuart Haber: HIPs and DRM/DPM systems;
- Thomas M.Breuel: Semantic Cryptography;
- Udi Manber: Web Exploits - What Bad People are Doing and How it Affects the Rest of Us.

5.4 Zero-Knowledge Based Identification Protocols

In the computer-computer security world, there are some practical zero-knowledge based identification schemes, such as the ones proposed in [7–10]. Although all current known ZK problems cannot be directly used to design SecHCI protocols, the structure of ZK identification protocols may bring new ideas to the design of SecHCIs. The general structure of ZK identification protocols is as follows (P=prover and V=verifier) [1]:

P$\longrightarrow$V: a public (random) witness
V$\longrightarrow$P: a (random) challenge
P$\longrightarrow$V: a response (dependent on the witness and the challenge)

In the above protocol, the witness is generated by P based on the secret knowledge. The witness is very important to realize ZK property. How to employ this structure to design a SecHCI protocol is another interesting topic in the future.

6 Our Opinions on Future Studies

6.1 A Comparison of Known SecHCI Protocols

Recall all proposed SecHCI protocols, only Hopper-Blum Protocol 2 can resist active peeping attack with the complexity of $O\left(\frac{n(n-1)}{2}\right)$ challenges, but this protocol has the least usability. It is the most crucial challenge how to design a SecHCI protocol with both acceptable security and usability.

Can we extend some SecHCI protocols to realize some new ones with the desired properties? Let us recheck all protocols one by one. 1) Matsumoto-Imai Protocol: In [58], C.-H. Wang et al. have tried to enhance Matsumoto-Imai Protocol with some remedies, but the usability is sacrificed much. It is not clear whether or not we can modify Matsumoto-Imai Protocol to reach the above target. 2) Matsumoto protocols: They can be considered as variants of Hopper-Blum Protocol 1 without adding noise into the responses. Essentially, it can only resist (both passive and active) peeping attacks with the complexity of $O(v)$ identifications. Thus, they are not suitable as candidates for more secure SecHCI protocols. 3) Hopper-Blum protocols: in [84], the authors pointed out that the structure of Hopper-Blum protocols has essential limitations and cannot be extended to realize more secure protocols. In addition, the security against active peeping attack is realized with the use of error-correcting function, which must reduce the usability. As a summary, to design a better SecHCI protocol, some new idea must be introduced.

Here, we qualitatively rank all proposed SecHCI protocols by different sides of the overall performance. We can see neither protocols can provide both acceptable usability and security against active peeping attack.

- By security against passive peeping attack: Matsumoto-Imai Protocol < Matsumoto Protocols < Hopper-Blum Protocol 2 < Hopper-Blum Protocol 1;
- By security against active peeping attack: Matsumoto-Imai Protocol < Matsumoto Protocols < Hopper-Blum Protocol 1 < Hopper-Blum Protocol 2;
- By usability: Hopper-Blum Protocol 2 < Matsumoto-Imai Protocol < (0,1)-version of Hopper-Blum Protocol 1 ≈ decimal version of Hopper-Blum Protocol 1 ≈ Matsumoto Protocols.

6.2 Our Opinions on Future Studies

In the future, the kernel in the design of SecHCI protocols should be the practical balance between security (against active peeping attack) and usability. In our opinion, two basic ideas will still be the keys to the final solution of SecHCI: ***intentional errors*** (used in Hopper-Blum protocols) and ***redundancies*** (used in Matsumoto-Imai protocol and Matsumoto protocol). Essentially speaking, intentional errors and redundancies have the same function on the peeping attacks: to force attackers to guess right information from wrong (or probabilistic wrong) information. However, generally errors and redundancies can only provide security against passive peeping attack.

To resist active peeping attack, some new countermeasures must be introduced. Hopper-Blum Protocol 2 uses error-correcting mechanism as a practical solution, but it is more of a passive solution than an active one. We think the key property to access security against peeping attack is ***balance*** (or uniformity), which means *all identification behaviors made by legal users (even including the movements of your eyes[22]) should be balanced (or uniform) from the viewpoint of observers who do not know the passwords.* Just because of the unbalance of the selection probabilities ($0 < \eta < \frac{1}{2} \Rightarrow 1 - \eta > \eta$), Hopper-Blum Protocol 1 is not secure to active peeping attack.

Furthermore, as we mentioned in the last section, we believe the research in many other related areas can provide more ideas to access the solution of a really satisfactory SecHCI, especially visual/graphical passwords, CAPTCHAs and fuzzy commitment.

Finally, as we have obtained from the user study in Sec. 2, a practical SecHCI must yield to the following two more requirements on usability: 1) the password length should not be larger than 16 except that some algorithms (which are simple enough for most people) are employed; 2) the identification time should not be too much (it is desired within 1 minute).

As a summary, to design a good SecHCI, we have three principles: ***intentional errors***, ***redundancies***, ***balance***, and two constrains: $k \leq 16$, HOE is better than $(0.1, 0.1, O(60))$.

[22] An human-computer identification protocol is unsuitable if it is designed to let (or force) users intentionally focus their attention on the secrets if they are displayed on the screen.

6.3 A Prototype Protocol

Following the three principles mentioned in the above subsection – ***intentional errors***, ***redundancies***, and ***balance***, it is possible to construct some SecHCI protocols and carefully check their security against (chiefly active) peeping attack. Here, we proposed a prototype protocol as an example of such SecHCI protocols, which is partially based on Hopper-Blum Protocol 1 and may be promising as a new solution of SecHCI.

> **Our Prototype SecHCI Protocol** *Given a (0,1)-vector* $\boldsymbol{x} \in \{0,1\}^n$ *as the secret password. The challenge is* $2m$ *(0,1)-vectors* $\boldsymbol{c}_1, \cdots, \boldsymbol{c}_{2m} \in \{0,1\}^n$*, and the correct response should be* $2m$ *bits* $r_1, \cdots, r_{2m}$ *that satisfy* $(r_{2i-1} - \boldsymbol{x} \cdot \boldsymbol{c}_{2i-1}) + (r_{2i} - \boldsymbol{x} \cdot \boldsymbol{c}_{2i}) = 1 \pmod 2$*, i.e.,* $r_{2i-1} - \boldsymbol{x} \cdot \boldsymbol{c}_{2i-1} = 0 \pmod 2 \wedge r_{2i} - \boldsymbol{x} \cdot \boldsymbol{c}_{2i} = 1 \pmod 2$ *or* $r_{2i-1} - \boldsymbol{x} \cdot \boldsymbol{c}_{2i-1} = 1 \pmod 2 \wedge r_{2i} - \boldsymbol{x} \cdot \boldsymbol{c}_{2i} = 0 \pmod 2$.

In the above SecHCI protocol, $(r_{2i-1} - \boldsymbol{x} \cdot \boldsymbol{c}_{2i-1}) + (r_{2i} - \boldsymbol{x} \cdot \boldsymbol{c}_{2i}) = 1 \pmod 2$ actually means that there is one wrong and one correct response in any two sequent responses r_{2i-1}, r_{2i}. The m intentional errors (random controlled by legal users) provide security against passive peeping attack, and the balance of wrong and correct r_i ($\frac{m}{2m} = \frac{1}{2}$) provides security against active peeping attack. For random response attack, the masquerading probability of this protocol is $\frac{1}{2^m}$. If attackers exhaustively guess the correct r_i, the complexity is $O(2^{n/2})$. To fulfill the requirements of a practical SecHCI protocol, $n = 160, m = 30$ may be OK. We can see the HOE is not very bad and may be enhanced to make it acceptable for most people.

This protocol have two problems: 1) To ensure the balance property, it is required that $Pr[\boldsymbol{x} \cdot \boldsymbol{c}_i = 0] = Pr[\boldsymbol{x} \cdot \boldsymbol{c}_i = 1] = \frac{1}{2}$, thus $\boldsymbol{c}_i$ should be generated uniformly at random. However, since n is general large enough, we must display n objects simultaneously on the screen, which cause the identification time too long. Thus, we should introduce some methods to solve this problem. 2) To ensure the balance property, it is required that legal users can make really uniform decision on how to response for each $\boldsymbol{c}_i$. However, we know human being is not an accurate machine and generally cannot make really balanced decision. If attackers guess such unbalance used by legal users, they can still carry out replay challenge attack to distinguish the correct r_i from the wrong ones, and then get the password by solving n independent linear equations. To avoid this defect, users should use some really random source (such as the time when you make the response) to determine their selections.

In the future research, we will give detailed analysis on the above prototype protocol and introduce some variants with better usability. We will also try to employ human vision capabilities (as the ones used in various visual/graphical passwords and CAPTCHAs) to simplify the usability of proposed SecHCIs. We hope several really practical and secure SecHCI protocols will be successfully designed.

7 Conclusion

In this paper, we give a comprehensive discussion on secure human-computer identification against peeping attacks. We call such an identification system SecHCI. The essence of SecHCI is the problem how a human can prove its identity to a trustworthy (local or remote) computer with untrustworthy input devices and via an insecure channel controlled by adversaries. Any input devices and auxiliary devices are untrustworthy under the following assumptions: the adversaries can record humans' operations on the devices, and can access the devices to replay the recorded operations. Strictly, only the common brain intelligence is available for the human.

Although peeping attacks is rather common and threaten our secrets everywhere (recall the discussions in Sec. 1.3 and the user study described in Sec. 2), the countermeasures to fight against peeping (i.e., SecHCI) are entirely fresh in today's security society and only a few contributions have devoted to its design and analysis. We survey the previous works on SecHCI in the last 10 years and point out that there is not yet a SecHCI with enough security against active peeping attack and acceptable usability. The most successful contributions are made by N. J. Hopper & M. Blum in [84]. They gave some formal definitions about SecHCI and proposed two protocols to provide enough security against passive and active peeping attack (but with impractical usability). By investigating all known SecHCI protocols, we also propose a prototype protocol as a new idea to the final solution of SecHCI. What's more, to enrich the ideas to design SecHCIs, we also introduce some related works, such as CAPTCHAs, HIPs and visual/graphical passwords.

References

[1] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography*. CRC Press Series on Discrete Mathematics and Its Applications. CRC Press, Inc., 1996. Available online at http://www.cacr.math.uwaterloo.ca/hac.

[2] Jeff Hayes. Smart cards, biometrics and tokens for VLANs and subnet access. Prensented at 17th Annual Computer Security Applications Conference (ACSAC 2001), 2001. Available online at http://www.acsac.org/2001/case/Thurs_C_1030_Hayes_Alcatel.pdf.

[3] Marc Boroditsky and Bruce Pleat. Security the edge: Making security and usability a reality with SSO. Available at http://www.passlogix.com/media/pdfs/security_at_the_edge.pdf, 2001.

[4] Leslie Lamport. Password authentication with insecure communication. *Communications of the ACM*, 24(11):770–772, 1981.

[5] Moni Naor and Benny Pinkas. Visual authentication and identification. In *Advances in Cryptology - CRYPTO'97*, Lecture Notes in Computer Science **1294**, pages 322–336. Springer-Verlag, Berlin, 1997.

[6] Ching-Nung Yang, Y. B. Yeh, and Chi-Sung Laih. A dynamic password visual authentication scheme through Internet. In *Proc. 16th Int. Telecommunication Symp. (ITS'98)*, pages 163–167, 1998.

[7] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *Advances in Cryptology - CRYPTO'86*, Lecture Notes in Computer Science **263**, pages 186–194. Springer-Verlag, Berlin, 1987.

[8] Uriel Feige, Amos Fiat, and Adi Shamir. Zero knowledge proofs of identity. In *Proc. the 9th annual ACM conference on Theory of computing (STOC'87)*, pages 210–217. ACM Press New York, 1987.

[9] Louis C. Guillou and Jean-Jacques Quisquater. A practical zero-knowledge protocol fitted to security microprocessor minimizing both transmission and memory. In *Advances in Cryptology - EUROCRYPT'88*, Lecture Notes in Computer Science **330**, pages 123–128. Springer-Verlag, Berlin, 1988.

[10] C.P. Schnorr. Efficient identification and signatures for smart cards. In *Advances in Cryptology - CRYPTO'89*, Lecture Notes in Computer Science **435**, pages 239–252. Springer-Verlag, Berlin, 1990.

[11] Peter J. Denning. Passwords. *American Scientist*, 80(2):117–120, 1992.

[12] Hans-Peter Königs. Cryptographic identification methods for smart cards in the process of standardization. *IEEE Communications Maganize*, 29(6):42–48, 1991.

[13] Aviel D. Rubin. Independent one-time passwords. *Computing Systems*, 9(1):15–27, 1996.

[14] Neil Haller. The S/Key™ one-time password system (also known as Internet RFC 1760). In *Proc. 1994 Symposium on Network and Distributed Systems Security (NDSS'94)*, pages 151–157. IEEE Computer Society, 1994.

[15] Liqun Chen and Chris J. Mitchell. Comments on the S/KEY user authentication scheme. *Operating Systems Review*, 30(4):12–16, 1996.

[16] Kazukumi Kobara and Hideki Imai. Limiting the visible space visual secret sharing schemes and their application to human identification. In *Advances in Cryptology - ASIACRYPT'96*, Lecture Notes in Computer Science **1163**, pages 185–195. Springer-Verlag, Berlin, 1996.

[17] The Biometric Consortium. Biometric consortium. please visist http://www.biometrics.org, 2002.

[18] William Rogers & Associates. Biometric information directory. Available online at http://www.biodigest.com/BiometricInformationDirectory/SneakPreview.pdf, 2001.

[19] UK Biometrics Working Group. Use of biometrics for identification and authentication: Advice on product selection. Issue 1.0, Available online at http://www.cesg.gov.uk/technology/biometrics/media/Biometrics%20Advice.pdf, 23 November 2001.

[20] UK Biometrics Working Group. Biometric device protection profile (BDPP). Draft Issue 0.82, Available at http://www.cesg.gov.uk/technology/biometrics/media/bdpp082.pdf, 5 September 2001.

[21] Tony Mansfield, Gavin Kelly, David Chandler, and Jan Kane. Biometric product testing final report. Issue 1.0, CESG/BWG Biometric Test Programme, Available online at http://www.cesg.gov.uk/technology/biometrics/media/Biometric%20Test%20Report%20pt1.pdf, 19 March 2001.

[22] Association for Biometrics (AfB) and International Computer Security Association (ICSA). 1999 glossary of biometric terms. Available online at http://www.afb.org.uk/downloads/glossuk2.pdf, 1999.

[23] Ton van der Putte and Jeroen Keuning. Biometrical fingerprint recognition: Don't get your fingers burned. In *Proc. IFIP TC8/WG8.8 4th Working Conference on Smart Card Research and Advanced Applications*, pages 289–303. Kluwer Academic Publishers, 2000. Also available online at http://cryptome.org/fake-prints.htm.

[24] Crypto-Gram. Biometrics: Truths and fictions. Available online at http://www.counterpane.com/crypto-gram-9808.html#biometrics, 1998.

[25] D. W. Davies and W. L. Price. *Security for Computer Networks: An Introduction to Data Security in Teleprocessing and Electronic Funds Transfer*. Wiley series in communication and distributed systems. John, Wiley & Sons, Inc., second edition, 1989.

[26] Steven King. Testing iris and face recognition in a personnel identification application. In *The Biometric Consortium Conference 2001*, February 13 - 15, 2002. Available online at http://www.itl.nist.gov/div895/isis/bc/bc2001/FINAL_BCFEB02/FINAL_1_Final%20Steve%20King.pdf.

[27] Jay Stanley and Barry Steinhardt. Drawing a blank: The failure of facial recognition technology in Tampa, Florida. An ACLU Special Report, Available online at http://www.aclu.org/issues/privacy/drawing_blank.pdf, Jan. 2002.

[28] The American Civil Liberties Union (ACLU). Airport security: Increased safety need not come at the expense of civil liberties. Available online at http://www.aclu.org/safeandfree/facts-airport.html, 2002.

[29] Tsutomu Matsumoto, Hiroyuki Matsumoto, Koji Yamada, and Satoshi Hoshino. Impact of artificial "gummy" fingers on fingerprint systems. In *Optical Security and Counterfeit Deterrence Techniques IV*, Proceedings of SPIE vol. 4677, pages 275–289. SPIE–The International Society for Optical Engineering, 2002. Also available online at http://cryptome.org/gummy.htm.

[30] John Leyden. Biometric sensors beaten senseless in tests. The Register, available online at http://www.theregister.co.uk/content/55/25400.html, 22 May 2002.

[31] The BioAPI Group. BioAPI Consortium. Please visit http://www.bioapi.org, 2002.

[32] Robert Morris and Ken Thompson. Password security: A case history. *Communications of the ACM*, 22(11):594–597, 1979.

[33] David C. Feldmeier and Philip R. Karn. UNIX password security - ten years later. In *Advances in Cryptology - CRYPTO'89*, Lecture Notes in Computer Science **435**, pages 44–63. Springer-Verlag, Berlin, 1990.

[34] Daniel V. Klein. "foiling the cracker": A survey of, and improvements to, password security. In *Proc. 2nd USENIX Security Workshop*, pages 5–14, 1990.

[35] Anne Adams and Martina Angela Sasse. Users are not the enemy. *Communications of the ACM*, 42(12):41–46, 1999.

[36] Sidney L. Smith. Authentication users by word association. *Computers & Security*, 6(6):464–470, 1987.

[37] Alma Whitten and J. D. Tygar. Usability of security: A case study. Technical Report CMU-CS-98-155, Carnegie Mellon University, Dec. 1998.

[38] Martina Angela Sasse, Sacha Brostoff, and Dirk Weirich. Transforming the 'weakest link' – a human/computer interaction approach to usable and effective security. *BT Technology J.*, 19(3):122–131, 2001.

[39] Anne Adams, Martina Angela Sasse, and Peter Lunt. Making passwords secure and usable. In H. Thimbleby, B. O'Conaill, and P. Thomas, editors, *People & Computers XII (Proceedings of HCI'97)*, pages 1–19. Springer, Berlin, 1997.

[40] Sacha Brostoff and Martina Angela Sasse. Safe and sound: A safety-critical approach to security (position paper). In *Proc. 2001 Workshop on New Security Paradigms*, pages 41–50. ACM, 2001.

[41] Dirk Weirich and Martina Angela Sasse. Pretty good persuasion: A first step towards effective password security in the real world. In *Proc. 2001 Workshop on New Security Paradigms*, pages 137–143. ACM, 2001.

[42] Marc Boroditsky. Passwords – security weaknesses & user limitations. Available at http://www.passlogix.com/media/pdfs/boroditsky.pdf, 1998.

[43] Bruce Schneier. *Secrets and Lies: Digital Security in a Networked World*. John Wiley & Sons, Inc., New York, 2000.

[44] James A. Haskett. Pass-algorithms: A user validation scheme based on knowledge of secret algorithm. *Communications of the ACM*, 27(8):777–781, 1984.

[45] Maria M. King. Rebus passwords. In *Proc. the 7th Annual Computer Security Applications Conference*, pages 239–243. IEEE Press, 1991.

[46] Ian Jermyn, Alain Mayer, Fabian Monrose, Michael K. Reiter, and Aviel D. Rubin. The design and analysis of graphical passwords. In *Proc. 8th USENIX Security Symposium*, pages 1–14, 1999. Available at http://www.usenix.org/publications/library/proceedings/sec99/jermyn.html.

[47] ID Arts Inc. Passfaces - the art of identification. Please visit http://www.idarts.com, 2002.

[48] Real User Corporation. Real User Corporate profile. Available at http://www.realuser.com/published/RealUserCorporateProfile.pdf, April, 2002.

[49] Real User Corporation. The science behind Passfaces. Available at http://www.realuser.com/published/ScienceBehindPassfaces.pdf, Sep., 2001.

[50] Real User Corporation. PKI and Passfaces^{TM}: Synergistic or competitive? Available at http://www.realuser.com/published/PassfacesAndPKI.pdf, Oct., 2001.

[51] Sacha Brostoff and M. Angela Sasse. Are Passfaces more usable than passwords? a field trial investigation. In S. McDonald, Y. Waern, and G. Cockton, editors, *People and Computers XIV - Usability or Else (Proceedings of HCI 2000)*, pages 405–424, Sunderland, UK, September 5th - 8th 2000. Springer, Berlin.

[52] Rachna Dhamija and Adrian Perrig. Déjà Vu: A user study using images for authentication. In *Proc. 9th USENIX Security Symposium*, pages 45–58, 2000. Available at http://www.usenix.org/events/sec2000/dhamija.html.

[53] Vince Sorensen. PassPic - Visual password management. Please visist http://www.authord.com/PassPic, 2002.

[54] Passlogix Inc. Welcome to passlogix. Please visit http://www.passlogix.com, 2002.

[55] Marc Boroditsky. v-GO^{TM}: Usable security technology. Available at http://www.passlogix.com/media/pdfs/usable_security.pdf, 2000.

[56] David Bensinger. Human memory and the graphical password. Available at http://www.passlogix.com/media/pdfs/bensinger.pdf, 1998.

[57] Tsutomu Matsumoto and Hideki Imai. Human identification through insecure channel. In *Advances in Cryptology - EUROCRYPT'91*, Lecture Notes in Computer Science **547**, pages 409–421. Springer-Verlag, Berlin, 1991.

[58] Chih-Hung Wang, Tzonelih Hwang, and Jiun-Jang Tsai. On the Matsumoto and Imai's human identification scheme. In *Advances in Cryptology - EUROCRYPT'95*, Lecture Notes in Computer Science **921**, pages 382–392. Springer-Verlag, Berlin, 1995.

[59] Ross J. Anderson. Why cryptosystems fail. *Communications of the ACM*, 37(11):32–40, 1994.

[60] Ross J. Anderson. Why cryptosystems fail. In *Proc. 1st ACM Conf. Computer and Communication Security (CCS'93)*, pages 215–227, 1993.

[61] searchSecurity.com. Shoulder surfing. Available online at http://searchsecurity.techtarget.com/sDefinition/0,,sid14_gci802244,00.html, Feb 14, 2002.

[62] BBC News. Chinese cameras spying on spouses. Available at http://news.bbc.co.uk/hi/english/world/asia-pacific/newsid_1885000/1885218.stm, 21 March, 2002.

[63] Mark Mitechell. Always on the lookout. *TIME Asia Maganize*, 159(12):15–27, April, 2002. Available at http://www.time.com/time/asia/magazine/article/0,13673,501020401-219895,00.html.

[64] Cassi Goodman. An introduction to TEMPEST. Available online at http://rr.sans.org/encryption/TEMPEST.php, April 18, 2001.

[65] James M. Atkinson. TEMPEST 101 - debunking the myth. Available online at http://www.tscm.com/TSCM101tempest.html, 2002.

[66] Wim van Eck. Electromagnetic radiation from video display units: An eavesdropping risk? *Computers & Security*, 4(4):269–286, 1985.

[67] Joe Loughry and David A. Umphress. Information leakage from optical emanations. *ACM Trans. Information and System Security,*, 5(3):262–289, 2002.

[68] Markus G. Kuhn. Optical time-domain eavesdropping risks of CRT displays. In *Proc. 2002 IEEE Sym. Security and Privacy (S&P'02)*, pages 1–16. IEEE Computer Society, 2002.

[69] Markus G. Kuhn and Ross J. Anderson. Soft tempest: Hidden data transmission using electromagnetic emanations. In *Information Hiding 1998*, Lecture Notes in Computer Science **1525**, pages 124–142. Springer-Verlag Berlin, 1998.

[70] Fort George G. Meade. TEMPEST fundamentals. Published by NSA (National Security Agency) as NACSIM 5000, available at http://cryptome.org/nacsim-5000.htm (redacted by John Young), Feb. 1982.

[71] The Office of the Law Revision Counsel of the U.S. House of Representatives. Wire and electronic communications interception and interception of oral communications. United States Code, Title 18, Part I, Chapter 119, available online at http://uscode.house.gov/DOWNLOAD/18C119.DOC, 2002.

[72] Christopher J. Seline. Eavesdropping on the electromagnetic emanations of digital equipment: The laws of Canada, England and the United States. Prepublication draft, a HTML version is available online at http://www.parallaxresearch.com/dataclips/pub/infosec/emsec_tempest/law/tempest-law.htm, 1990.

[73] David O. Tyson. Emissions from bank computer systems make eavesdropping easy, expert says. *American Banker*, page 1, March 26, 1985.

[74] Barry Fox. New-wave spies: Electronic eavesdropping is becoming mere child's play. *New Scientist*, 164(2211):11, 06 November 1999.

[75] Frank Jones. Nowhere to run... nowhere to hide...: The vulnerability of CRT's, CPU's and peripherals to TEMPEST monitoring in the real world. Available online at http://www.parallaxresearch.com/dataclips/pub/infosec/emsec_tempest/info/TEMPESTMonitor.txt, 1996.

[76] Aladdin Center of Carnegie Mellon University. Human interactive proofs (HIPs). Available at http://www.aladdin.cs.cmu.edu/hips, 2002.

[77] Manuel Blum and Nick Hopper. Cs 827: Security and cryptography. Please visit http://www-2.cs.cmu.edu/%7Ehopper/cs827-f01, Fall September, 2001.

[78] Aladdin Center of Carnegie Mellon University. The CMU HumanAut project. Available at http://www.captcha.net/humanaut, 2002.

[79] National Security Agency. National Security Agency specification for shielded enclosures. Specification NSA No. 94-106, available at http://cryptome.org/nsa-94-106.htm, 24 Oct. 1994.

[80] Thomas Wu. The secure remote password protocol. In *Proc. 1998 Network and Distributed System Security (NDSS'98) Symposium*, pages 97–111. Internet Society (ISOC), 1998. Also available online at http://www.isoc.org/isoc/conferences/ndss/98/wu.pdf.

[81] Tsutomu Matsumoto. Cryptographic human identification. In *Analysis, Design and Evaluation in Human-Computer Interaction*, volume III of *Proc. 6th Int. Conf. on Human-Computer Interaction (HCI International'95)*, pages 147–152, 1995.

[82] Tsutomu Matsumoto. Human-computer cryptography: An attempt. In *Proc. ACM Conf. on Computer and Communication Security*, pages 68–75. ACM Press, 1996.

[83] Nicholas J. Hopper and Manuel Blum. A secure human-computer authentication scheme. Technical Report of Carnegie Mellon University CMU-CS-00-139, Available online at http://reports-archive.adm.cs.cmu.edu/anon/2000/abstracts/00-139.html, May, 2000.

[84] Nicholas J. Hopper and Manuel Blum. Secure human identification protocols. In *Advances in Cryptology - ASIACRYPT 2001*, Lecture Notes in Computer Science **2248**, pages 52–66. Springer-Verlag, Berlin, 2001.

[85] N. J. Hopper. Answer e-mails to "some questions about your paper on AsiaCrypt 2001". Private Communications between N. J. Hopper and Shujun Li, Aug. 2002.

[86] G. Miller. The magic number seven plus or minus two: Some limits on your capacity for processing information. *Psychological Review*, 63(1):81–96, 1956.

[87] Rachel Rue. Eighty-six bits of memory magic. Presented at the First Workshop on Human Interactive Proofs (HIP), abstract available at http://www.aladdin.cs.cmu.edu/hips/events/abs/rue_abstract.pdf, 2002.

[88] The Open Group. Security Forum - Single Sign-On. Please visit http://www.opengroup.org/security/l2-sso.htm, 2002.

[89] Keys Botzum. Single Sign On - A Contrarian View. Document on IBM Software Services for WebSphere, available at http://www7b.boulder.ibm.com/wsdd/library/techarticles/0108_botzum/botzum.html, August 2001.

[90] Microsoft Corporation. Microsoft® .NET passport. Please visit http://www.passport.com, 2002.

[91] CERT® Coordination Center (CERT/CC). Denial of service attacks. Software Engineering Institute, Carnegie Mellon University, Available online at http://www.cert.org/tech_tips/denial_of_service.html, Initial release on Oct 02, 1997, converted to new web format on Feb 12, 1999, and updated links on June 4, 2001.

[92] Jeanette P. Schmidt, Alan Siegel, and Aravind Srinivasan. Chernoff-Hoeffding bounds for applications with limited independence. *SIAM J. Discrete Mathematics*, 8(2):223–250, 1995.

[93] Aladdin Center of Carnegie Mellon University. Web site of Aladdin center. Please visit http://www.aladdin.cs.cmu.edu, 2001.

[94] Lius von Ahn, Manuel Blum, and John Langford. Telling humans and computers apart (automatically) or How lazy cryptographers do AI. Technical Report CMU-CS-02-117, Carnegie Mellon University, Feb. 2002.

[95] Aladdin Center of Carnegie Mellon University. The CAPTCHA project. Available at http://www.captcha.net, 2002.

[96] Bruce Schneier. *Applied Cryptography – Protocols, algorithms, and souce code in C.* John Wiley & Sons, Inc., New York, second edition, 1996.

[97] A. L. Coates, H. S. Baird, and R. J. Fateman. Pessimal print: A reverse turing test. In *Proc. the 6th Int. Conf. on Document Analysis and Recognition (ICDAR 2001)*, pages 1154–1158. IEEE Computer Society, 2001.

[98] Moni Naor. Verification of a human in the loop or Identification via the turing test. Unpublished, available at http://www.wisdom.weizmann.ac.il/%7Enaor/PAPERS/human_abs.html, Sep. 1996.

[99] Mark D. Lillibridge, Martin Abadi, Krishna Bharat, and Andrei Z. Broder. Method for selectively restricting access to computer systems. US Patent No. 6195698, Feb. 27, 2001.

[100] The CAPTCHA Project. Gimpy. Available at http://www.captcha.net/captchas/gimpy, 2002.

[101] Andrei Broder. Preventing bulk URL submission by robots in AltaVista. Presented at the First Workshop on Human Interactive Proofs (HIP), abstract available at http://www.aladdin.cs.cmu.edu/hips/events/abs/andrei_abstract.pdf, 2002.

[102] Henri Baird. The ability gap between human & machine reading systems. Presented at the First Workshop on Human Interactive Proofs (HIP), abstract available at http://www.aladdin.cs.cmu.edu/hips/events/abs/baird_abstract.pdf, 2002.

[103] The CAPTCHA Project. Bongo. Available at http://www.captcha.net/captchas/bongo, 2002.

[104] The CAPTCHA Project. Pix. Available at http://www.captcha.net/captchas/pix, 2002.

[105] The CAPTCHA Project. Sounds. Available at http://www.captcha.net/captchas/sounds, 2002.

[106] The CAPTCHA Project. Sound oriented CAPTCHA: Byan. Available at http://drive.to/research, 2002.

[107] Nancy Chan. Sound oriented CAPTCHA. Presented at the First Workshop on Human Interactive Proofs (HIP), abstract available at http://www.aladdin.cs.cmu.edu/hips/events/abs/nancy_abstract.pdf, 2002.

[108] Daniel Lopresti, Chilin Shih, and Gregory Kochanski. Human interactive proofs for spoken language interface (extended abstract). Presented at the First Workshop on Human Interactive Proofs (HIP), abstract available at http://www.aladdin.cs.cmu.edu/hips/events/abs/dlopresti_abstract.pdf, 2002.

[109] Philip Brighten Godfrey. Text-based CAPTCHA algorithms. Presented at the First Workshop on Human Interactive Proofs (HIP), abstract available at http://www.aladdin.cs.cmu.edu/hips/events/abs/godfreyb_abstract.pdf, 2002.

[110] Bartosz Przydatek. On the (im)possibility of a text-only CAPTCHA. Presented at the First Workshop on Human Interactive Proofs (HIP), abstract available at http://www.aladdin.cs.cmu.edu/hips/events/abs/bartosz_abstract.pdf, 2002.

[111] Ke Yang. Issues for Chinesse CAPTCHA. Presented at the First Workshop on Human Interactive Proofs (HIP), abstract available at http://www.aladdin.cs.cmu.edu/hips/events/abs/yangke_abstract.pdf, 2002.

[112] Andrej Bauer. The gallery of random art. Please visit http://gs2.sp.cs.cmu.edu/art/random, 2002.

[113] Nick Hopper. Security and complexity aspects of human interactive proofs. Presented at the First Workshop on Human Interactive Proofs (HIP), abstract available at http://www.aladdin.cs.cmu.edu/hips/events/abs/hopper_abstract.pdf, 2002.

[114] Tom Berson. Preliminary ideas about human interactive proofs. Presented at the First Workshop on Human Interactive Proofs (HIP), abstract available at http://www.aladdin.cs.cmu.edu/hips/events/abs/berson_abstract.pdf, 2002.

[115] Jitendra Malik. Visual shape recognition and CAPTCHAs. Presented at the First Workshop on Human Interactive Proofs (HIP), abstract available at http://www.aladdin.cs.cmu.edu/hips/events/abs/malik_abstract.pdf, 2002.

[116] Serge Belongie, Jitendra Malik, and Jan Puzicha. Shape matching and object recognition using shape contexts. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 24(4):509–522, 2002.

[117] Adrian Perrig and Dawn Song. New directions for user authentication: Reflex instead of reflection. Presented at the First Workshop on Human Interactive Proofs (HIP), abstract available at http://www.aladdin.cs.cmu.edu/hips/events/abs/perrig_abstract.pdf, 2002.

[118] Fabian Monrose, Michael K. Reiter, and SusanneWetzel. Password hardening based on keystroke dynamics. *Int. J. Information Security*, 1(1):1–15, 2001.

[119] Net Nanny Software Inc. BioPassword.com home page. Please visit http://www.biopassword.com, 2002.

[120] Moni Naor and Adi Shamir. Visual cryptography. In *Advances in Cryptology - EUROCRYPT'94*, Lecture Notes in Computer Science **950**, pages 1–12. Springer-Verlag, Berlin, 1995.

[121] Doug Stinson. Visual cryptography & threshold schemes. *Dr. Dobb's J. Software Tools for Professional Programmer*, 23(4):36, 38–43, April 1998.

[122] Yvo G. Desmedt, Shuang Hou, and Jean-Jacques Quisquater. Cerebral cryptography. In *Information Hiding 1998*, Lecture Notes in Computer Science **1525**, pages 62–72. Springer-Verlag, Berlin, 1998.

[123] Ari Jueles. At the juncture of cryptography and humanity. Presented at the First Workshop on Human Interactive Proofs (HIP), abstract available at http://www.aladdin.cs.cmu.edu/hips/events/abs/juels_abstract.pdf, 2002.

[124] Ari Juels and Martin Wattenberg. Error-tolerant password recovery. In *Proc. 8th ACM Conf. Computer and Communications Security (CCS'01)*, pages 1–9, 2001.

[125] Ari Juels and Martin Wattenberg. A fuzzy commitment scheme. In *Proc. 6th ACM Conf. Computer and Communications Security (CCS'99)*, pages 28–36, 1999.

[126] Carl Ellison, Chris Hall, Randy Milbert, and Bruce Schneier. Protecting secret keys with personal entropy. *Future Generation Computer Systems*, 16(4):311–318, 2000.