On Limitations of Designing Leakage-Resilient Password Systems: Attacks, Principles and Usability

Qiang Yan, Jin Han, Yingjiu Li, Robert H. Deng
School of Information Systems, Singapore Management University
{qiang.yan.2008, jin.han.2007, yjli, robertdeng}@smu.edu.sg

Abstract

The design of leakage-resilient password systems (LRPSes) in the absence of trusted devices remains a challenging problem today despite two decades of intensive research in the security community. In this paper, we investigate the inherent tradeoff between security and usability in designing LRPS. First, we demonstrate that most of the existing LRPS systems are subject to two types of generic attacks - brute force and statistical attacks, whose power has been underestimated in the literature. Second, in order to defend against these two generic attacks, we introduce five design principles that are necessary to achieve leakage resilience in the absence of trusted devices. We also show that these attacks cannot be effectively mitigated without significantly sacrificing the usability of LRPS systems. Third, to better understand the tradeoff between security and usability of LRPS, we propose for the first time a quantitative analysis framework on usability costs of password systems. By decomposing the authentication process of existing LRPS systems into atomic cognitive operations in psychology, we show that a secure LRPS in practical settings always imposes a considerable amount of cognitive workload on its users, which indicates the inherent limitations of such systems and in turn implies that an LRPS has to incorporate certain trusted devices in order to be both secure and usable.

1 Introduction

Password has been the most pervasive means for user authentication since the advent of computers. Compared to its alternatives, such as biometrics and smartcard which are cumbersome to use and require the existence of an underlying infrastructure, password is much easier and cheaper to create, update, and revoke. However, the use of password has intrinsic problems. Among them, secret leakage is one of the most common security threats [21], in which an adversary steals the password by capturing (e.g. by shoulder-surfing or key logging) and analyzing a user's inputs during an authentication session. Traditional password systems ask a user to directly input his entire plaintext password recalled from the user's memory so that an observation of a single authentication session is sufficient to capture the password. In order to prevent secret leakage during password entry, a user needs to input the password indirectly, which imposes an extra burden on the user. How to design a password system that *minimizes secret leakage and is still easy to use* is the fundamental problem in the design of leakage resilient password systems (LRPSes).

An ideal LRPS allows a user to generate a *one-time password* (OTP) for each authentication session based on an easy-to-remember secret. This can be easily achieved when a secure channel is available between user and authentication service. The secure channel blinds the adversary by decoupling a user input from the underlying secret, when the message delivered over the secure channel is not revealed to the adversary. However, the prerequisite of a secure channel may be infeasible or introduces other vulnerabilities in practical settings. For example, when the secure channel is formed by a trusted device such as secure token or mobile phone, that device is subject to theft or loss. This motivates the existing research on usable and secure LRPS systems with only the support of human cognitive capabilities [22, 15, 26, 20, 31, 32, 35, 4, 27, 2]. A few representative systems include Convex Hull Click (CHC) [32], Cognitive Authentication Scheme (CAS) [31], and Predicate-based Authentication Service (PAS) [4].

The difficulty in designing an LRPS system stems from the *capability asymmetry* between user and strong adversary. A strong adversary may use a hidden camera or malicious software to record complete interactions between user and his computer and then analyze the data with powerful machines. Many LRPS systems [15, 20, 31, 32, 35, 4, 27, 2] have been proposed to defend against this type of secret-leakage attacks. However, as we will demonstrate later in the paper, all the existing proposals with acceptable usability are vulnerable to either or both types of generic attacks: brute force attack and statistical attack.

Brute force attack is a pruning process for the entire candidate password set, whose strength has often being underestimated in prior research. Our experiments show that brute force attack is able to recover the secrets of certain existing LRPS systems from a small number of observations of authentication sessions. Statistical attack, on the other hand, represents a learning process to extract a user's secret due to statistical significance of the secret. We introduce two types of statistical attack, probabilistic decision tree and multi-dimensional counting. Rigorous experiments are conducted to show the effectiveness of these two attacks in breaking existing schemes.

We note that these two generic attacks are different from other specific attacks that have been systematically studied in the literature, including SAT [13] and Gaussian elimination [19]. SAT attacks can be efficiently prevented by asking a user to select only one of the correct responses while multiple correct responses can be derived from each challenge, since this would increase the size of the SAT expression exponentially with the number of observations. On the other hand, Gaussian elimination-based algebraic attacks can be efficiently prevented by using a non-linear response function [20] or introducing noises from user's intentional mistakes [15]. Unlike these specific attacks, brute force and statistical attacks cannot be easily defended without significantly sacrificing the system's usability, which implies inherent limitations of LRPS without using trusted devices. In order to defend against these attacks, we introduced five design principles which should be followed to achieve leakage resilience. Using counterexamples, we show that an LRPS system can be easily broken when these principles are violated.

To further understand the tradeoff between security and usability in the design of LRPS systems, we propose for the first time a quantitative analysis framework on usability costs of LRPS systems. This framework decomposes the process of human-computer authentication into atomic cognitive operations. Performance data of average human-beings reported in psychology literatures [28, 12, 9, 30, 10, 23, 25, 7, 33, 34, 16, 6, 14] are used to estimate usability costs of existing LRPS systems [15, 20, 31, 32, 35, 4, 27, 2]. Our analysis results are consistent with the experimental results reported in the original literatures, while the hidden costs previously not addressed are identified. Our results show that a secure LRPS in practical settings [15, 2] always leads to a considerable amount of cognitive workload, which explains why some of the existing LRPS systems require extremely long authentication time and have high authentication error rate. This limitation has not been, and will not be easily solved in the design of LRPS in the absence of trusted device.

In a nutshell, the contributions of this paper are three-fold:

- We analyze and demonstrate the effectiveness of two types of generic attacks, brute force and statistical attacks, against LRPS systems. We propose two statistical attack techniques, probabilistic decision tree and multi-dimensional counting, and show their effectiveness against existing schemes.
- We introduce five principles that are necessary to mitigate brute force and statistical attacks. We use typical existing LRPS proposals as counterexamples to show that an adversary can easily obtain user secrets in the schemes violating our principles.
- We establish the first quantitative analysis framework on usability costs of the existing LRPS systems. This framework utilizes the performance models of atomic cognitive operations in authentication to estimate usability costs. Our analysis result shows that there is a strong tradeoff between security and usability in the existing LRPS systems. It implies that an unaided human may not be competent enough to effectively use a secure LRPS system in practical settings; in other words, it is inevitable to incorporate certain trusted device in LRPS design.

2 Definitions and Threat Model

In this section, we introduce related notions and our threat model. We focus on the fundamental problem of designing LRPSes for unaided humans, i.e. *when a secure channel or trusted device is unavailable*. We exclude LRPSes using secure channel or trusted device in our discussion unless explicitly mentioned.

2.1 Leakage-Resilient Password System

An LRPS is essentially a challenge-response protocol between human and computer. We refer to human as *user*, and computer as *server*. During registration, a user and a server agree on a *root secret*, usually referred to as a password. The user later uses the root secret to generate *responses* to *challenges* issued by the server to prove his identity. Unlike traditional password systems, a response in LRPS is an obfuscated message derived from the root secret, rather than the plaintext of the root secret itself. Considering the limited cognitive capabilities of unaided humans, a usable obfuscation function F is usually a many-to-one mapping from a large candidate set to a small answer set. The small size of the answer set increases the success rate of *guessing attack* where an adversary attempts to pass the authentication by randomly picking an answer from the answer set. For this reason, an *authentication session* of LRPS often requires executing multiple rounds of the challenge-response procedure in order to reach an expected authentication strength D (specifically, the resistance against random guessing, e.g. $D = 10^{-6}$ for 6-digit PIN), where each round is referred to as an *authentication round*. We use d to denote the average success rate of guessing attack per authentication round. Given d and D, the minimum number m of authentication rounds for an authentication session is $\lceil \log_d D \rceil$.

To imbue the server with a high flexibility in challenge generation, the *k-out-of-n paradigm* [15] has been adopted for secret agreement in most existing LRPS systems [15, 20, 31, 32, 35, 27, 2]. In this paradigm, the root secret consists of k independent elements randomly drawn from a pool of n elements. An element can be an image, a text character, or any symbol in a notational scheme. The set of k secret elements is called the *secret set* (and forms the root secret of the user), and the complementary set is called the *decoy set*. The server knows the secret set chosen by the user, and uses a subset or all of these k elements to generate the challenge in each round. We refer to the chosen portion of the root secret for an authentication round as a *round secret*.

Based on the above notions, the common system parameters of the most existing LRPS systems [15, 20, 31, 32, 35, 4, 27, 2] can be described by a tuple (D, k, n, d, w, s), where D is the expected authentication strength of an authentication session, k is the number of secret elements drawn from an alphabet of n candidate elements, d is the average success rate of guessing attack in a single round, w is the average window size which is the number of elements appearing on the screen for an authentication round, and s is the average length of user's decision path which is the number of decisions that a user has to make before producing the correct response for an authentication round. The total round number m can be derived from D and d. The parameters m, w, and s are required for usability evaluation. More details will be given in Sections 5 and 6.

2.2 Threat Model and Experimental Setting

There are two types of passive adversary models for secret leakage attacks used in prior research. The weaker passive adversary model (e.g. *cognitive shoulder-surfing* [26]) assumes that the adversary is not able to capture the complete interaction between a user and the server [26]. Such an assumption actually forms a secure channel between user and server, which transforms the secret leakage problem to the protection of the secure channel. However, this assumption may not hold for a prepared adversary who deploys a *hidden camera, key logger, or phishing web site* to capture the whole password entry process. To address such realistic concerns, recent efforts [20, 31, 32, 35, 4, 27, 2] have focused on the strong passive adversary model, where the adversary is allowed to record the complete interaction between the user and the server.

In the strong passive adversary model, secret leakage during human-computer authentication is unavoidable. The user's response is based on his knowledge of the secret, which distinguishes it from a random choice as required for the authentication purpose. This difference leaks information about the secret. After recording a sufficient number of authentication rounds, the adversary may use any reasonable computation resources to analyze and recover the underlying secret. The research problem under such a threat model is to lower the secret leakage rate while maintaining acceptable usability for unaided humans.

In this paper, we consider both brute force attack and statistical attack under this strong passive adversary model. The security strength of an LRPS is defined as *the resistance against these two generic attacks given the same success rate of random guessing* (i.e. the same authentication strength for a legitimate user). We will use simulation experiments to evaluate the security strength of existing schemes, whose process is summarized as follow: 1) Generate a random password as the root secret; 2) Generate a challenge for an authentication round; 3) Generate a response based on the password and the underlying system design; 4) Analyze the collected challenge-response pairs after each authentication round assuming that the adversary has full knowledge of the system design except the password; 5) Repeat steps 2, 3, and 4 until the exact password is recovered. The final findings shown in the following sections are the average results of 20 runs for each system.

3 Brute Force Attack and Its Defense Principles

3.1 Attack Strategy

Brute force attack is a general pruning-based learning process, where the adversary keeps removing irrelevant candidates when more and more cues are available. Its procedure can be described as follows: 1) List all possible candidates for the password in the target system; 2) For each independent observation of a challenge-response round, check the validity of each candidate in the current candidate set by running the verification algorithm used by the server, and remove invalid candidates from the candidate set; 3) Repeat the above step until the size of candidate set reaches a small threshold.

The above procedure shows that the efficiency of brute force attack in the leakage resilience setting is *design-independent*, and is only limited by the size of the candidate set. We introduce two statements to further describe the power of brute force attack. These statements apply not only to root secret, but also to round secrets when the adversary is able to reliably group the observations for individual round secret.

Statement 1: *The verification algorithm used in brute force attack for candidate verification is at least as efficient as the verification algorithm used by server for response verification.*

The proof is trivial as the verification process for candidate pruning is essentially the same as the verification process for the server to check correct response. It is also possible for the adversary to design a more efficient algorithm if there are correlations between candidates.

Statement 2: *The average shrinking rate for the size of valid candidate set is the same as one minus the average success rate of guessing attack.*

The average success rate of guessing attack is defined as the probability of generating correct response by randomly picking a candidate from the candidate set. This is an equivalent definition of average shrinking rate of the valid candidate set. Given X as the size of the candidate set, and d as the average success rate of guessing attack, the average number of rounds to recover the exact secret is $m = \lceil \log_{1/d} X \rceil$, assuming that each candidate is independent of each other. If each candidate is not independent, the average number of rounds to recover the exact secret will be smaller than m. This statement can be used to estimate the average success rate of guessing attack, $d = X^{-\frac{1}{m}}$, when the precise analysis is difficult to perform (see later examples). The statement also explains why most password systems [26] reveal the entire secret after one or two authentication sessions recorded by the adversary, as their expected success rates of guessing attack are sufficiently low so that the whole candidate set rapidly collapse to the exact secret. This implies that, when brute force attack is feasible, enhancing strength against guessing attack is strictly at the cost of sacrificing leakage resilience.

3.2 P1: Large Root Secret Space Principle

Principle 1: *An LRPS system with secret leakage should have a large candidate set for the* **root secret**.

The first principle requires a large password space as the basic defense against brute force attack, where large means that it is computational infeasible for the adversary to enumerate all candidates in a practical setting (the same meaning of *large* will be used in the following discussion). This principle seems trivial but actually not, as the necessity of involving a large password space depends on whether an LRPS system has secret leakage under a given threat model, which is not straightforward to decide. In general, there are three possible leakage sources in an LRPS system: *the response alone*, *the challenge-response pair*, and *the challenge alone*. Among them, the last source has not been well recognized. We use Undercover [27] as a counterexample to show that secret leakage could happen even when a secure channel is present.

Undercover is a typical scheme based on the k-out-of-n paradigm. During registration, a user is assigned k images as his secret from a pool of n images. In each authentication round, the user is asked to recognize if there is a secret image from w candidate images and report the position of that image if the secret image is shown in the current window; otherwise the user reports the position of the "none" symbol. Before the user reports the position, a haptics-based secure channel is deployed to map the real position to a random position decided by the hidden message delivered via the secure channel.

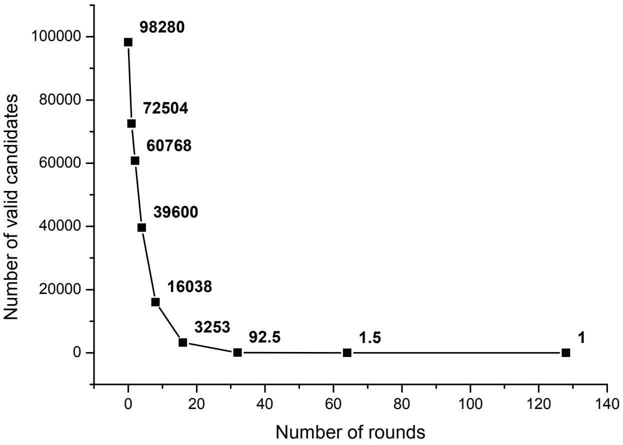

Figure 1. The average number of valid candidates shrinks for Undercover.

The hidden mapping blinds the adversary from learning any information from the response. The authors suggested a small password space is sufficient so that the default parameters are $k = 5$, $n = 28$, and $w = 4{+}1$ (i.e. four images and a "none" symbol). The number of candidate root secrets is $C_{28}^{5} = 98280$. However, this scheme does not prevent the challenge alone from becoming a source of leakage. *In Undercover, there is at most one secret image among the w candidate images for each authentication round. This implies a candidate of the root secret is invalid if two images in this candidate appeared in an authentication round.* Since it has a small candidate space, we can use brute force to recover the secret with the information from the challenge alone. Figure 1 shows how the size of the candidates shrinks as the number of observed authentication rounds increases. On average, 53.06 rounds (6 sessions) are sufficient to recover the exact secret, and the size of the candidate set can be reduced to less than 10 after 43.55 rounds (5 sessions). This result shows that *a secure channel alone is not sufficient to prevent secret leakage.*

The same problem also appears in the Convex Hull Click (CHC) scheme [32], where the default parameters are $k = 5$, $n = 112$, $w = 83$. The size of the candidate set for its root secret is $C_{112}^{5} = 1.34 \times 10^{8}$. In our simulation, we are able to recover the exact secret within 12.28 rounds (2 sessions). Another interesting finding for CHC is that we can now estimate the average success rate of guessing attack from the results of brute force attack, though a precise analysis is difficult [32]. According to *Statement 2*, the average success rate is $21.78\% = (C_{112}^{5})^{-\frac{1}{12.28}}$. This technique can also be applied to other complex LRPS systems to determine their security strength when the other analysis techniques are infeasible.

3.3 P2: Large Round Secret Space Principle

Principle 2: *An LRPS system with secret leakage should have a large candidate set for the* **round secret***.*

This principle emphasizes that a large candidate set for the *root secret* is necessary but not sufficient to defend against brute force attack. The large candidate set for the root secret can be *broken down* based on the attack to the round secrets. We use Predicate-based Authentication Services (PAS) [4] as a counterexample to show that a round secret with a small candidate set can be easily recovered and later used to reveal the root secret.

During registration of PAS, a user is asked to remember p secret pairs, each of which includes a secret position and a secret word. At the beginning of each authentication session, the server prompts for an integer index I. Then the user uses I to calculate p predicates as follows: For each pair, the corresponding predicate is the secret position and a secret character. The secret character is the xth character in the secret word (1-based indexing), where $x = 1 + ((I - 1) \mod len)$, and len is the length of the secret word. For example, given two secret pairs ($\langle 2{,}3 \rangle$, sente), ($\langle 4{,}1 \rangle$, logig) and $I = 15$, the predicates are ($\langle 2{,}3 \rangle$, e) and ($\langle 4{,}1 \rangle$, g), where $x = 5 = 1 + ((15 - 1) \mod 5)$, and the secret position $\langle a, b \rangle$ means "at row a and column b". Given these p predicates, the user examines the cells at secret positions in l challenge tables to check whether a secret character is present in its corresponding cell. It yields an answer vector that consists of $p \cdot l$ "present" or "absent" answers with a candidate space of 2^{pl}. This vector is then used to lookup another response table, which provides a many-to-one mapping from 2^{pl} elements to 2^{l} elements. Finally, the user inputs one of those 2^{l} elements indexed by the answer vector to finish an authentication round.

The above many-to-one mapping is used in PAS to confuse the adversary. However, when the round secret only has a small candidate set, many mappings will have the same pre-image and the effective mapping space collapses to the candidate set of the round secret. In PAS, the size of the candidate set for the round secret is $422500 = (25 \times 26)^2$ for the default parameters, where $p = 2$, and there are 25 cells in each challenge table and 26 possible letters for the secret character. It is not difficult to use brute force to recover the round secret of PAS. Figure 2 shows the shrinking of the candidate set size as the number of observed authentication rounds increases. On average, 9.4 rounds are sufficient to recover the exact round secret (1 session). Since all the predicates generated from the same secret pair share the same secret position, after recovering the first round secret, it is easy for the adversary to recover the other round secrets and finally the root secret. A similar attack technique has been used in [18]. The same problem also appears in the S3PAS scheme [35], which is a variant of the CHC scheme [32]. In our experiments, we are able to discover the exact root secret in 8 sessions.

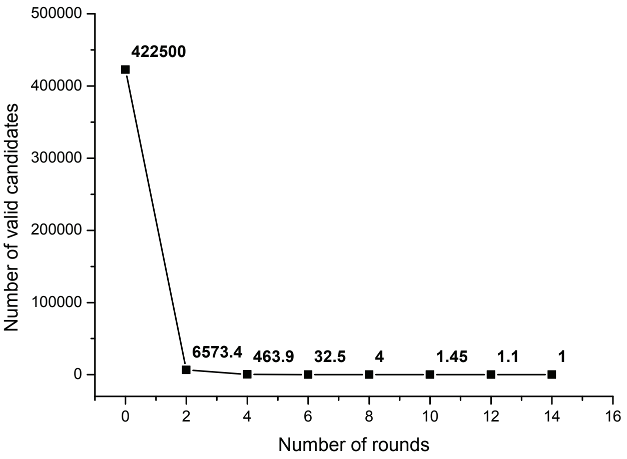

Figure 2. The average number of valid candidates shrinks for PAS.

4 Statistical Attack and Its Defense Principles

4.1 Attack Strategy

Statistical attack is an accumulation-based learning process, where an adversary gradually increases its confidence on relevant targets when more and more cues are available. Compared to brute force attack, statistical attack has fewer limitations as it can be applied to schemes with a large password space. Recall that a user response is statistically biased towards his knowledge of the secret. Theoretically there exists a specific statistical attack for any password system. The efficiency of statistical attack is *design-dependent* and varies with different schemes and different analysis techniques. Here we introduce two general statistical analysis techniques that are able to efficiently extract the root secret of most existing schemes.

The first technique is *probabilistic decision tree*. It works efficiently for the existing schemes based on simple challenges [31, 32, 35, 4]. The procedure is described as follows: 1) Create a score table for each possible individual element or affordable-sized element group in the alphabet of the root secret, where *affordable* means computational feasible to maintain. We refer to a score table whose entry contains t individual elements as an *t-element score table*. 2) For each independent observation of a challenge-response pair, the adversary enumerates every *consistent* decision path that leads to the current response. Each possible decision path is assigned a probability calculated based on the uniform distribution. For the k-out-of-n paradigm, the probability is $p1 = k/n$ for a decision event in which the corresponding individual element belongs to the secret set, and $p0 = 1 - p1$ for the complementary event. For the example decision path X given in Figure 3, its probability is $p(X) = p1 \cdot (p0 \cdot p1)$. After enumerating all consistent decision paths, the adversary sums up the probabilities of these paths and uses the sum p_c to normalize the probability $p(X)$ for each decision path to its conditional probability $p(X|C) = p(X)/p_c$. The conditional probability represents the probability that a decision path is the path chosen by the user when the current response C is observed. After the normalization, the adversary updates the score table using $p(X|C)$. For an entry that appears in a consistent decision path X, its score will be added by $p(X|C)$ if the corresponding event is that the entry belongs to the secret set, otherwise its score will be deducted by $p(X|C)$. 3) Repeat the above step until the number of entries with different score levels reaches a threshold (e.g. finding out k entries with the highest/lowest scores when each entry represents a single element).

The second technique is *counting-based statistical analysis*. The basic idea is to simply maintain a counting table for the occurrences of elements. Multiple counting tables can be maintained simultaneously according to different response groups. The procedure proceeds as follows: 1) Create l counting tables for l response groups. The adversary creates a counting table for each possible response if affordable. "Any response" is still a useful response group if the secret elements appear more or less frequently than the decoy elements *in the challenge*. An entry in a counting table can be an individual element or affordable-sized element group. We refer to a counting table whose entry contains t individual elements as an *t-element counting table*. When $t \geq 2$, we call this type of statistical analysis as *multi-dimensional counting*. 2) For each independent observation of a challenge-response pair, the adversary first decides which counting table is updated according to the observed response. Then each entry in the chosen counting table is incremented by the number of occurrences of the corresponding individual element or element group. If the group of "any response" is used, its counting table is always updated for each observation. 3) Repeat the above step until the number of entries with different score levels reaches a threshold (e.g. finding out k entries with the highest/lowest scores when each entry represents a single element). The score for an entry is a weighted sum of the count values for the same entry in different tables. The weight function is dependent on the specific target scheme and the response grouping strategy.

A *decision path* is an emulation of the user's decision process that consists of multiple decision nodes. Each *decision node* represents a decision event decided by the membership relation of a corresponding entry in the score table, whether or not it belongs to the secret set.

Consider a scheme which shows a four-element window $\langle S_1{:}1,\ S_2{:}2,\ S_3{:}1,\ D_1{:}1 \rangle$ and asks the user to report the sum of the numbers associated with the *first* and *last* secret elements displayed in the window, where $S_i{:}x$ represents a secret element associated with number x, and $D_i{:}y$ represents a decoy element associated with number y. Since the correct response for this challenge is 2 by adding the numbers associated with the first and third elements, its decision path is $X = \langle S_1{:}1 \rangle | \langle D_1{:}1;\ S_3{:}1 \rangle$. There are two segments in this decision path. The first segment implies that S_1 is a secret element, and the second segment implies that D_1 is a decoy element and S_3 is a secret element. There usually exist other decision paths leading to the same response, such as $\langle S_1{:}1 \rangle | \langle D_1{:}1 \rangle$.

Figure 3. Definition and example for decision path

4.2 P3: Uniform Distributed Challenge Principle

Principle 3: *An LRPS system with secret leakage should make the distribution of the elements in each* **challenge** *as uniformly distributed as possible.*

This principle requires that an LRPS system should be able to generate the challenges without knowing the secret[1]. For example, if there is a *structural requirement* in the challenge generation, secret leakage is very likely to happen. Non-uniformly distributed elements in a challenge leave cues for the adversary to recover the secret even without knowing the response. Undercover [27] is a typical counterexample to show secret leakage from biased challenges.

[1] Even if server knows the secret, the secret (or its alternative form, e.g. hash value) should be only used to verify the response.

Undercover ensures that the distribution for each image is unbiased by showing every candidate image exactly once for each authentication session. However, its 2-dimensional distribution is biased in each authentication round, as secret-secret pairs cannot appear in the challenge (at most 1 secret image appearing). We use *2-element counting table* to recover the secret from the challenge. For each pair of candidate images, the count value is zero only if both of them belong to the secret set after a sufficient number of observations. On average, it is sufficient to recover the exact secret within 172.7 rounds (20 sessions), and recover 80% secret elements (five secret images in total) after 126.9 rounds (15 sessions).

The same problem also appears in the CHC scheme [32] and in the low-complexity CAS scheme [31]. Both of them require that at least k secret elements appear in the challenge window, while the challenge window only holds a subset of candidate elements. These structural requirements make the distribution of the elements in each challenge deviate from the uniform distribution. Under default parameters, we are able to recover the exact root secret within 18.18 rounds (2 sessions) for CHC. For the low-complexity CAS scheme, we can recover the exact root secret (i.e. 60 independent secret images) within 2087.2 rounds (105 sessions), and recover 90% secret elements within 870.4 rounds (44 sessions).

The above discussion shows that the consequence of the distribution bias caused by structural requirements in the challenge is subtle to identify and has not been well recognized. In order to prevent leakage from biased challenges, the distribution of the elements in each challenge should be indistinguishable from the uniform distribution. If a structural requirement is *compulsory* in a password system (e.g. at least k secret elements being displayed) but the element distribution in each challenge is not uniform when the challenge window only shows a subset of candidate elements, the scheme should display *all* the candidate elements in each challenge.

4.3 P4: Large Decision Space or Indistinguishable Individual Principle

Principle 4: *An LRPS system with secret leakage should make each individual element* **indistinguishable** *in the probabilistic decision tree if the candidate set for decision paths is* **enumerable**.

This principle is critical to limit the feasibility of probabilistic decision tree attack. The power of probabilistic decision tree stems from its emulation of all possible decision processes leading to the observed response. The emulation creates a tight binding between each challenge and its response, from which the adversary is able to extract the subtle statistical difference during the user's decision if individual elements are distinguishable on consistent decision paths. It is not easy to make each individual element indistinguishable, especially when weight or order information is used in the challenge design. We use the high-complexity CAS scheme [31] as a counterexample to show how probabilistic decision tree efficiently discovers the root secret even when a number of decision paths lead to the same answer.

The high-complexity CAS scheme is another typical scheme based on the k-out-of-n paradigm. During registration, a user is assigned $k = 30$ images as his secret from a pool of $n = 80$ images. In each authentication round, a challenge is an 8×10 grid consists of all the images, one image for each cell. The user is asked to mentally compute a path starting from the cell in the upper-left corner. The computation rule is described as follows: Initially the current cell is the cell in the upper-left corner. If the image in the current cell belongs to the secret set, move down by one cell, otherwise move right by one cell; if the next moving position is out of the grid, it is referred to as an *exit position*. The path computation ends with an exit position. The user reports the answer associated with that exit position to finish an authentication round. The answer is an integer from $[0, 3]$, and is randomly assigned to each exit position. Since the same answer is assigned to multiple exit positions (i.e. 4 answers assigned to 18 exit positions), the adversary cannot easily tell which the exact exit position is. For each exit position, there are also many possible paths leading to it, which further increases the difficulty for the adversary.

Since the default parameters are large ($k = 30$, $n = 80$), brute force attack is infeasible for this scheme. The scheme also follows *Principle 3* to display all the candidate images in each challenge so that the adversary cannot extract the secret only by analyzing the challenges. However, each individual element is distinguishable in this scheme during the decision process, as each element has different impact on the transition of decision paths. One can use probabilistic decision tree to recover the secret from the observations of challenge-response pairs.

Each possible path leading to the observed response forms a decision path in the probabilistic decision tree. The probability of a decision path is decided by the movements on this path. For example, a path $X = \langle$*DOWN*, *RIGHT*, *RIGHT*, *DOWN*$\rangle$ means the first and the fourth images belong to the secret set, while the second and third images do not. The probability $p(X)$ is $p1 \cdot p0 \cdot p0 \cdot p1$, where $p1 = k/n$ and $p0 = 1 - p1$. Initially, we create a *1-element score table*. Given a response with the answer i, we enumerate all consistent decision paths leading to this answer, and update the score table according to the conditional probability $p(X|\text{response} = i)$.

For an 8×10 grid specified by the default parameters, there are 43758 possible decision paths in total, with aver-

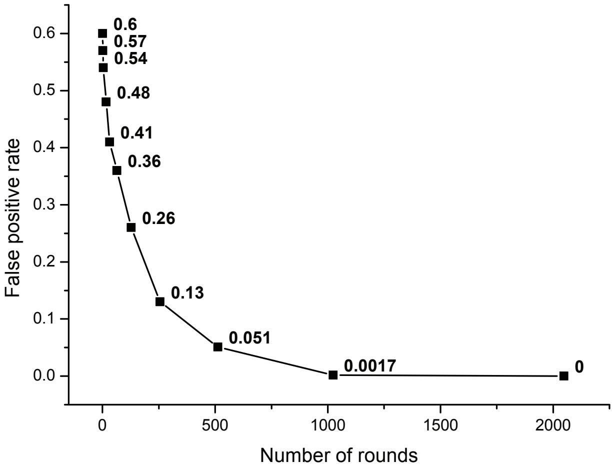

Figure 4. The average false positive rate decreases for the high-complexity CAS scheme.

age path length of 14.5539. For each candidate image, its score is at a significantly high level if it belongs to the secret set after a sufficient number of observations. Figure 4 shows the false positive rate decreasing along with the increasing number of observed authentication rounds. On average, it is sufficient to discover the exact secret within 640.8 rounds (65 sessions), and discover 90% secret elements after 264.7 rounds (27 sessions). Although the required number of session observations is larger, it is still possible for the adversary to collect them using a key logger, and such security strength is achieved only when the user is able to remember 30 independent secret images.

Probabilistic decision tree can also be applied to the low-complexity CAS scheme [31], the CHC scheme [32], the S3PAS scheme [35], and the PAS scheme [4]. All of them are based on simple challenges with an enumerable candidate space for decision paths and the individual element has different impact on the transition of decision paths.

From these counterexamples, we can see that *it is necessary to increase the number of candidate decision paths if it is infeasible to make each individual element indistinguishable in the probabilistic decision tree.* The only known designs that satisfy this indistinguishability requirement are the counting-based schemes [15, 20]. In those schemes, there is no order or weight information associated with each candidate element, which usually distinguishes the elements in decision paths. The user is asked to count their secret elements appearing in the challenge. The final response is based on the count value. For these schemes, probabilistic decision tree attack does not apply, but they may still subject to counting-based statistical analysis attack.

4.4 P5: Indistinguishable Correlation Principle

Principle 5: *An LRPS system with secret leakage should minimize the statistical difference in low-dimensional correlations among each possible* **response***.*

This principle is complementary to *Principle 4* to limit the efficiency of counting-based statistical analysis. Although counting-based statistical analysis is straightforward, it cannot be completely prevented without a secure channel, as the user's response is always statistically biased towards his knowledge of the secret. In the extreme case, the adversary is able to maintain a counting table to hold every candidate for the root secret, and update the table according to every available observation. Using these counting tables, the statistical difference caused by the knowledge of the secret is always identifiable even when the user is asked to make intentional mistakes at a predefined probability only known by the server. Its informal proof is given in Appendix B. In this sense, the counting-based statistical analysis is more powerful than brute force attack if sufficient resources are available to the adversary.

In reality, the resources available to the adversary are not unbounded. The cost of maintaining *t-element counting tables* is $O(n^t)$, which increases exponentially with the number of elements t contained in a table entry, where n is the number of total individual elements. If the adversary fails to maintain a high-dimensional counting table, the correlation information in these tables is safe from the adversary. However, it is still possible for the adversary to exploit the low-dimensional correlation to recover the secret. We use SecHCI [20] as a counterexample to show how it works while brute force and probabilistic decision tree are infeasible.

During registration of SecHCI, a user is assigned k icons as his secret from a pool of n icons. In each authentication round, the challenge is a window consisting of w icons. The user is asked to count how many secret icons appearing in the window. After getting the count value x, the user calculates $r = \lfloor (x \mod 4)/2 \rfloor$. The final response r is either 0 or 1. The challenge is designed so that each individual candidate has the same probability to appear in the window for either response. Hence, it is impossible for the adversary to extract useful information based on *1-element statistical analysis*.

Since the default parameters are large, $k = 14$, $n = 140$, brute force attack is not applicable. Also because it is a counting-based scheme, it is not subject to probabilistic decision tree attack according to *Principle 4*. However, 2-dimensional counting attack is still applicable. Compared to decoy icons, there are 0.599 more pairs on average among secret icons for response 0, and 0.599 less pairs on average among secret icons for response 1. So we can use two *2-element counting tables* to recover its secret, one table

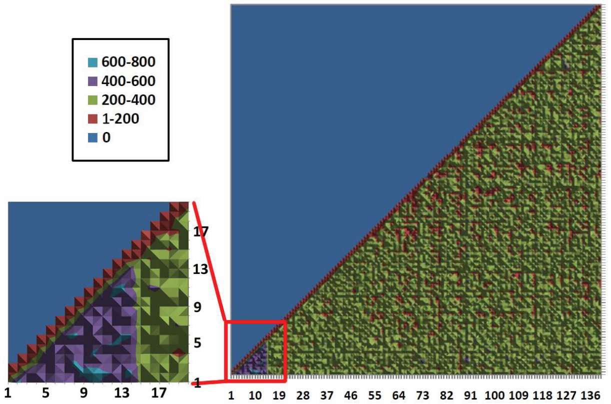

Figure 5. The pair-based score distribution is distorted for SecHCI. The first 14 elements are the secret icons, whose pair-based scores are distinguishable from the scores of other icons.

for each response. We update the count value for each pair displayed in each challenge and each response. The score for each entry is calculated as the value difference between these two tables. For each pair of candidate icons, the score is at a significantly high level if both of them belong to the secret set after a sufficient number of observations. Figure 5 shows the pair-based score distribution after 20000 authentication rounds, from which the secret icons can be easily distinguished. On average, it is sufficient to recover the exact secret with 14219.4 rounds (711 sessions), and recover 90% secret elements after 10799.8 rounds (540 sessions). Since SecHCI follows most of our principles, these numbers are much larger than the schemes we analyzed previously, but it is still far less secure than it is claimed to be [20]. Its security strength is achieved by imposing a high cognitive workload where the user is asked to correctly examine 600 icons (30 icons per round × 20 rounds) one by one for each authentication session.

The secret leakage on pair-based statistics for SecHCI can be fixed by changing its response function from $r = \lfloor (x \mod 4)/2 \rfloor$ to $r = x \mod 2$, where x is the number of secret icons in the challenge window, but this fix will make SecHCI subjects to algebraic attack based on Gaussian elimination [20]. This is also the original motivation of the scheme to use its current function. To further defend against this algebraic attack, a user has to produce incorrect answers with a fixed error probability to create noises as suggested in [15]. This certainly further decreases the scheme's usability. Another design limitation on counting-based scheme is that the response function cannot be in the form of $r = x \mod q$, where q is an integer larger than 2. The detailed explanation for this limitation is given in Appendix C.

5 Usability Costs of Defense Principles

In this section, we provide a qualitative analysis for usability costs of our defense principles. We show the relation and tradeoff among the constraints imposed by our principles and the requirements on human capabilities. This section aims to provide a high level understanding of the quantitative tradeoff analysis to be presented in the next section.

As defined in Section 2, the common parameters of an LRPS system is a tuple (D, k, n, d, w, s). All of the parameters except D (the expected authentication strength) are affected by our principles. The principles related to brute force attack mainly dictates the memory demand for the secret, and the principles related to statistical attack mainly increase the computation workload for each authentication session. Their impacts are also interrelated.

Principles 1 and 2 require a large candidate set for the root secret and the round secret. This implies that either k increases or n increases. An increase in k requires the user to memorize more elements as his secret. An increase in n will not raise the memory demand, but will increase statistical significance of the secret in the whole candidate set, which indirectly increases the computation workload as analyzed later. Principle 2 also directly raises the computation workload, as it indicates a challenge is not safe against brute force attack if it can be solved by using a small number of possible secret elements. In order to increase the candidate space of the round secret, the round secret must be either randomly selected from the root secret [20, 31, 32] or use all elements in the root secret [15, 2]. The former choice requires the user to recognize the current displayed secret elements that change in every round; the latter requires the user to recall a large number of secret elements that would be difficult when k is large. Finally, more elements appearing in a challenge means more computation workload to aggregate them into the correct response. This demands much more effort compared to using a fixed short round secret in a traditional password system.

Principles 3, 4, and 5 have more impact on (d, w, s). Principle 3 requires that the elements in the challenge should be uniformly drawn from the candidate set. Due to previous requirements of large secret space and our preference of minimizing the memory demand for the secret, the value of k is to be small and the value of n is to be large. The consequence of this is that the average number of secret elements displayed in a challenge window, $w \cdot k/n$, cannot be large enough if the window size w is not large. This restricts the number of possible responses to a small value, which raises the success rate d of guessing attack and increases the round number required to achieve an expected authentication strength D. On the other hand, if the window size is large, the LRPS system is limited only for large screen devices and it also increases the difficulty for the user to examine the elements in the challenge window. Regardless of the window size, this principle imposes increased computation workload and the error rate for the user. Principles 4 and 5 further rule out most schemes based on simple challenges. Principle 4 states if a leakage-resistant challenge design is not complex enough to aggregate a large number of secret elements into a response, it leads to a counting problem. Principle 5 further states that only 0 and 1 can be safely used as the response for a counting problem if the modular operation is the only operation used to generate the final response. Hence, the three possible choices for a challenge are: 1) a complex challenge using many secret elements - the round number will be small but the challenge will be very difficult for the user to respond (the average length s of decision paths significantly increased); 2) a counting-based challenge using the modular operation - the round number will be large and the challenge will be relatively easier to respond; and 3) a counting-based challenge using a specially designed response function that has a large number of possible responses and satisfies the correlation indistinguishability condition; however, it will be a challenge to design such a function with acceptable usability. All of the three choices impose a considerable burden on the user.

6 Quantitative Tradeoff Analysis

In this section, we establish a quantitative analysis framework for evaluating the usability cost of typical existing LRPS systems. This framework decomposes the process of human-computer authentication into atomic cognitive operations in psychology. There are four types of atomic cognitive operations commonly used: single/parallel recognition, free/cued recall, single-target/multi-target visual search and simple cognitive arithmetic. Their definitions and performance models are given in Appendix A, which characterizes the relation between experiment parameters and reaction time of an average human. These performance models are used to evaluate the cognitive workload for typical existing LRPS systems. The result in this section provides quantitative assessment of the tradeoff between security and usability of LRPS systems. According to conventions in psychology literature, we will refer user as *subject* in this section.

6.1 Quantitative Analysis Framework

There are two components in our quantitative analysis framework, *Cognitive Workload* (C) and *Memory Demand* (M). Cognitive workload is measured by the total reaction time required by the involved cognitive operations. Long reaction time for each *authentication round* implies that it is difficult for the subject to answer each challenge and the overall error rate is also high. Long reaction time for each *authentication session* implies that the overall cognitive workload is high and the involvement of attention and patience is also high. Memory demand is measured by the number of elements that must be memorized by the subject, which is the prerequisite of any password system. Since this prerequisite process is independent from the authentication process, we consider it as a separate component. Since the precise relation between overall error rate and total reaction time is difficult to measure in controlled psychology experiments, our framework provides *lower bound* estimation for the usability of a human-computer authentication system. The detailed calculation for both components is described as follows.

For cognitive workload, the cost for each authentication round is the sum of average reaction time for all involved atomic cognitive operations. This cost represents the average thinking time of a subject required to answer a challenge. A typical authentication round consists of at least a memory retrieval operation and a simple arithmetic operation. For the graphic-based scheme, visual search is also common. According to the working memory capability theory [25, 9, 30, 29], the average reaction time is not shortened by repetitive rehearsal, when the subject has to maintain more than $4(\pm 1)$ items in his working memory. The rehearsal only improves the accuracy, which represents an inherent limitation of human capabilities. This limitation is also applied to other non-memory operations such as visual search when the item positions are shuffled in each challenge [33]. Overall, the cognitive workload of an authentication session is calculated as the product of the cognitive workload of an authentication round and the round number when the number of the secret items is larger than 5. For the schemes [4, 32] with no more than 5 secret items, we only count once for their memory retrieval operations, assuming that the secret will not be flushed out due to the limitation of working memory capacity.

Besides the reaction time, other usability measurements for cognitive workload (such as user frustration level, concentration load, and motivational effort) are usually collected from standardized testing questionnaires. However, these measurements are susceptible to many implementation and environmental factors, such as screen size, graphic or text-based interface design, and the education background of subjects. In contrast, the influence of those unstable factors has been minimized in more than a century's development of experimental psychology. So the advantage of using performance models of atomic cognitive operations is that they are *implementation-independent*. This property is necessary for a fair comparison between different LRPS designs. Consequently, our estimation of cognitive workload is very consistent with the time costs reported in the

	k	**n**	**Win size**	**Password space**	**Guess Rate /round**	**No. of rounds /login**	**Reported Time /round(sec)**	**HP (C) /round (sec)**	**HP (C) /login (sec)**	**HP (M)**	**HP Total =M$\times$C ($\times 10^2$)**
LPN[15]	15	200	200	1.463×10^{22}	0.50	20	**23.71**	**33.423**	668.45	50.68	**338.74**
APW[2]	16	200	200	8.369×10^{24}	0.10	6	**35.50**	**57.928**	347.57	54.05	**187.87**
CAS Low[31]	60	240	20	2.433×10^{57}	0.50	20	**5.00**	**6.073**	121.46	70.75	**85.94**
CAS High[31]	30	80	80	8.871×10^{21}	0.25	10	**20.00**	**22.099**	220.99	35.38	**78.18**
SecHCI[20]	14	140	30	6.510×10^{18}	0.50	20	**9.00**	**10.638**	212.76	16.51	**35.13**
CHC[32]	5	112	83	1.341×10^{8}	0.22	10	**10.97**	**9.326**	93.26	16.89	**15.75**
PAS[4]	4	N/A	13	4.225×10^{5}	0.25	10	**8.37**	**6.837**	68.37	13.51	**9.24**

Table 1. Tradeoff comparison of representative leakage-resilient password systems for their default parameters.

original papers [20, 31, 32, 4].

For memory demand, the cost for each scheme is a ratio k/λ_{op} between the number of secret items, k, and the *accuracy rate* of corresponding memory retrieval operation within a fixed memorization time, λ_{op}. Since recognition is much easier than recall [14, 25, 29, 23, 10], it is necessary to distinguish the difficulty for different memory retrieval operations. According to [14], λ_{op} is 29.6% for recall and 84.8% for recognition. A better estimation for the memory demand could be the minimum time for the subject to remember all the secrets. However, the lower bound of memorization time is difficult to measure in experimental psychology, as the subject may not realize the precise time point when he just remembers all the secrets. An unconfident subject may take more time to rehearsal than that actually required. Other memory factors, like password interference and recall accuracy over extended periods, may also be considered but are not integrated in our current analysis framework.

Finally, an overall score, HP (standing for *Human Power*), is calculated as the product of cognitive workload score HP(C) and memory demand score HP(M). This score (HP) indicates the expected human capability requirement for a human-computer authentication system.

6.2 High Security at Cost of Heavy Cognitive Demand

Table 1 shows the security strength and HP for the representative LRPS systems based on our quantitative analysis framework. Those systems are listed in the descend order of their HP. All the schemes use their default parameter values except that the round number is adjusted to make the successful rate of random guessing to reach the same level (i.e. the authentication strength of 6-digit PIN). This adjustment is necessary to make a fair comparison as they now have the same strength to defend against an adversary without prior knowledge. The other two points in this table which need explanation are about PAS [4] and CHC [32]. In PAS, we consider the root secret for each authentication session as the predicates instead of the complete secret pairs, due to that the same predicates are used for all the rounds in an authentication. The predicates are the actual root secret of each authentication session. In CHC, the expected successful rate of guessing attack is not reported in the original paper. We estimate it based on *Statement 2*, which is 21.78% derived from our simulation results. The detailed computation of the cognitive workload for those schemes is given in Table 2 of Appendix A.

The column "HP(C)/round" in this table shows the cognitive workload required to solve the challenge in each authentication round. It shows the average thinking time. All of them except LPN [15] and APW [2] are very close to the average time cost reported in the original literatures [20, 31, 32, 4]. For LPN, there is no report on a controlled user study. The scheme is implemented as a public web page, to which the subjects can freely access and get a reward for each successful login. There is no evidence showing that the subjects were asked to memorize their root secret (which are 15 secret positions), and then recall them in each authentication round. Thus, the average time cost reported for each round is very likely to be underestimated, as the recall operations are probably replaced by directly reading their written-down secrets. For APW, its time cost is directly estimated based on the results of LPN (with no actual user study conducted), which implies it could also be underestimated.

This table shows three tiers in these representative schemes. From bottom to top, the schemes in an upper tier have better security against secret leakage at the cost of lower usability. The schemes at the bottom are PAS [4] and CHC [32], which are susceptible to both brute force and statistical attacks. When moving to the middle tier (consisting of CAS [31] and SecHCI [20]), the memory demand increases to make brute force attack infeasible. However, they are still susceptible to statistical attack as the simple challenge used in these schemes is not sufficient to hide the statistical significance of the secret. More cognitive workload is required to mix the secret items with the other items. The top tier consists of LPN [15] and APW [2], which follow all of our design principles. They are immune to both brute force and statistical attacks in practical settings, but impose significantly high usability cost.

There is an interesting finding when looking at the two schemes in the top tier. In our quantitative analysis framework, LPN has a higher HP score but a smaller password space compared to APW. This is because our security measurement is limited to brute force and two generic statistical attacks. It is still possible to find out other more efficient attacks that lower the security strength of APW. The tradeoff relation under our quantitative analysis framework may not strictly follow the order of HP, as it is always feasible to design a scheme with a lower usability for a given security strength. But it is required that the human capability should reach a *lower bound* so as to achieve a high security strength.

The above results provide quantitative evidence for the inherent limitations in the design of LRPS. They indicate the incompetence of human cognitive capabilities in using secure LRPS systems without a secure channel in practical settings. This may also explain why the problem is still open since its first proposal [22] twenty years ago.

7 Related Work

As one of the most important security tools of modern society, the design problem of a secure and usable password system has been extensively investigated. We summarize the closely related research work from the following aspects: attacks, principles, and tradeoff analysis for LRPS systems.

Most of proposed LRPSes have been broken. The recent works on representative attack and analysis include: Golle and Wagner proposed the SAT attack [13] against the CAS schemes [31]; Li et al. demonstrated the brute-force attack [18] against the PAS scheme [4]; they later presented a Gaussian elimination-based algebraic attack [19] against the virtual password system [17]; Asghar et al. introduced a statistical attack [1] against the CHC scheme [32]; Dunphy et al. analyzed a replay-based shoulder surfing attack for recognition-based graphical password systems under a weaker threat model [11]. Compared to them, our paper provides security analysis in a more generic setting, which presents two types of generic attacks that can be used to analyze any LRPS systems. Furthermore, we introduce a new statistical attack, probabilistic decision tree, and a generalized version of existing statistical attacks, multi-dimensional counting. We analyze and re-examine the existing LRPS systems with these new attack tools. Thereby, we discover the vulnerabilities of Undercover [27] and SecHCI [20] that have not been reported before. We notice that a recent work by Perkovic et al. [24] also identified the design flaw of Undercover independently.

Some other design principles have been proposed for LRPS systems. Roth et al. [26] proposed the basic principle of using cognitive trapdoor game, where the knowledge of secret should not be directly revealed during password entry. Li and Shum [20] later suggested another three principles that require time-variant responses, randomness in challenges and responses, and indistinguishability against the statistical analysis. Our principles further extend the coverage by including the defense principles against brute force attack, and provide more concrete guidelines against two generic statistical attacks introduced in our paper.

Until now it is still a challenge to provide a quantitative tradeoff analysis among multiple LRPS systems [5]. As pointed out by Biddle et al. [5], the usability evaluation in prior research lacks consistency, which makes it is difficult to compare those results. Our quantitative analysis framework is the first attempt to provide a uniform usability measurement based on experimental psychology. Based on this framework and our security analysis, we discover that the tradeoff between security and usability is strong, which indicates the inherent limitation in the design of LRPS systems. This limitation was first addressed by Hopper and Blum [15], where they hoped the future research could find out practical solutions for unaided humans that satisfy both security and usability requirements. Unfortunately, from our results, such solution may not exist. Coskun and Herley [8] also reached a similar conclusion by analyzing the efficiency of brute force attack with regards to response entropy. Their conclusion is based on the assumption that a user has to make a large number of sequential binary decisions so as to increase response entropy. However, this assumption may not be valid as humans have a strong parallel processing capability when performing certain visual tasks (e.g. visual search). Other prior research related to LRPS systems can be found in a recent survey paper [5], which summarized the development of new password systems in the past decade.

We remark that our quantitative analysis framework is still in its preliminary stage. We would like to point out two limitations in our current work: 1) Since the cognitive workload is not totally independent with the memory demand, it is possible to improve the overall score calculation instead of using the product operation (i.e. HP= M×C); 2) Error rate is currently not included in our analysis framework as it is difficult for experimental psychology to provide the general relation between thinking time and error rate. Certain approximation can be added to improve the precision of this framework in the future.

8 Conclusion

In this paper, we provided a comprehensive analysis for the inherent tradeoff between security and usability in designing a leakage-resilient password system. We analyzed the impacts of two types of generic attacks, brute force and statistical attacks, on the existing schemes designed for unaided humans. Unlike the specific attacks proposed before (such as SAT [13] and Gaussian elimination [19]), these two generic attacks, as demonstrated in our paper, cannot be mitigated without involving considerable demand on human capabilities. We introduced five principles that are necessary to achieve leakage resilience when a secure channel is unavailable. Usability costs for these principles are analyzed. Our findings indicate that either high memory demand or high cognitive workload is unavoidable in the design of secure LRPS for unaided humans. To further understand the tradeoff between security and usability, we established the first quantitative analysis framework on usability costs. Our result shows that there is a strong tradeoff between security and usability, indicating that an unaided human may not be competent enough to use a secure leakage-resilient password system in practical settings.

References

[1] H. J. Asghar, S. Li, J. Pieprzyk, and H. Wang. Cryptanalysis of the convex hull click human identification protocol. In *Proceedings of the 13th international conference on Information security*, pages 24–30, 2010.

[2] H. J. Asghar, J. Pieprzyk, and H. Wang. A new human identification protocol and coppersmith's baby-step giant-step algorithm. In *Proceedings of the 8th international conference on Applied cryptography and network security*, pages 349–366, 2010.

[3] A. D. Baddeley. *The Essential Handbook of Memory Disorders for Clinicians*, chapter 1, pages 1–13. John Wiley & Sons, 2004.

[4] X. Bai, W. Gu, S. Chellappan, X. Wang, D. Xuan, and B. Ma. Pas: Predicate-based authentication services against powerful passive adversaries. In *Proceedings of the 2008 Annual Computer Security Applications Conference*, pages 433–442, 2008.

[5] R. Biddle, S. Chiasson, and P. C. van Oorschot. Graphical passwords: Learning from the first twelve years. In *Technical Report TR-11-01*, 2011.

[6] J. I. D. Campbell and Q. Xue. Cognitive arithmetic across cultures. *Journal of Experimental Psychology: General*, 130(2):299–315, 2001.

[7] L. Corbina and J. Marquer. Effect of a simple experimental control: The recall constraint in sternberg's memory scanning task. *European Journal of Cognitive Psychology*, 20(5):913–935, 2008.

[8] B. Coskun and C. Herley. Can "something you know" be saved? In *Proceedings of the 11th international conference on Information Security*, pages 421–440, 2008.

[9] N. Cowan. The magical number 4 in short-term memory: a reconsideration of mental storage capacity. *Behavioral and Brain Sciences*, 24(1):87–114, 2001.

[10] F. I. Craik and J. M. McDowd. Age differences in recall and recognition. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 13(3):474–479, 1987.

[11] P. Dunphy, A. P. Heiner, and N. Asokan. A closer look at recognition-based graphical passwords on mobile devices. In *Proceedings of the Sixth Symposium on Usable Privacy and Security*, pages 3:1–3:12, 2010.

[12] D. L. Fisher. Central capacity limits in consistent mapping, visual search tasks: Four channels or more? *Cognitive Psychology*, 16(4):449–484, 1984.

[13] P. Golle and D. Wagner. Cryptanalysis of a cognitive authentication scheme (extended abstract). In *Proceedings of the 2007 IEEE Symposium on Security and Privacy*, pages 66–70, 2007.

[14] R. M. Hogan and W. Kintsch. Differential effects of study and test trials on long-term recognition and recall. *Journal of Verbal Learning and Verbal Behavior*, 10(5):562–567, 1971.

[15] N. J. Hopper and M. Blum. Secure human identification protocols. In *Proceedings of the 7th International Conference on the Theory and Application of Cryptology and Information Security: Advances in Cryptology*, pages 52–66, 2001.

[16] T. S. Horowitz and J. M. Wolfe. Search for multiple targets: Remember the targets, forget the search. *Perception & Psychophysics*, 63(2):272–285, 2001.

[17] M. Lei, Y. Xiao, S. Vrbsky, C.-C. Li, and L. Liu. A virtual password scheme to protect passwords. In *Proceedings of IEEE International Conference on Communications*, pages 1536–1540, 2008.

[18] S. Li, H. Asghar, J. Pieprzyk, A.-R. Sadeghi, R. Schmitz, and H. Wang. On the security of pas (predicate-based authentication service). In *Proceedings of the 2009 Annual Computer Security Applications Conference*, pages 209–218, 2009.

[19] S. Li, S. A. Khayam, A.-R. Sadeghi, and R. Schmitz. Breaking randomized linear generation functions based virtual password system. In *Proceedings of the 2010 IEEE International Conference on Communications*, pages 23–27, 2010.

[20] S. Li and H. yeung Shum. Secure human-computer identification (interface) systems against peeping attacks: SecHCI. In *Cryptology ePrint Archive, Report 2005/268*, 2005.

[21] J. Long and J. Wiles. *No Tech Hacking: A Guide to Social Engineering, Dumpster Diving, and Shoulder Surfing*. Syngress, 2008.

[22] T. Matsumoto and H. Imai. Human identification through insecure channel. In *Proceedings of the 10th annual international conference on Theory and application of cryptographic techniques*, pages 409–421, 1991.

[23] P. A. Nobel and R. M. Shiffrin. Retrieval processes in recognition and cued recall. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 27(2):384–413, 2001.

[24] T. Perkovic, A. Mumtaz, Y. Javed, S. Li, S. A. Khayam, and M. Cagalj. Breaking undercover: Exploiting design flaws and nonuniform human behavior. In *Proceedings of the Seventh Symposium on Usable Privacy and Security*, 2011.

[25] D. Rohrer and J. Wixted. An analysis of latency and interresponse time in free recall. *Memory. & Cognition*, 22(5):511–524, 1994.

[26] V. Roth, K. Richter, and R. Freidinger. A pin-entry method resilient against shoulder surfing. In *Proceedings of the 11th ACM conference on Computer and communications security*, pages 236–245, 2004.

[27] H. Sasamoto, N. Christin, and E. Hayashi. Undercover: authentication usable in front of prying eyes. In *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, pages 183–192, 2008.

[28] S. Sternberg. Memory-scanning: Mental processes revealed by reaction-time experiments. *American Scientist*, 57:421–457, 1969.

[29] N. Unsworth and R. W. Engle. The nature of individual differences in working memory capacity: Active maintenance in primary memory and controlled search from secondary memory. *Psychological Review*, 114(1):104–132, 2007.

[30] E. Vogel and M. Machizawa. Neural activity predicts individual differences in visual working memory capacity. *Nature*, 428(6984):748–751, 2004.

[31] D. Weinshall. Cognitive authentication schemes safe against spyware (short paper). In *Proceedings of the 2006 IEEE Symposium on Security and Privacy*, pages 295–300, 2006.

[32] S. Wiedenbeck, J. Waters, L. Sobrado, and J.-C. Birget. Design and evaluation of a shoulder-surfing resistant graphical password scheme. In *Proceedings of the working conference on Advanced visual interfaces*, pages 177–184, 2006.

[33] G. F. Woodman and M. M. Chun. The role of working memory and long-term memory in visual search. *Visual Cognition*, 14(4-8):808–830, 2006.

[34] G. F. Woodman and S. J. Luck. Visual search is slowed when visuospatial working memory is occupied. *Psychonomic Bulletin and Review*, 11(2):269–274, 2004.

[35] H. Zhao and X. Li. S3pas: A scalable shoulder-surfing resistant textual-graphical password authentication scheme. In *Proceedings of the 21st International Conference on Advanced Information Networking and Applications Workshops - Volume 02*, pages 467–472, 2007.

A Atomic Cognitive Operations and HP(C) Calculation

There are four types of atomic cognitive operations commonly used in human-computer authentication systems. Their definitions and performance models are introduced in this section, which characterize the relation between experiment parameters and reaction time (RT) of an average human. These performance models are utilized to evaluate the cognitive workload for the existing LRPS systems, as shown in Table 2.

A.1 (Single/Parallel) Recognition

Recognition is the process to correctly judge whether a presented item have been encountered before. Recognition can be considered as a matching process of comparing presented items with those stored in memory. The reaction time of a recognition operation depends on the number of items which a subject memorizes. The item set in the subject's memory is referred to as a *positive set*. For single item recognition, that is, only one item is shown to the subject each time, one of the most well-known recognition models [28] evaluates the reaction time as $RT = 0.3964 + 0.0383 \cdot k$, where k is the size of the positive set. When multiple items are present simultaneously, the subject is able to perform recognition in parallel. According to the working memory capacity theory [12, 9, 30], the maximum number of parallel recognition channels is limited to 4 for an average subject. The reaction time of recognizing x items displayed simultaneously can be estimated as $RT = (0.3964 + 0.0383 \cdot k) \cdot \lceil x/4 \rceil$.

Recognition is a common operation in LRPS, which is used by the subject to judge whether an element appearing in the challenge belongs to the positive set. The high-complexity CAS scheme [31] is an example for single item recognition, where the subject is asked to recognize an image in the current position before deciding which image will be recognized in the next move. The low-complexity CAS scheme [31] and SecHCI [20] are examples of parallel recognition. In the low-complexity CAS scheme, the subject needs to find out the first and the last secret image appearing in a window consisting of 20 images; while in SecHCI, the subject needs to identify all his secret images among 30 candidate images.

A.2 (Free/Cued) Recall

Recall is the other principal method of memory retrieval [3], which is defined as reproducing the stimulus items. Compared to recognition, the recall process is much slower [10, 23]. The common interpretation of this is that recall is associated with greater resource costs than recognition [10]. Recall might be carried out as a slow process of serial search while recognition as a fast process of parallel retrieval [23].

Free recall and cued recall are two basic recall types. In free recall, the subject is given a list of items to remember and then is tested by recalling them in any order [25]. In cued recall, the subject is given a list of items with cues to remember, and cues are given in the test. Cues act as guides to what the person is supposed to remember. For example, given "a body of water", the phrase is the cue for the word "pond" [10]. Many psychological experiments have shown that the reaction time of free recall increases *exponential* as the size of positive set increases [25, 29]. In contrast, the reaction time for cued recall is much shorter and only increases *linearly* [10, 23].

Some LRPS systems require subjects to recall all his secret items during the authentication. The LPN scheme [15] and the APW scheme [2] are two examples, where the subject has to recall all the secret items and their corresponding locations in order to read the challenge digit associated with

	Atomic Cognitive Operations	Calculation of HP (C) per round
LPN[15]	Cued-recall with position, counting, mod	$(0.3964 + 0.0383 \cdot k \cdot \varphi \cdot \gamma) \cdot k + (k/2 - 1) \cdot \alpha_0 + 1 \cdot \alpha_0$
APW[2]	Cued-recall with position, large addition, mod	$((0.3694 + 0.0383 \cdot k \cdot \varphi \cdot \gamma) + 1 \cdot \alpha_3 + 1 \cdot \alpha_0) \cdot k$
CAS Low[31]	Parallel recognition, xor	$(0.3694 + 0.0383 \cdot k) \cdot \lceil 7.4038/4 \rceil + 1 \cdot \alpha_0$
CAS High[31]	Recognition	$(0.3694 + 0.0383 \cdot k) \cdot 14.5539$
SecHCI[20]	Parallel Recognition, counting, mod, small division	$(0.3694 + 0.0383 \cdot k) \cdot (\lceil 30/4 \rceil) + 2 \cdot \alpha_0 + 1 \cdot \alpha_0 + 1 \cdot \alpha_2$
CHC[32]	Cued-recall, Multi-target visual search (3-based)	$((0.3694 + 0.0383 \cdot k \cdot \varphi) \cdot 5/10) + (0.583 + 0.0529 \cdot 83) \cdot 1.8$
PAS[4]	Cued-recall, single-target visual search, small addition	$(0.3694 + 0.0383 \cdot 2 \cdot \varphi) \cdot 4/10 + (0.583 + 0.0529 \cdot 13) \cdot 4 + 2 \cdot \alpha_1$

Table 2. Detailed computation of cognitive workload for representative leakage-resilient password systems. $\alpha_0 = 0.738$**,** $\alpha_1 = 0.773$**,** $\alpha_2 = 0.959$**,** $\alpha_3 = 0.924$ **are the average reaction time for arithmetic problems involving 0 or 1, small addition, small division, and large addition correspondingly.** $\varphi = 1.969$ **is the ratio of cued recall compared to single item recognition, while** $\gamma = 1.317$ **is the additional penalty caused by simultaneously recalling the position of an item. For CAS Low and High,** 7.4038 **and** 14.5539 **are the average lengths of their decision paths, respectively.**

each secret item. These recall processes should be classified as free recall as cues are not presented. However, no experimental data have been provided in psychology literatures for a large positive set consisted of 15 items required by these schemes, while the common size for a positive set is 8 for free recall. Since it is difficult to decide whether the exponential trend still holds when the positive set is large, we use the reaction time of cued recall as a conservative estimation for free recall used in those schemes. According to the experimental results in [23, 7], the formula for the reaction time of cued recall is $RT = (0.3964 + 0.0383 \cdot \varphi \cdot \gamma \cdot k)$, where φ is the ratio of cued recall compared to single item recognition ($\varphi = 1.969$ in [23]), while γ is the additional penalty if subjects are required to simultaneously recalling the position of an item ($\gamma = 1.317$ in [7]).

A.3 (Single-target/Multi-target) Visual Search

Visual search is a perceptual task that involves an active scan of the visual environment for particular targets among other distractors. The measure of the involvement of attention in visual search is often manifested as a slope of the response time function over the number of items displayed (referred to as *window size*) [33]. For single-target visual search, searching a single target among a set of items, its reaction time is believed to be linear as the window size increases [34, 33] and can be estimated as $RT = 0.583 + 0.0529 \cdot w$ [34], where w is the window size. For multi-target visual search, the reaction time is accelerated instead of increasing linearly as the number of targets increases in a fixed-sized window [16].

Visual search is usually used in LRPS systems based on simple challenges. PAS [4] and CHC [32] are examples of using single-target visual search and multi-target visual search, respectively. In PAS, the subject is asked to scan a table cell containing 13 random letters to check whether a secret letter is present or not. In CHC, the subject needs to locate 3 secret elements in a window to form a triangle. According to the results from [16], the reaction time of 3-targets visual search in CHC is approximately 1.8 times longer than that of single-target visual search in the same window.

A.4 Simple Cognitive Arithmetic

Simple cognitive arithmetic is a mental task to solve simple problems involving basic arithmetic operations (e.g., $3+4$, $7-3$, 3×4, $12 \div 3$). The simple arithmetic problems can be further divided into three subsets, small, large and zero-and-one problems [6]. For both addition and multiplication, small problems are defined as those with the product of two operands smaller than or equal to 25, and large problems are defined as those with the product of two operands larger than 25. The small and large problems in subtraction and division are defined on the basis of the inverse relationships between addition and subtraction and between multiplication and division. Zero-and-one problem is defined as involving 0 or 1 as an operand or answer. The common instances of zero-and-one problems include counting, exclusive-or, and mod 2. As reported in the experiments of [6], the average reaction time is 0.773 seconds for small addition, 0.959 seconds for small division, 0.924 seconds for large addition, and 0.738 seconds for zero-and-one problems.

Simple cognitive arithmetic is usually used in LRPS systems based on algebra problems. The counting-based schemes [15, 20] are examples, where the subject is asked to count the number of secret icons appearing in the challenge, and use the count value to calculate a response based on a simple algebraic function.

B Strength of Multi-dimensional Counting

Assuming the user makes mistakes in the responses with a fixed error probability ρ, the average success rate of guessing attack on the "correct" response for each authentication round is d, the number of candidate root secrets is N, the adversary cannot distinguish the true secret only when the equation $\frac{1-\rho}{(1-\rho)(Nd-1)+\rho \cdot Nd} = \frac{1}{N-1}$ holds, which means the decoys get the same count value as that of the secret. Solving the equation gives $\rho = 1 - d$. Therefore, the user should make the correct response with probability $1 - \rho = d$. This implies that the user's decision process is similar to a random guessing, which defeats the purpose of the authentication.

C Design Limitation of Counting-Based LRPSes

In our simulation experiments, we discover that pair-based statistical difference in Counting-based LRPSes appears when q is larger than 2, and increases with the value of $|r - w \cdot k/n|$, where r is the response value, w is the window size, k is the number of secret elements, and n is the total number of elements. This can be explained as follows: For a response, if the expected number of secret elements in a window is less than the expected number $w \cdot k/n$ derived from the uniform distribution, the number of pairs among secret elements is also less than the expected number $C^2_{wk/n}$, and the number of pairs among decoy elements is larger than the expected number derived from the uniform distribution, and vice versa. The adversary is then able to distinguish the secret elements from the other elements by grouping the observations of different responses. Such attack restricts a counting-based scheme from using a larger q and thus reducing the number of rounds of an authentication session without using a more complex response function.