JOURNAL OF COMBINATORIAL OPTIMIZATION, , 1–13 ()
© Kluwer Academic Publishers, Boston. Manufactured in The Netherlands.

Practical Human-Machine Identification over Insecure Channels*

XIANG-YANG LI AND SHANG-HUA TENG xli2, steng@cs.uiuc.edu
Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL 61801.

Editor: Ding-Zhu Du

Abstract. Human-machine identification is an important problem in cryptography that has applications in network access, electronic commerce, and smart-card design. It is a hard problem largely because human users have a very limited capacity in memorizing secrets and in performing protocols. Therefore, in addition to the requirement that a human-machine identification scheme must be provably secure, the scheme has to be practical in the sense that it must be feasible for a human user to participate. In this paper, we develop a new scheme for this problem. Our scheme improves upon some of the previously proposed human-machine identification schemes. We present a vigorous security analysis of our scheme. We also present some attacks to show previously proposed schemes could be vulnerable.

Keywords: cryptography, human-machine identification, network security.

1. Introduction

In several practical applications, it is necessary to "prove" one's identity. There are two interesting cases. (1) a computer, with a given access from a user, electronically proves its identity to another computer. (2) a human user identifies himself/herself to a computer in scenarios such as network access, electronic commerce, and transactions over telephones. The development of zero-knowledge-based identification protocols [1, 4, 2, 3, 6, 8, 9, 10] makes it possible to solve the identification problem among computers with a reasonable complexity. However, most of these protocols do not apply to the second case for the identification between a human user and a machine, largely because a human user has a very limited capacity in memorizing secrets and in performing protocols. Most zero-knowledge protocols require a human user to remember too much; these protocols are too complex for a human user to be involved. A human-machine identification scheme must be practical in the sense that it must be simple enough for a human user to participate.

Several human-machine identification protocols have been proposed. Some use biometrics such as the fingerprint of a user, the voice of a user, and the retinal blood-vessel pattern of human eyes. In these approaches, special physical devices are designed for obtaining and for checking the related biometric information. The rate of recognition is main block to use them in human-machine identification.

Most password-based identification and authentication schemes that do not use any auxiliary device are often vulnerable to peeping attack, replay attack, and several other forms of attacks. Some enhancements are made for password-based schemes. For example, Security Dynamics proposed a human-machine identification scheme. A user and a verifier agree upon a secret seed number in advance, which is used to generate a pseudo-random number sequence. The user receives a SecurID card from the verifier. The card generates a new pseudo-random number every minute starting from the agreed seed number. The generated random number is called a *token*. It is computationally impossible to predict the token of the user by observing the history token sequence, provided that the seed number is unknown. When the user needs to prove his/her identity

* Supported in part by an NSF CAREER award (CCR-9502540), an Alfred P. Sloan Research Fellowship, and an Hitachi Software research grant.

to the verifier, he/she should send his/her password and the number showed on the card at that time. From the user's name and password, the verifier finds the seed number in the password database. The verifier uses the seed number and the current time to generate the correct token. When the token is received, the verifier compares it with the token computed. The verifier accepts the identification if they are equal.

In 1991, Matsumoto and Imai [7] presented a human-machine identification scheme over insecure channels. It is designed to accommodate human user's ability of memorizing secrets and of processing protocols. They use a challenge-and-response scheme. Each challenge is a permutation of a predetermined alphabet of letters. Among these letters, some are used to define a window in the challenge. The response should contain the secret word as a subsequence string at the location as specified by the window. Details of this scheme will be given in Section 2.

However, this identification scheme could be vulnerable to some attacks. We will present some of these attacks in Section 2. Based on our security analysis, we propose a new identification scheme in Section 3, which is an improvement of the scheme of Matsumoto and Imai. Our scheme will retain the simplicity of their original scheme so that it is still suitable for practical use; but the level of security of our scheme is much improved.

2. The Scheme of Matsumoto and Imai

The object of the scheme of Matsumoto and Imai [7] is to accommodate human user's ability of memorizing secrets and of processing protocols, and to prevent an adversary from recovering the password transmitted from a user to a machine. In this section we review this identification scheme and present some attacks to show that this scheme could be vulnerable.

2.1. The Details of the Scheme

The identification scheme defines two finite sets: the *question alphabet* $Q = \{q_0, q_1, \ldots, q_{L-1}\}$ and the *answer alphabet* $A = \{a_0, a_1, \ldots, a_{|A|-1}\}$. There are two players in the scheme, a user U and a machine M. U and M agree upon some secrets in advance: the first secret is a word $\boldsymbol{s} = (s_0 \circ s_1 \ldots \circ s_{k-1}) \in A^k$ of length k and the second secret is a subset $W \subseteq Q$ that has k elements; $\boldsymbol{s}$ is called the *password* and W is called the *window alphabet.* To be secure, $|Q|$ should be reasonably larger than k.

The identification scheme uses the challenge-and-response method. Each challenge from M is a permutation Π of Q. Let $S(Q)$ denote the set of all permutations of Q. When receiving a challenge $(\Pi(q_0) \circ \Pi(q_1) \ldots \circ \Pi(q_{L-1}))$, U provides a reply which is also a word $(a_0 \circ a_1 \circ \ldots \circ a_{L-1})$ of L characters, where a_i is chosen according to the following procedure: if $\Pi(q_i) \notin W$, a_i is a random element chosen uniformly from A; if $\Pi(q_i)$ is the $(j+1)$th element of the challenge in W from left to right, a_i is s_j. This kind of reply is called *correct answer* for Π. Let reply(Π) be the set of all correct answers for challenge Π.

Scheme `Matsumoto-Imai-Identification`(m, n)

1. Machine M generates m permutations $\Pi_1, \ldots \Pi_m \in S(Q)$ and sends them to user U (a simple way to generate these challenges is to select permutations uniformly at random from $S(Q)$).
2. U generates m answers $\boldsymbol{a}_1, \boldsymbol{a}_2, \ldots, \boldsymbol{a}_m$ and sends them back to M.
3. M generates a "correct set", $C = \{i | \boldsymbol{a}_i \in \text{reply}(\Pi_i)\}$.
4. M accepts U if $|C| \geq n$, and denies U if $|C| < n$.

2.2. A Simple Example

A simplified example with $m = n = 1$ is described here to illustrate the scheme. In this example, the question alphabet is $Q = \{1,2,3,4,5,6,7,8\}$, the window alphabet is $W = \{1,2,4,6\}$, and the answer alphabet is $A = \{1,2,3,4\}$. User U has a secret word $\boldsymbol{s} = 3124$ shared with machine M. Figure 1 shows a challenge and a response between M and U. The bars in the positions of q show the window positions of the challenge.

The question Π proposed by M is

$\overline{2}$	8	5	$\overline{1}$	7	3	$\overline{6}$	$\overline{4}$

and, a possible answer generated by U is

3	1	2	1	3	4	2	4

Figure 1: A simplified example.

M verifies whether answer $\boldsymbol{a} \in \text{reply}(\Pi)$. In this example, M accepts user U.

If an intruder wants to get the secret word of the user, he/she must guess the window alphabet correctly. Since the window size is 4 and the challenge size is 8, the probability that an intruder guesses the window W correctly is $1/\binom{|Q|}{|W|}=1/\binom{8}{4}$.

2.3. Possible Attacks

The identification scheme of Matsumoto and Imai is attractive for practical use. However, if the parameter of the scheme is not selected properly, this scheme could be weak against attacks. To a certain extent, some of the weaknesses are inevitable.

The goal of attacks to this identification scheme is to reveal the window alphabet W and the corresponding secret word $\boldsymbol{s}$. There are two possible ways to achieve this: one is to obtain the window position for a specific permutation, the other is to get the window alphabet and the secret word statistically from the challenges and the responses. We call the first one the *replay challenge attack*, and the second one the *statistical attack.*

Recall that the question alphabet Q of the scheme is divided into two subsets: the window subset, and the subset for generating random part of the permutation, which is used to hide the information of the window subset. The challenge is the permutation of the question alphabet. The response given by user U is determined by the challenge proposed by machine M. Therefore it is possible to execute the replay challenge attack.

We illustrate some potential attacks to this scheme. Some of these attacks have been studied before. We mention them here to motivate the analysis of our new scheme.

2.3.1. Known-A Random Attack This attack was presented by Matsumoto and Imai [7]. An attacker knows the protocol and also knows the concrete value of Q, A, and $|W| = k$, but does not know the concrete value of W nor $\boldsymbol{s}$. In reply to a permutation Π, the attacker transfers a random answer which is uniformly selected from A^L.

The success probability of the known-A attack is given by

$$p_k = \sum_{j=n}^{m} \binom{m}{j} p^j (1-p)^{(m-j)},$$

where $p = 1/(|A|^k(1-\sum_{i=1}^{|A|-1}\binom{|A|}{i}(\frac{i}{|A|})^L))$, the probability of successfully guessing a correct reply to a permutation.

Therefore, if the answer alphabet and the window size are sufficiently large, it is difficult to execute this attack.

2.3.2. A-guessing Random Attack This attack was also given by Matsumoto and Imai [7]. An attacker obtains a subset of the answer alphabet by only observing one reply given by the user, which makes it easier to execute the guess answer attack. In fact, after getting this subset, the known-A random attack is possible.

The success probability of the A-guessing random attack is given by

$$p_g = \frac{p_k}{\binom{|\omega|}{|A|}},$$

where p_k is the success probability of the known-A attack and ω is the possible answer alphabet subset guessed by the attacker, assuming $A \subset \omega$.

2.3.3. Passive Attack This attack was given by Wang, Hwang and Tsai [11]. It is based on the following observation: according to the scheme of Matsumoto and Imai, the password $\boldsymbol{s}$ contains every element of the answer alphabet at least once. The attacker can reduce the number of trials to reveal the secret password by this observation. According to their analysis, the number of trials needed in this attack will be less than $\binom{|Q|}{|W|}$.

For simplicity, they [11] consider the case where $m = n = 1$. Let :

- $A = \{a_0, a_1, a_2, \ldots, a_{|A|-1}\}$,
- t_i denotes the number of occurrence of a_i in the answer, hence $|Q| = \sum_{i=0}^{|A|-1} t_i$,
- s_i denotes the number of occurrence of a_i in the secret password $\boldsymbol{s}$, hence $|W| = \sum_{i=0}^{|A|-1} s_i$.

Wang, Hwang and Tsai [11] show that the secret word would be revealed in at most

$$\sum_{s_0+s_1+\ldots+s_{|A|-1}=|W|} \binom{t_0}{s_0}\binom{t_1}{s_1}\cdots\binom{t_{|A|-1}}{s_{|A|-1}}$$

trials. They have proved that the number of trials of passive attack is less than the number $\binom{|Q|}{|A|}$.

2.3.4. Common Answer Subsequence Statistical Attack To the best of our knowledge, this attack is new. It is based on the following observation: every reply given by U contains the secret word $\boldsymbol{s}$ as a subsequence string. If a string is a subsequence string in all of the r replied answers, we call it a *common subsequence string* of the r answers. Any common subsequence string is a candidate of the secret word. An attacker maintains all k-length common subsequence strings in all answers given by user U.

Let $M_m(\boldsymbol{s})$ be the set of all strings of m characters that contain a string $\boldsymbol{s}$ of k characters as a subsequence. We will show that for $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$ of k characters, $|M_m(\boldsymbol{s}_1)| = |M_m(\boldsymbol{s}_2)|$. Let $p_q(m,k)$ denote the probability that a string $\boldsymbol{s}$ of length k occurs as a subsequence in a random string of length m. This is well defined because the size of $M_m(\boldsymbol{s})$ is independent of the choice of $\boldsymbol{s}$, i.e., $p_q(m,k) = |M_m(\boldsymbol{s})|/q^m$ for any k-character string $\boldsymbol{s}$. We will show that the concrete value of $p_q(m,k)$ is

$$p_q(m,k) = 1 - \frac{\sum_{i=0}^{k-1}(\binom{m}{i}(q-1)^{m-i})}{q^m},$$

where q is the size of the answer alphabet. The number of common subsequence strings in all r replies is $q^k p_q(m,k)^r$. To prevent the common answer subsequence statistical attack, the probability $p_q(m,k)$ should be close to 1. Otherwise, the attacker can get only one common subsequence string by observing a few replies. This common subsequence string should be the secret word because the secret word $\boldsymbol{s}$ is one of the common subsequence strings.

As proposed in [7], the practical value of m is 36, the size of the window alphabet is about 10, and the size of the answer alphabet is about 4. In this case, $p_q(m,k)$ is very close to 0. We can use common answer subsequence attack to get the common secret word by observing about 2^7 replies. This is far more efficient than the attack of guessing the position of the window in the challenge.

2.3.5. Replay Challenge Attack This attack is also proposed by Wang, Hwang and Tsai [11]. An attacker impersonates the host to replay an intercepted challenge to the user to obtain the answers from the user. By collecting a few answers of the same challenge, the attacker can figure out the secret word of the user with a high probability. If the attacker discovers that some corresponding positions in the reply answers $\boldsymbol{a}$ and $\boldsymbol{a}'$ whose contents are distinct, these positions can not be the positions of the window W, corresponding to the permutation Π. They prove the following two theorems about the efficiency of the replay attack.

Theorem 1 *The window W of the scheme of Matsumoto and Imai with $m = n = 1$ can be found in*

$$\sum_{t=0}^{|Q|-|W|} \frac{\binom{|Q|-|W|}{t}(|A|-1)^t}{|A|^{|Q|-|W|}} \binom{|Q|-t}{|W|}$$

expected trials, if the attacker replays the same permutation once.

Theorem 2 *The window W of the scheme of Matsumoto and Imai with $m = n = 1$ can be found in*

$$\sum_{t=0}^{|Q|-|W|} \frac{\binom{|Q|-|W|}{t}(|A|^r-1)^t}{|A|^{r(|Q|-|W|)}} \binom{|Q|-t}{|W|}$$

expected trials, if the attacker replays the same permutation r times.

Corollary 3 *If the number of times that the attacker replays the same permutation tends to infinity, the expected number of trials to find the window W will degrade to 1.*

They also give examples to show the validity of their replay attack. For $|Q| = 50, |W| = 10, |A| = 3, m = 1$, and $n = 1$, it only needs about 9 replays to determine the secret word with very high probability.

The replay attack is efficient only when it is feasible for the attacker to replay the challenge many times. When it is difficult for the attacker to perform replay attack, the common answer subsequence statistical attack is among one of the most efficient forms of attacks.

3. A New Human-Machine Identification Scheme

We present a new scheme for the human-machine identification problem. Our scheme is more robust than the original scheme of Matsumoto and Imai and some of its improvements. We give a high-level description of our scheme.

Let Q be a finite set of letters. Unlike the scheme of Matsumoto and Imai in which Q must have a reasonable size, our scheme is working even for $Q = \{0, 1\}$. User U and machine M share three pieces of secrets, a shift pattern $\boldsymbol{shift}$, a key pattern $\boldsymbol{key}$, and a secret word $\boldsymbol{s} = (s_1 \circ s_2 \ldots \circ s_k)$, all are words of Q. In addition, the length of $\boldsymbol{key}$ is equal to that of $\boldsymbol{s}$.

The challenge from M is a word $\boldsymbol{q} = (q_0 \circ q_2 \ldots \circ q_{m-1})$ from Q^m for some positive integer m. It is in fact more convenient to think that $\boldsymbol{q}$ is embedded on a ring, i.e., we will use $i+1$ for $(i+1) \bmod m$ for the subscript variables. For example $q_m = q_0$.

When receiving a challenge $\boldsymbol{q}$, U finds the subsequence of $\boldsymbol{q}$ which is the lexical-first occurrence of $\boldsymbol{shift}$ in $\boldsymbol{q}$. Notice that a subsequence may not be consecutive. We will formally define lexical-first occurrence later. Assume the last letter of this subsequence is q_i. U finds the subsequence of $\boldsymbol{key}$ in Q starting from location i. Keep in mind that the subscript arithmetic is mod m. U then generates a response word by replacing this subsequence by $\boldsymbol{s}$ and by replacing each other letter by a random letter from Q.

We shall show that the combination of $\boldsymbol{shift}$, $\boldsymbol{key}$ and $\boldsymbol{s}$ is important to the security of our scheme.

3.1. The New Scheme over $\{\mathbf{0}, \mathbf{1}\}$

In this scheme $Q = \{0, 1\}$. We assume that $\boldsymbol{shift}$, $\boldsymbol{key}$ and $\boldsymbol{s}$ are from $\{0, 1\}^k$, where k is the key length. To be secure, the value k should be reasonably large. For example, the probability that an attacker could guess these secret words correctly should be small (i.e., $1/2^k$ should be small enough).

This is a challenge-and-response-based identification scheme. The machine generates challenges which are strings selected uniformly at random from Q^m.

We describe the procedure for finding the lexical-first subsequence of a string D in a source string S: starting from the first character of S, we find the first character in S that is same as the first character in D. This gives the location of the first character of the lexical-first subsequence. Then, we find the first character following the last found location in S that is same as the second character in D. This gives the location of the second character of the lexical-first subsequence. Repeating these steps, we are able to find all locations of the lexical-first subsequence of D in S, provided that such a subsequence exists. Let $SS(S, D) = 1$ if D is contained in S as a subsequence and $SS(S, D) = 0$ otherwise.

Let $\boldsymbol{q} = (q_0 \circ q_1 \ldots \circ q_{m-1})$ be a challenge from the machine. In our scheme, let $lex(\boldsymbol{q}, \boldsymbol{shift})$ denote the set of sorted positions of the lexical-first subsequence of $\boldsymbol{shift}$ in $\boldsymbol{q}$. For each integer r, let $\boldsymbol{q}(r) = q_r q_{r+1} \ldots q_{m-1} q_0 q_1 \ldots q_{r-1}$. A challenge $\boldsymbol{q}$ is valid if it satisfies: $SS(\boldsymbol{q}, \boldsymbol{shift}) = 1$, and $SS(\boldsymbol{q}(r), \boldsymbol{key}) = 1$, where $r = max(lex(\boldsymbol{q}, \boldsymbol{shift}))$.

When receiving a challenge $\boldsymbol{q}$, the user generates a response $\boldsymbol{a} = (a_0 \circ a_1 \circ \ldots \circ a_{m-1}) \in Q^m$, where a_i is chosen according to the following procedure: if i is not in $lex(\boldsymbol{q}(r), \boldsymbol{key})$, a_i is a random element chosen uniformly from Q; if i is the $(j+1)$th location in $lex(\boldsymbol{q}(r), \boldsymbol{key})$, $a_i = s_j$.

Let reply$(\boldsymbol{q})$ denote all correct responses to challenge $\boldsymbol{q}$. Here are details of our identification scheme.

Scheme `Binary-Identification`(m, n)

1. Machine M randomly selects n strings $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$ from Q^m and sends them to user U.
2. After receiving $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$, U uniformly selects a random string $\boldsymbol{r} = (r_0 \circ r_2 \ldots \circ r_{m-1}) \in Q^m$, and sends it to M.
3. M checks whether one of the challenge question strings $\boldsymbol{q}_i = \boldsymbol{b}_i$ XOR $\boldsymbol{r}$ is valid. If it is not, it goes back to 1; otherwise, suppose $\boldsymbol{q}_i$ is the first valid string. Let $\boldsymbol{q} = \boldsymbol{q}_i$. M sends $\boldsymbol{q}$ and i to U.
4. U checks whether $\boldsymbol{q} = \boldsymbol{b}_i$ XOR $\boldsymbol{r}$. If it is correct, U continues; otherwise, U quits.
5. U generates a response $\boldsymbol{a}$ to $\boldsymbol{q}$, and sends it to M.
6. M checks whether $\boldsymbol{a} \in \text{reply}(\boldsymbol{q})$. If it is, M accepts the identification of U; otherwise, M denies the identification of U.

We will prove that the probability that $\boldsymbol{q}$ is valid is high, provided that the ratio $\alpha = k/m$ is small. Hence, the probability that the machine needs to go back to step 1 in step 3 is very small.

3.2. An Example

The following is an example when $m = 12$, $n = 1$, and $k = 4$. Let $\boldsymbol{shift} = 0101$, $\boldsymbol{key} = 1101$, and $\boldsymbol{s} = 1010$.

Figure 2 shows a challenge and a response between M and U. The bars in the positions of $\boldsymbol{q}$ show the location of the lexical-first subsequence of $\boldsymbol{shift}$ followed by that of $\boldsymbol{key}$.

The challenge question proposed by the machine is:

$\bar{0}$	0	$\bar{1}$	1	$\bar{0}$	$\bar{1}$	0	$\bar{1}$	0	$\bar{1}$	$\bar{0}$	$\bar{1}$

and, a possible response generated by the user is:

1	1	0	0	1	0	1	0	0	1	1	0

Figure 2: A simplified example of the scheme based on {0,1}.

3.3. Security Analysis

Notice that our scheme is robust against the replay attack because a challenge string is XOR-ed with a random string generated by the user. Therefore, the machine has no complete control on its challenge. We will show that our scheme can prevent common subsequence statistical attack.

Let $p(m,k)$ denote the probability that a binary string $\boldsymbol{s}$ of length k occurs as a subsequence in a random binary string of length m. This is well defined because the size of $M_m(\boldsymbol{s})$ is independent of the choice of $\boldsymbol{s}$, i.e., $p(m,k) = |M_m(\boldsymbol{s})|/2^m$, for any k-character string $\boldsymbol{s}$.

With notation $p(m,k)$, the expected number of k-length subsequence strings in a randomly chosen m-length string is $\sum_1^{2^k} p(m,k) = 2^k p(m,k)$. Consequently, the expected number of the k-length strings that are common subsequence to a set of n random m-length strings is $\sum_1^{2^k} p(m,k)^n = 2^k p(m,k)^n$. The probability that a random m-length string contains both $\boldsymbol{shift}$ and $\boldsymbol{key}$ is $p(m,k)^2$. Recall both $\boldsymbol{shift}$ and $\boldsymbol{key}$ have k characters.

We now prove some lemmas that will be needed in the security analysis of our scheme.

Lemma 1 *For any k-character string $\boldsymbol{s}$ over $\{0,1\}$, for any positive integer $m \geq k$,*

$$|M_m(\boldsymbol{s})| = 2^m - \sum_{i=0}^{k-1} \binom{m}{i} = \sum_{i=k}^{m} \binom{m}{i}.$$

Proof: We prove that $|M_m(\boldsymbol{s})|$ is independent of the choice of $\boldsymbol{s}$ by an induction on m. If $\boldsymbol{s}$ has length zero, then clearly $M_m(\boldsymbol{s}) = \{0,1\}^m$. Moreover,when $m = k$, $M_m(\boldsymbol{s}) = \{\boldsymbol{s}\}$.

Inductively, we assume for each substring $\boldsymbol{s}_1$, $|M_{m-1}(\boldsymbol{s}_1)|$ is independent of the choice of $\boldsymbol{s}_1$.

Let $\boldsymbol{c}$ be a string of m characters. There are two cases:

1. the first character of $\boldsymbol{c}$ is equal to the first character of $\boldsymbol{s}$.

2. the first character of $\boldsymbol{c}$ is different from the first character of $\boldsymbol{s}$.

We can thus divide $M_m(\boldsymbol{s})$ into two subsets M_1 and M_2, where M_1 contains all strings from $M_m(\boldsymbol{s})$ whose first character is same as that of $\boldsymbol{s}$ and M_2 contains all other strings in $M_m(\boldsymbol{s})$.

Assume $\boldsymbol{s} = (s_0 \circ s_1 \ldots \circ s_{k-1})$, let $\boldsymbol{s}_1 = (s_1 \ldots \circ s_{k-1})$. We have $|M_1| = |M_{m-1}(\boldsymbol{s}_1)|$ and $|M_2| = |M_{m-1}(\boldsymbol{s})|$. It thus follows from the induction hypothesis, that $|M_m(\boldsymbol{s})| = |M_1| + |M_2|$ is independent of the choice of $\boldsymbol{s}$.

Let $s(m,k) = |M_m(\boldsymbol{s})|$ for any k-length string $\boldsymbol{s}$. We have $s(m,k) = s(m-1,k-1) + s(m-1,k)$. We can then use induction on m and k to show

$$s(m,k) = 2^m - \sum_{i=0}^{k-1} \binom{m}{i}.$$

□

Consequently, we obtain the following lemma.

Lemma 2 *For all k and $m \geq k$,*

$$p(m,k) = 1 - \frac{\sum_{i=0}^{k-1} \binom{m}{i}}{2^m}.$$

Suppose $\boldsymbol{s}$ is a binary string of length k and $\boldsymbol{c}$ is binary string of length $m-k$. For each choice of k indices $0 \leq i_1 < i_2 < \ldots < i_k < m$, we can generate a string $\boldsymbol{y}(\boldsymbol{s},\boldsymbol{c})$ from $\boldsymbol{s}$ and $\boldsymbol{c}$ such that $\boldsymbol{s}$ is a subsequence string of $\boldsymbol{y}(\boldsymbol{s},\boldsymbol{c})$ with locations given by $(i_1, i_2, \ldots, i_k)$ and $\boldsymbol{c}$ is a subsequence string of $\boldsymbol{y}(\boldsymbol{s},\boldsymbol{c})$ with locations given by other indices. Let $Tb_{\boldsymbol{s},\boldsymbol{c}}$ be the table of all $\binom{m}{k}$ m-length strings that can be generated this way. Let $Tb_{\boldsymbol{s}}$ be the table of all $\binom{m}{k}2^{m-k}$ m-length strings that can be generated from $\boldsymbol{s}$ and all $(m-k)$-length strings from $\{0,1\}^{m-k}$. Notice that some m-length strings may have multiple copies in the table. Let $\boldsymbol{k}$ be a binary string of length k. Let $Tb_{\boldsymbol{s}}(\boldsymbol{k})$ be the table of all strings from $Tb_{\boldsymbol{s}}$ that contain $\boldsymbol{k}$ as a subsequence. Let Tb be the table of all strings collection of $Tb_{\boldsymbol{s}}$ for all $\boldsymbol{s}$. Then Tb is the table of $\binom{m}{k}$ copies of $\{0,1\}^m$. Recall that, in $\{0,1\}^m$, there are $M_m(\boldsymbol{k})$ strings containing $\boldsymbol{k}$ as a subsequence string. We obtain the following lemma.

Lemma 3 *Let $\boldsymbol{s}$ be a random binary string of length k. Let $\boldsymbol{c}$ be a random binary string from $Tb_{\boldsymbol{s}}$. Then the expected number of k-length strings that are subsequence strings of $\boldsymbol{c}$ is $p(m,k) * 2^k$.*

Proof: Let $E[x]$ denote the expected value of a random variable x.

$$\begin{aligned}
& E(|\{\boldsymbol{k}|\boldsymbol{k} \text{ is a subsequence string of } \boldsymbol{c}\}|) \\
= & \frac{\sum_{\boldsymbol{s}} \sum_{\boldsymbol{k}} |Tb_{\boldsymbol{s}}(\boldsymbol{k})|}{\sum_{\boldsymbol{s}} |Tb_{\boldsymbol{s}}|} \\
= & \frac{\sum_{\boldsymbol{k}} \sum_{\boldsymbol{s}} |Tb_{\boldsymbol{s}}(\boldsymbol{k})|}{\binom{m}{k} 2^{m-k} 2^k} \\
= & \frac{\binom{m}{k} |M_m(\boldsymbol{k})| 2^k}{\binom{m}{k} 2^m} \\
= & \; p(m,k) 2^k
\end{aligned}$$

□

Lemma 4 *If $\alpha = k/m \leq 0.1$, $\binom{m}{k}/2^m < 2^{-m/2}$.*

Proof: Using Stirling formula, we have $\binom{m}{k} \approx (em/k)^k$. When $\alpha \leq 0.1$, we also have $(e/\alpha)^\alpha \leq 1.392 < 2^{1/2}$. The lemma then follows. □

We now estimate $p(m,k)$.

Theorem 5 *If $\alpha = k/m \leq 0.1$, $p(m,k) \geq 1 - k2^{-m/2}$.*

Proof: It follows directly from Lemma 4. □

We now analyze the expected number of replies that are needed to reduce the number of common subsequence strings to a constant. Recall that if a set of replies induce a unique common subsequence string, we can recover the key pattern $\boldsymbol{key}$ and secret word $\boldsymbol{s}$. Therefore, this expected number is directly related to the level of security of the identification scheme.

Theorem 6 *If $\alpha = k/m \leq 0.1$, the expected number of replies that are needed to generate a constant-sized set of k-length common subsequence strings is about $2^{m/2}$.*

Proof: By Theorem 5, if $\alpha = k/m \leq 0.1$, we have $p(m,k) \geq 1 - k/(2^{m/2})$. In addition, for n replies, the expected number of the common subsequence strings they generate is $2^k * p(m,k)^n$. Thus, $2^k * p(m,k)^n \approx 1$ implies that $n \approx 2^{m/2}$. □

With these theorems, we can estimate the size of m and k needed in a practical implementation of our scheme. We would like to choose m and k such that the probability of a random string of length m that contains $\boldsymbol{shift}$ and $\boldsymbol{key}$ as subsequence strings is close to 1. This probability is $p(m,k)^2$. So we would like to find m and k such that $p(m,k)$ is sufficiently near 1 and m is not too large. Note that $\alpha = k/m \leq 0.1$. For example, one choice is that $k = 30$ and $m = 300$. In this case, the success probability of guessing attack on $\boldsymbol{shift}$, $\boldsymbol{key}$, and $\boldsymbol{s}$ is about 2^{-60}. The expected number of replies that are needed to generate a constant-sized common subsequence string is about 2^{150}.

To avoid attack on guessing $\boldsymbol{shift}$, $\boldsymbol{key}$, and $\boldsymbol{s}$, we need k to be reasonably large which will also make m large. In the next subsection, we will show that we can reduce m and k by using a larger alphabet Q.

3.4. The Scheme Based on Alphabet Q

If the alphabet for generating the question is Q, we can get the scheme similar to the proposed scheme based on $\{0,1\}$. In this scheme, the question alphabet Q is a letter set, and the answer alphabet A is a subset of the letter set. We assign an integer index to every letter in Q according to the lexical order. We define the addition operation on Q as adding two indices mod $|Q|$, then getting the corresponding letter as the result. We can prove similar lemmas for the general alphabet Q as for {0,1}. Let $q = |Q|$ and $a = |A|$. Usually, Q and A are same alphabet.

In this scheme, a user U and a machine M agree upon the secrets $\boldsymbol{shift}$, $\boldsymbol{key} \in Q^K$, and $\boldsymbol{s} \in A^K$ in advance. This is a challenge-and-response-based identification scheme. M generates challenges which are strings selected uniformly from Q^m.

Scheme `Q-Identification`(m, n, r)

1. Machine M randomly selects n strings $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$ from Q^m and sends them to user U.
2. After receiving $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$, U uniformly selects a random string $\boldsymbol{r} = (r_0 \circ r_2 \ldots \circ r_{m-1}) \in Q^m$, and sends it to M.
3. M counts the number v of the valid challenges in strings $\boldsymbol{q}_i = \boldsymbol{b}_i + \boldsymbol{r}$. If $v \geq r$, M sends the n challenge questions $\boldsymbol{q}_i$ to U; otherwise, it goes bask to step 1.
4. U checks whether the received challenges $\boldsymbol{q}_i = \boldsymbol{b}_i + \boldsymbol{r}$. If it is correct, U continues; otherwise, U quits.
5. For all $\boldsymbol{q}_i$, U generates response $\boldsymbol{a}_i$ to $\boldsymbol{q}_i$, and sends them to M.
6. M generates a "correct set" $\boldsymbol{d} = \{i | \boldsymbol{a}_i \in \text{reply}(\boldsymbol{q}_i)\}$ for all valid challenges $\boldsymbol{q}_i$. If $|\boldsymbol{d}| \geq r$, M accepts the identification of U; otherwise, M denies the identification of U.

We will prove that the probability that there are r valid challenges in n random strings is high, if the ratio $\alpha = k/m$ is small, and r, n are properly chosen. Hence, the probability that the machine needs to go back to step 1 in step 3 is very small.

3.5. An Example

A simplified example with $m = 12$, $n = 1$ and $k = 4$ is described here to illustrate the scheme. In this example, the question alphabet is $Q = \{a - z, 0 - 9\}$, the answer alphabet is $A = \{a - e, 0 - 4\}$, and the shift pattern is $\boldsymbol{shift} = wxyz$, the key pattern is $\boldsymbol{key} = bcde$, the secret key is $\boldsymbol{s} = 1a2b$.

Figure 3 shows a challenge and a response between a machine and a user. The bars in the positions of challenge $\boldsymbol{q}$ show the location of the lexical-first subsequence of $\boldsymbol{shift}$ followed by that of $\boldsymbol{key}$.

The challenge question proposed by the machine is

$\overline{w}$	a	$\overline{x}$	b	$\overline{y}$	$\overline{z}$	g	$\overline{b}$	k	$\overline{c}$	$\overline{d}$	$\overline{e}$

and, a possible answer generated by the user is

t	1	w	a	2	b	u	p	s	v	c	z

Figure 3: A simplified example of the scheme based on Q.

3.6. Security Analysis

Notice that our scheme is robust against the replay attack for the same reason described before for the {0,1} scheme. We now show that our scheme can prevent statistical attack.

Similar to binary strings, we can show that for any $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$ of k characters, $|M_m(\boldsymbol{s}_1)| = |M_m(\boldsymbol{s}_2)|$. Let $p_q(m,k)$ denote the probability of that a string $\boldsymbol{s}$ of length k occurs as a subsequence in a random string selected from Q^m. This is well defined because the size of $M_m(\boldsymbol{s})$ is independent of the choice of $\boldsymbol{s}$, i.e., $p_q(m,k) = |M_m(\boldsymbol{s})|/q^m$ for any k-character string $\boldsymbol{s}$.

With notation $p_q(m,k)$, the expected number of k-length subsequence strings in a randomly chosen m-length string is $\sum_1^{q^k} p_q(m,k) = q^k p_q(m,k)$. Consequently, the expected number of k-length strings that are common subsequence strings to a set of n random m-length strings is $\sum_1^{q^k} p_q(m,k)^n = q^k p_q(m,k)^n$. The probability that a random m-length string contains both ***shift*** and ***key*** is $p_q(m,k)^2$. The expected number of valid challenges in n challenges is $np_q(m,k)^2$. Recall both ***shift*** and ***key*** have k characters.

The proof of the following lemma is similar to that of lemma 1.

Lemma 7 *For any k-character string $\boldsymbol{s}$ over Q, for any positive integer $m \geq k$,*

$$|M_m(\boldsymbol{s})| = q^m - \sum_{i=0}^{k-1} \binom{m}{i} (q-1)^{(m-i)}.$$

Suppose $\boldsymbol{s}$ is a string selected from Q^k and $\boldsymbol{c}$ is a string selected from Q^{m-k}. Then similar to the binary scheme, let $Tb_{\boldsymbol{s},\boldsymbol{c}}$ be the table of all $\binom{m}{k}$ m-length strings that can be generated; let $Tb_{\boldsymbol{s}}$ be the table of all $\binom{m}{k} q^{m-k}$ m-length strings that can be generated from $\boldsymbol{s}$ and all $(m-k)$-length strings from Q^{m-k}. Notice that some m-length strings may have multiple copies in the table. Let $\boldsymbol{k}$ be a string selected from Q^k. Let $Tb_{\boldsymbol{s}}(\boldsymbol{k})$ be the table of all strings from $Tb_{\boldsymbol{s}}$ that contain $\boldsymbol{k}$ as a subsequence. Similarly, we can define the table Tb, and get the following lemmas.

Lemma 8 *Let $\boldsymbol{s}$ be a random string selected from Q^k. Let $\boldsymbol{c}$ be a random string from $Tb_{\boldsymbol{s}}$. Then the expected number of k-length strings that are subsequence strings of $\boldsymbol{c}$ is $p_q(m,k) * q^k$.*

Proof: The proof is similar to the proof of lemma 3. □

Lemma 9 *If $k \leq m/q$, then*

$$\sum_{i=0}^{k} \binom{m}{i} (q-1)^{(m-i)} \leq k \binom{m}{k} (q-1)^{m-k}.$$

Proof: For any $1 \leq i \leq k$, and $k \leq m/q$, we have

$$\binom{m}{i} (q-1)^{(m-i)} \leq \binom{m}{k} (q-1)^{(m-k)}.$$

We must note that the value of $\binom{m}{k}(q-1)^{m-k}$ is very close to q^k, if $k = m/q$ and q is sufficiently large. We now estimate $p_q(m,k)$.

Lemma 10

$$\frac{\sum_{i=0}^{k} \binom{m}{i} (q-1)^{(m-i)}}{q^m} \leq \frac{k\binom{m}{k}(q-1)^{m-k}}{q^m}$$

$$= k\binom{m}{k} p^k (1-p)^{m-k} \approx ((et)^{\alpha}(1-\alpha t)^{1-\alpha})^m.$$

where $p = 1/q, \alpha = k/m, t = p/\alpha, 0 \leq \alpha \leq p.$

Proof: Using the Stirling formula $\binom{m}{k} \approx (em/k)^k$, and the previous lemma, then we get the lemma. $\square$

Let $f(\alpha,t) = (et)^\alpha(1-\alpha t)^{1-\alpha}$. Then we get $p_q(m,k) \approx 1 - f(\alpha,t)^m = 1 - 2^{m\log_2 f(\alpha,t)}$. If $f(\alpha,t)$ is small, then $p_q(m,k)$ will be close to 1. Let $g(\alpha,t) = -1/\log_2 f(\alpha,t)$, then

$$p_q(m,k) \approx 1 - 2^{-m/g(\alpha,t)}.$$

The expected number of valid challenges in n random strings is $n(1-2^{-m/g(\alpha,t)})^2$. The expected number of k-length common subsequence strings in a set of r random m-length replies is $q^k(1-2^{-m/g(\alpha,t)})^r$. For security need, $g(\alpha,t)$ should be a small constant by properly choosing α, t. For instance, $g(1/16,4) \approx 6$, $g(1/8,2) \approx 18$, and $g(1/20,2) \approx 47$. Note that the corresponding $q = 1/(\alpha t)$ is 4, 4, 10 respectively.

We now analyze the expected number of replies that are needed to reduce the number of common subsequence strings to a constant. If a set of replies induce a unique common subsequence string, then we can recover the key pattern ***key*** and the secret word ***s***.

Theorem 11 *If* $p = 1/q, k = \alpha m, \alpha t = p, \alpha < p$, *the expected number of replies that are needed to give a constant sized set of common subsequence strings is about* $k\ln(q)2^{m/g(\alpha,t)}$.

Proof: By Lemma 10, and $(1-2^{-x})^{2^x} \approx 1/e$, when x grows up to infinity. Thus $q^k * p(m,k)^n \approx 1$ implies that $n \approx k\ln(q)2^{m/g(\alpha,t)}$. $\square$

From above analysis, to prevent the common answer subsequence statistical attack, the same shift pattern or the same secret word can not be used by the user for too many times. We suggest that the times of the same pattern or the secret word be used no more than $2^{m/g(\alpha,t)}$ times. We call $2^{m/g(\alpha,t)}$ the *life cycle* of the scheme. Because if these are not used more than $2^{m/g(\alpha,t)}$ times, the number of common subsequence strings in all of the replies will be about q^k/e, which is similar to the expected trials in attack by guessing password pattern(i.e.,q^k).

For instance, let $q = 10$, $a = 4$, $m = 128$, and $k = 8$, then $\alpha = k/m = 1/16$, $t_Q = 1/(\alpha q) = 1.6$, $t_A = 1/(\alpha a) = 4$, $g(1/16,4) \approx 6$, and $g(1/16,1.6) \approx 45$. Note that $p_q(m,k) \approx 1 - 2^{-m/g(\alpha,t)}$. Then the probability that a challenge question contains the shift pattern and the password pattern is $(1-2^{-128/45})^2$, which is about 0.5. The expected number of valid challenges in n questions is about $n/2$. The times of the secret word to be used is about 2^{21}, which is the life cycle $2^{128/g(1/16,4)} \approx 2^{21}$. This life cycle of this example is acceptable for practical use.

4. Final Remarks and Open Questions

In this paper, we present a new scheme for human-machine identification. In our scheme, a user needs to memorize three keys ***shift***, ***key***, and ***s***, all with about 20–40 binary bits. We have also considered the tradeoff between the size of the alphabet Q and the length of these keys. Our scheme is provably secure against replay attacks and many statistical attacks. We have provided a tight bound on the number of common subsequence strings and have shown that this number reduces slowly as the number of challenge response pairs increases. Hence, the life-time of our scheme is reasonably long.

Our scheme can be used in the identification among computers in an intranet. It will be more efficient than many existed schemes, due to the simplicity of our scheme. This scheme can also be used to enhance the security of Kerberos system. If an attacker knows the password of a user, it is easy to get the session key. In a workstation environment, it is quite simple for an intruder to replace the *login* command with a version that records users' passwords before employing them in the Kerberos dialog. If the client crashes down after receiving the password from a user, an attacker can get the password from the disk file. If our scheme is used to identify a user to a server, the user's password will never be showed up explicitly. The client and the server compute the hash value of the challenge and response as the session key.

There are many interesting questions that we would like to have satisfiable answers. Our scheme requires a user to remember three keys. We can combine ***shift*** and ***key*** into a single key of length $2k$. But this does not reduce the number of bits that a user should remember. In addition, our scheme is based on the assumption that most human users have the ability to perform the lexical-first match between two strings. We would like to know whether these requirements are close to the minimum that can be hoped on effective human-machine identification protocols.

We would like to understand more about the tradeoff between the security level and human user's ability in memorizing secrets and in processing protocols. This tradeoff may provide an upper bound on the security level of all identification schemes/applications that involve human users, because, most often, the human user's ability is the bottleneck in the interaction between human and machines.

References

1. T. Baritaud, M. Campana, P. Chauvaud, H. Gilbert: On the Security of the Permuted Kernel Identification Scheme. *Advances in Cryptology-CRYPTO'92*, Springer-Verlag, Santa Barbara, 1992, *pp*305-312.
2. A. Fiat and A. Shamir: How to prove yourself: Practical Solutions to Identical Solutions to Identification and Signature Problems. *Advances in Cryptology-CRYPTO'86,* Springer-Verlag, Santa Brabara, USA *pp*186-194.
3. U. Feige and A. Shamir: Zero Knowledge Proofs of Knowledge in Two Rounds. *Advances in Cryptology-CRYPTO'89* , Springer-Verlag, Berlin 1990,*pp*526-545.
4. A. Fuchsberger, D. Gollman, P. Lothian, K. G Paterson and A. Sidiropoulos: Public-key Cryptography on Smart Cards. *Cryptography: Policy and Algorithms,* Springer-Verlag, Australia, 1995, *pp*250-270.
5. G. Gaskell, M. Looi: Integrating Smart Cards into Authentication Systems, Cryptography on Smart Cards. *Cryptography: Policy and Algorithms*, Springer-Verlag, Australia, 1995, *pp*270-282.
6. Joe, Kilian, S. Micali, and R. Ostrovsky: Minimum Resource Zero-Knowledge Proofs. *Advances in Cryptology-CRYPTO'89,* Springer-Verlag, Berlin 1990,*pp*545-546.
7. T. Matsumoto and H. Imai: Human Identification Through Insecure Channel. *Advances in Cryptology-EUROCRYPT'91,* Springer-Verlag, Berlin, Germany, 1991 *pp*409-421.
8. T. Okamoto: Provably Secure and Practical Identification Schemes and Corresponding Signature Schemes. *Advances in Cryptology-CRYPTO'92,* Springer-Verlag, Santa Barbara, 1992, *pp*31-54.
9. A. Shamir: An Efficient Identification Scheme Based on Permuted Kernels. *Advances in Cryptology-CRYPTO'89,* Springer-Verlag, Berlin 1990,*pp*606-609.
10. C.P. Schnorr: Efficient Identification and Signatures for Smart Cards. *Advances in Cryptology-CRYPTO'89,* Springer-Verlag, Berlin 1990, *pp*239-252.
11. C.-H. Wang, T. Hwang and J.-J. Tsai. On the Matsumoto and Imai's Human Identification Scheme. *Advances in Cryptology-EUROCRYPT'95*, Springer-Verlag, Saint-Malo, France, 1995, *pp*382-392.