On the Security of PAS (Predicate-based Authentication Service)*

Shujun Li
Department of Computer and Information Science
University Konstanz, Fach M697
Universitätsstraße 10
D-78457 Konstanz, Germany
www.hooklee.com

Hassan Jameel Asghar and Josef Pieprzyk
Center for Advanced Computing – Algorithms and Cryptography
Macquarie University
Sydney NSW 2109, Australia
{*hasghar,josef*}*@science.mq.edu.au*

Ahmad-Reza Sadeghi
System Security Group
Ruhr-University of Bochum
Universitätsstraße 150
D-44780 Bochum, Germany
ahmad.sadeghi@trust.rub.de

Roland Schmitz
Department of Computer Science and Media
Stuttgart Media University
Nobelstrasse 10
D-70569 Stuttgart, Germany
schmitz@hdm-stuttgart.de

Huaxiong Wang
Division of Mathematical Sciences
School of Physical & Mathematical Sciences
Nanyang Technological University
50 Nanyang Avenue, 639798, Singapore
hxwang@ntu.edu.sg

Abstract

Recently a new human authentication scheme called PAS (predicate-based authentication service) was proposed, which does not require the assistance of any supplementary device. The main security claim of PAS is to resist passive adversaries who can observe the whole authentication session between the human user and the remote server.

In this paper we give a detailed security analysis of PAS and show that PAS is insecure against both brute force attack and a probabilistic attack. In particular we show that the security of PAS against brute force attack was strongly overestimated. Furthermore, we introduce a probabilistic attack, which can break part of the password even with a very small number of observed authentication sessions. Although the proposed attack cannot completely break the password, it can downgrade the PAS system to a much weaker system similar to common OTP (one-time password) systems.

*This is the full edition of a paper (with the same title) published in Proc. 25th Annual Computer Security Applications Conference (ACSAC 2009) by the IEEE. This paper is available online at http://www.hooklee.com/Papers/ACSAC2009_Full.pdf, and a preprint of the published edition at http://www.hooklee.com/Papers/ACSAC2009.pdf.

1 Introduction

An important and foremost requirement of every security system is user or entity authentication. A user authentication method or protocol enables a system (the verifier) to give access to legitimate users while denying access to impersonators. Roughly speaking, user authentication methods can be divided into the following basic three categories according to how the verifier authenticates a user: 1) "what you know" – via a secret shared between the legitimate user and the verifier; 2)

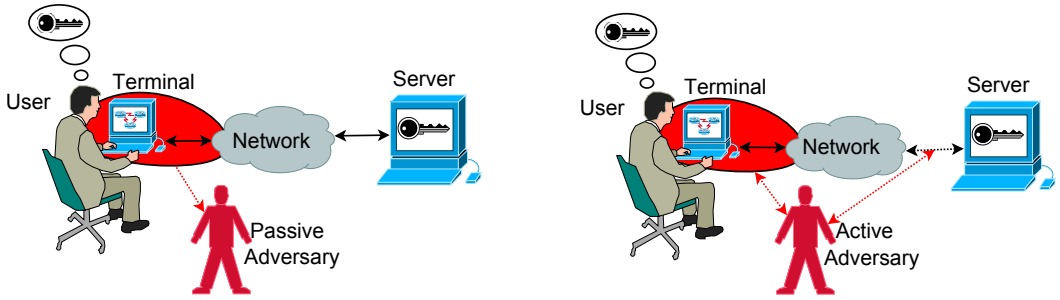

Figure 1. Adversaries in Matsumoto-Imai threat model, where the area in red shows the information source available to the adversaries, and the arrows denote directions of information flow.

"what you possess" – via a physical token the user possesses; 3) "who you are" – via an inherent characteristic of the user. Typical examples of user authentication systems belonging to the above three categories include password-based systems ("what you know"), smart card based systems ("what you possess") and biometrics-based systems ("who you are").

Different user authentication methods are designed to be secure under different threat models. One important threat model involves adversaries who can eavesdrop on or even tamper communications between the user and the verifier. Apparently, fixed passwords are not secure under this threat model, since they can be simply recorded and replayed later by an adversary to impersonate the protected identities. As possible solutions, dynamic passwords like one-time passwords (OTP) or more complicated challenge-response user authentication protocols have to be adopted. To assist human users to calculate the one-time passwords or correct responses to dynamic challenges, some special-purpose hardware/software is often a must.

In a stronger threat model described by Matsumoto & Imai in [1], it is assumed that the human user does not have access to any special-purpose hardware/software. Instead, the only resource a human user can use is his/her own brain. There are two types of adversaries in this threat model – passive and active adversaries as shown in Figure 1. Passive adversaries can observe all the user's interaction with the terminal and/or all the communications between the terminal and the remote server. In comparison, active adversaries can further modify the communications between the terminal (i.e., the user) and the remote server. A lot of practical attacks belong to or have close link to the above threat model, such as shoulder-surfing attack, key/screen-logger attack, phishing/pharming attack, malware-based attack, man-in-the-middle attack, and so forth. In the literature, the term "observer attack", "observation attack" and "peeping attack" are also used to cover all the attacks under this threat model [2].

Generally speaking, a secure user authentication system under Matsumoto-Imai threat model is a challenge-response protocol based on a secret shared between the user and the server. The user has to make correct responses to a number of challenges dynamically generated by the server to prove his/her identity. There are several design goals of such a challenge-response user authentication protocol:

1. *Usability*: the correct response to each challenge is easy for a legitimate user to calculate (from the secret and the challenge) mentally.

2. *Security against passive adversaries*: it is computationally infeasible to derive the secret or part of it from a number of observed authentication sessions (i.e., challenge-response pairs).

3. *Security against active adversaries*: it is computationally infeasible to choose some challenges to ease the derivation of the secret or part of it.

Since the 1990s there have been a number of attempts at designing user authentication systems which are secure against passive adversaries, which will be introduced in Section 2. A recent design was predicate-based authentication service (PAS) proposed by Bai et al. [3], which was designed to resist passive adversaries. In this paper, we show that the original security claims given by Bai et al. in [3] are not correct. A probabilistic attack is proposed to partially break the secret shared between the user and the server, which downgrades the PAS scheme to a much weaker authentication system.

The rest of the paper is organized as follows. Some related work against adversaries under Matsumoto-Imai threat model is introduced in the next section. Then, we briefly describe how the PAS scheme works in Section 3. A re-evaluation of security and usability of the PAS scheme is given in Section 4, and a probabilistic

attack is proposed in Section 5. The last section concludes the paper.

2 Related Work

To the best of our knowledge, the earliest attempt was made by Matsumoto and Imai after they introduced the threat model [1]. Wang et al. showed that the Matsumoto-Imai protocol was not secure enough against active adversaries [4, 5]. Wang et al. also proposed a modified scheme, but its usability is too low for common users in practice. In [6] Matsumoto proposed several new protocols based on the dot product of two vectors. According to the security analysis in [2], these dot-product-based protocols are not sufficiently secure against passive adversaries, in the sense that the secret can be revealed with a linear (in the size of the secret) number of observed authentication sessions (which was also pointed out in [7]).

In [8] Li and Teng proposed a new protocol based on lexical shifting and matching. No cryptanalysis was reported on Li-Teng protocol, but its usability is doubtful since the user has to remember three different kinds of secrets, each of which is of a considerable length.

Two protocols based on hard mathematical problems were proposed by Hopper and Blum in [7]. The main problem with Hopper-Blum protocols is again about usability: the password has to be long enough to ensure security, which makes usability relatively low. One Hopper-Blum protocol also requires the user to make intentional errors with probability η, which may not be an easy task for many common users. According to the user study on a prototype system reported in [7], the average login time is around 160 seconds, which may be too long for a practical system.

In [9], Li and Shum suggested some principles, and also two general structures of designing challenge-response protocols secure under Matsumoto-Imai threat model – Twins and Foxtail, which are based on making balanced errors and hiding direct responses to challenges, respectively. A Foxtail protocol and a graphical implementation were designed. No cryptanalysis has been reported on this work, but the usability of the graphical implementation is also questionable, since the login time is considerably long (around 3 to 4 minutes when the success rate of random guess is 2^{-20}).

Jameel et al. proposed a new image-based solution in [10] and shortly after extended it for devices with limited display in [11]. This solution is based on a hidden rule which is used to classify images into two different categories. One problem with this design is how the server can collect images in different categories, since the hidden rule may not be executable by a computer. If the classification of all images involved has to be done manually by the user, it is doubtful if the solution can offer an acceptable level of usability while at the same time maintaining a sufficient level of security. Additionally, security of this solution is based on a conjecture rather than concrete security parameters.

In [12] Weinshall proposed two new solutions based on image recognition capabilities of humans. However, Golle and Wagner showed that both solutions are insecure against SAT (satisfiability solver) attack [13]. This attack is very effective, since it requires only a small number of observed authentication sessions to be successful. Not only are the solutions not secure, but their usability is also questionable, since the user has to remember 30 – 80 pictures, which may not be an easy task for most common users.

Besides the above proposed solutions, there is also quite a lot of work aiming at the weakest type of passive adversaries – shoulder surfers [14, 15, 16, 17, 18, 19, 20, 21, 22, 23]. The main goal is to avoid password leaking from a few number of authentication sessions observed by a shoulder surfer. Since the security level is considerably relaxed, it becomes much easier to design practical solutions secure against shoulder surfers.

While most designs try to hide the password or correct responses from being observed by attackers, recently Sasamoto et al. proposed to hide part of the challenges [24]. In this specific design called UnderCover, the hidden part of the challenge is realized via a tactile channel such that the user's palm resting on a haptic device obscures any external observation. While this solution does not ask the user to bring any special-purpose hardware, the terminal equipped with the haptic device has to be trustable, which cannot be ensured in some real attacks. In addition, a passive adversary may be able to reveal some of the hidden challenges by observing the behavioral change of the user.

The main difficulty of designing a user authentication protocol secure under Matsumoto-Imai threat model is to find an acceptable balance between security and usability. Many solutions can be made secure by merely increasing the password size, but this makes the systems unusable in practice. Another noticeable difficulty is the imbalance between the human users and the potential adversaries. While human users can depend only on their brains, adversaries generally have access to more powerful computational resources such as a supercomputer or even a distributed computing platform like a botnet under his control.

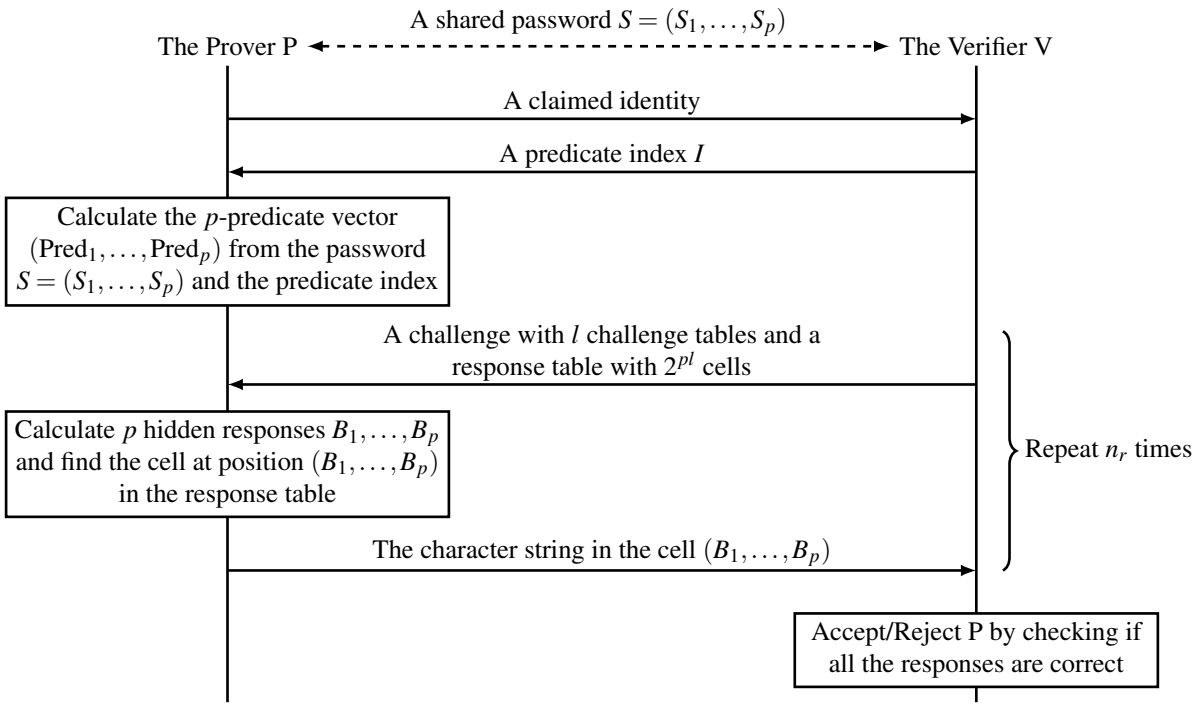

Figure 2. The authentication process of the PAS scheme.

3 Predicate-based Authentication Service

In this section, we try to keep the original notations used in the original paper, but some of them are changed to avoid potential confusion and to maintain consistency among different notations.

In the PAS scheme, the prover P (the human user) and the verifier V (the PAS server) share a *password* S composed of p *secrets* $S_1, \ldots, S_p$. Each secret S_i consists of a 2-D *secret cell index* (u_i, v_i) and a *secret word* of size len $W_i = w_i[1] \cdots w_i[len]$. The 2-D index denotes a cell at position (u_i, v_i) in an $m \times n$ 2-D grid, so $1 \le u_i \le m$ and $1 \le v_i \le n$. Each character of the secret word belongs to an alphabet $\mathbb{H}$ of size H. Since the 2-D cell index can simply be transformed to a 1-D index $c_i = (u_i - 1) \cdot n + v_i \in \{1, \ldots, M = mn\}$, in this paper we will analyze the PAS system by replacing (u_i, v_i) with the equivalent 1-D cell index $c_i \in \{1, \ldots, M\}$. That is, each secret will be represented as $S_i = (c_i, W_i) = (c_i, w_i[1] \cdots w_i[len])$. A password with parameter $p = 2$, $len = 7$, $M = 25$ looks like "(12,catchme; 25,beathim)".

Broadly speaking, the PAS scheme is a challenge-response protocol, in which the verifier V raises a number of challenges and the provers P must give correct responses to all challenges in order to pass the authentication process. To achieve security against passive adversaries, Bai et al. suggested using p "*predicates*" (instead of the password S) to make responses to challenges. The p predicates are dynamically calculated by the prover P from the secret S and a *predicate index* I, which is sent from the verifier V to the prover P at the beginning of each authentication session. The prover P calculates $\hat{I} = (I \bmod len) + 1$ and generates the p predicates as follows: $i = 1, \ldots, p$, $\text{Pred}_i = (c_i, h_i)$, where $h_i = w_i[\hat{I}]$. In this paper, we say $\text{Pred} = (\text{Pred}_i)_{i=1}^{p}$ is a *p-predicate vector*. When $p = 2$, the p-predicate vector is also called a *predicate pair*. The predicate pair derived from the password "(12,catchme; 25,beathim)" and $I = 2$ will be "(12,a; 25,e)".

Each challenge raised by the verifier V includes l challenge tables, each of which contains M cells filled with a certain number of distinct characters in $\mathbb{H}$. To ensure that each character occurs in each cell with probability 0.5, the number of characters in each cell is always $H/2$ when H is even, and is $(H-1)/2$ or $(H+1)/2$ with probability $\beta = 0.5$ when H is odd. To simplify our analysis, in this paper we assume H is even and so each cell always contains $H/2$ characters. Note that in the default setting of the PAS scheme $H = 26$. In addition to the l challenge tables, the verifier V also sends a p-dimensional response table to the prover P. Each dimension of the response table has 2^l possible values, so there are 2^{pl} cells in the response table. All

the cells are filled with 2^l distinct character strings, each of which occurs exactly in $2^{(p-1)l}$ cells.[1] See Figs. 1 and 2 in [3] for examples of the challenge and the response table.

The prover P constructs a response to each challenge based on the response table and p *hidden responses* generated from the p predicates. For the i-th predicate $\text{Pred}_i = (c_i, h_i)$, the corresponding hidden response is an l-bit integer $B_i = b_i[1] \cdots b_i[l]$, where $b_i[j] = 1$ if h_i occurs in the c_i-th cell of the j-th challenge table and $b_i[j] = 0$ otherwise. With the p hidden responses, the prover P finds the cell at the position $(B_1, \ldots, B_p)$ in the response table, and sends the character string in that cell as the response to the challenge.

A step-by-step description of the authentication process of the PAS scheme is shown in Fig. 2.

In [3], Bai et al. did not clearly mention how the predicate index I should be generated. Instead, they discussed the number of authentication sessions (denoted by t) each predicate index $\hat{I}$ can be used. The maximal number $t_{\max}$ turns out to be 1 for the default setting of the PAS scheme. This means that each possible value of $\hat{I}$ is used for one authentication session only, and the password has to be renewed after all the len possible values are exhausted. The predicate indices of the len authentication sessions may simply be chosen as $1, \ldots, len$ or a permutation of the len values. In this paper, we assume the PAS scheme runs in a "random permutation mode", in which a random permutation of $1, \ldots, len$ determines the predicate index used for each authentication session. Note that this is the most complicated (and thus the most "secure") way one can adopt to assign the len values of the predicate index to all the authentication sessions.

Bai et al. also extended the above basic PAS scheme to allow $k > 1$ cell indices in each secret S_i. In this case, the i-th secret in the password is redefined as $S_i = (c_{i,1}, \ldots, c_{i,k}, W_i)$. Accordingly, k predicate indices $I_1, \ldots, I_k$ will be sent from V to P for each authentication session. The prover P calculates the i-th predicate Pred_i as a set of k *sub-predicates* $\{\text{Pred}_{i,j}\}_{j=1}^{k}$, where $\text{Pred}_{i,j} = (c_{i,\hat{I}_{j,k}}, h_{i,j})$, $h_{i,j} = w_i[\hat{I}_{j,len}]$, $\hat{I}_{j,k} = (I_j \bmod k) + 1$ and $\hat{I}_{j,len} = (I_j \bmod len) + 1$. With this extended predicate containing k sub-predicates, the hidden response B_i of the i-th predicate is obtained as follows: the prover P first calculates k *hidden sub-responses* $B_{i,1}, \ldots, B_{i,k}$ for the k sub-predicates in the same way as in the basic PAS scheme, and then determines B_i as the bitwise OR of the k hidden sub-responses: $B_i = B_{i,1} \vee \cdots \vee B_{i,k}$. To ensure uniform distribution of B_i over $\{0, \ldots, 2^l - 1\}$, the number of distinct characters in each cell of each challenge table and the corresponding probability β should be determined by Eqs. (6) and (8) in [3], respectively.

A list of the parameters (with the default values) and notations involved in the description of the PAS scheme is given in Table 1.

In [3], the security of the PAS scheme was analyzed against three different possible attacks: brute force attack, random guess attack and SAT (satisfiablity solver) attack. Three different attack targets were checked: password, predicate, and response. By assuming each predicate index is used for t authentication sessions, the security was measured in term of the cardinality of the attack set, i.e., the size of the reduced target space, or the number of candidate targets passing all the observed authentication sessions. Table 2 shows the results reported by Bai et al. in [3]. By setting a minimal security level for each possible attack, Bai et al. also described how to get $t_{\max}$, the maximal number of authentication sessions a predicate index $\hat{I}$ can be repeatedly used. For the default setting of the basic PAS scheme, it was claimed that $t_{\max} \approx 1$ so that the same password S can be used for *at least* $t_{\max} \cdot len = 10$ times before renewal.

Bai et al. also did a usability study based on a prototype system with the default parameters and $n_r = 2, 3, 4, 5$. The average time consumed on deriving the predicates from secrets was around 35 seconds, and that for each challenge round ranged from 8.37 to 10.5 seconds. When $n_r = 5$, the total login time for one authentication session was around 84 seconds on average. A survey on the upper bound of the login time was also conducted, and more than half of the participants chose 2 minutes. We will use these statistical data to discuss the relationship between security and usability of the PAS scheme.

4 Re-Evaluating Security and Usability

First of all, we point out that the definitions of two of the three attacks in [3] are problematic. Observing Table 2, one can see there are two "NA"-s for brute force attack, and security against brute force attack is the same as security against random guess attack. In fact, according to the definitions given in [3], the brute force attack and the random guess attack are actually the same attack if the target is the password.

In our opinion, the brute force attack should be defined as exhaustively searching the whole password/predicate space $\mathbb{S}$ to determine a subspace (i.e., the so-called "attack set" according to the term of Bai et al. in [3]) $\mathbb{S}^* \subseteq \mathbb{S}$, which is composed of all can-

[1] Note that for $p = 2$, this rule actually implies a $2^l \times 2^l$ Latin square filled with 2^l distinct elements.

Table 1. List of parameters/notations used in the description of the PAS scheme.

Parameter	Description	Default
p	The number of secrets in the password	2
len	The number of characters in a secret word	10
$\mathbb{H}$	The set of all possible characters in a secret word	$\{A,\cdots,Z\}$
H	The size of $\mathbb{H}$, i.e., the number of all possible characters	26
l	The number of challenge tables in a challenge	2
$M=mn$	The number of cells in a challenge table	25
n_r	The number of challenges (rounds) in an authentication session	5
k	The number of cell indices in each secret S_i	1
	The number of sub-predicates in each predicate Pred_i	
Notation (Basic Scheme)	Description	
$S=(S_1,\ldots,S_p)$	The password shared between P and V	
$S_i=(c_i,W_i)$	The i-th secret in the password S	
$c_i\in\{1,\ldots,M\}$	The secret cell index in the i-th secret S_i	
$W_i=w_i[1]\cdots w_i[len]$	The secret word in the i-th secret S_i, where $w_i[1],\ldots,w_i[len]\in\mathbb{H}$	
$I\in\mathbb{Z}^+$	The predicate index sent from V to P	
$\hat{I}=(I \bmod len)+1$	The predicate index modulo len	
$\text{Pred}=(\text{Pred}_i)_{i=1}^{p}$	The p-predicate vector used by P in an authentication session	
$\text{Pred}_i=(c_i,h_i)$	The i-th predicate, where $h_i=w_i[\hat{I}]$	
$B_i=b_i[1]\cdots b_i[l]$	The hidden response corresponding to the i-th predicate Pred_i	
$b_i[j]=1$ (or 0)	h_i occurs (or does not occur) in the c_i-th cell of the j-th challenge table	
t	The number of authentication sessions a predicate index can be used	
Notation (Extended Scheme)	Description	
$S_i=(c_{i,1},\ldots,c_{i,k},W_i)$	The i-th secret in the password S	
$c_{i,k}\in\{1,\ldots,M\}$	The k-th secret cell index in the i-th secret S_i	
$I_1,\ldots,I_j,\in\mathbb{Z}^+$	The predicate indices sent from V to P	
$\hat{I}_{j,len}$ and $\hat{I}_{j,len}$	The j-th predicate index modulo k and len, respectively	
$\text{Pred}_i=\{\text{Pred}_{i,j}\}_{j=1}^{k}$	The i-th predicate, where $h_i=w_i[\hat{I}]$	
$\text{Pred}_{i,j}=(c_{i,\hat{I}_{j,k}},h_{i,j})$	The j-th sub-predicate in the i-th predicate, where $h_{i,j}=w_i[\hat{I}_{j,len}]$	
$B_i=B_{i,1}\vee\cdots\vee B_{i,k}$	The hidden response corresponding to the i-th predicate Pred_i	
$B_{i,j}$	The hidden sub-response corresponding to the sub-predicate $\text{Pred}_{i,j}$	

didates of the password/predicate that pass all the authentication sessions observed by a passive adversary. Apparently, the correct password/predicate used by the human prover P is always in the subspace $\mathbb{S}^*$. When $|\mathbb{S}^*|=1$ or small enough, we say the brute force attack is successful. Just as its name implies, the random guess attack should be defined as randomly guessing the correct password, predicate or response of each challenge in order to pass the authentication session. Note that in the brute force attack the goal is to (maybe partially) reveal the password, but in the random guess attack the goal is to simply impersonate a claimed identity without trying to break any target.

In [3] Bai et al. claimed that brute force attack does not take the predicates as the target, because they vary from session to session. We have a different opinion. Since the cell indices remain the same for all predicates, breaking the cell indices (as part of each predicate) may help an attacker pass a later authentication attempt with higher probability before password renewal. As a consequence, it is important to consider brute force attack targeting predicates. In fact, the first step of the probabilistic attack described in the next section of this paper is based on the brute force attack on predicates.

In the following, we re-evaluate the security of PAS, and point out that the security of the PAS scheme was over-estimated in [3]. Our new estimation is shown in Table 3. We also point out that the extended PAS scheme is not practical in terms of usability, which allows us to focus only on the basic PAS scheme in the

Table 2. The security of PAS, estimated by Bai et al. (Table 1 in [3]).

	Password	Predicate	Response
Brute Force	$M^{pk}H^{p\cdot len}$	NA	NA
Random Guess	$M^{pk}H^{p\cdot len}$	$(MH)^{pk}/(k!)^p$	2^{ln_r}
SAT	$\left(M\left(1-(1-\frac{1}{M}^N\right)^{len/k}\right)^{pk}H^{p\cdot len}$, where $N=pk(MH)^{pk}/(2^{ln_rt}(k!)^p)$	$\left(M\left(1-(1-\frac{1}{M})^N\right)^{len/k}H\right)^{pk}/(k!)^p$	NA

next section.

4.1 Security against Brute Force Attack Targeting Predicates

To facilitate the following discussion, denote the number of distinct p-predicate vectors by $N(p,k)$. In [3], the value of $N(p,k)$ was estimated to be $(MH)^{pk}/(k!)^p$. Unfortunately, this estimation is wrong. This can be easily verified when $k>1$ and $\gcd(MH,k)=1$. In this case, $(MH)^{pk}/(k!)^p$ is not an integer. To derive the correct value of $N(p,k)$, note the following fact: the number of distinct sub-predicates in the i-th predicate ranges from 1 to k. Thus, we immediately have $N(p,k)=(\binom{MH}{1}+\binom{MH}{2}+\ldots+\binom{MH}{k})^p=\binom{MH+k-1}{k}^p=(\frac{(MH+k-1)\cdots(MH)}{k!})^p\geq(MH)^{pk}/(k!)^p$.

Although Bai et al. did not over-estimate the value of $N(p,k)$, they neglected the influence of n_r and t on the size of the attack set. However, when the attacker tries to use a randomly selected incorrect p-predicate vector to calculate the response to each challenge, the probability of getting the correct response is only $1/2^l$ (under the assumption that the calculated response uniformly distributes over all the 2^l possible responses). Assuming that the responses of different challenges are independent of each other (which is so if all the challenges are generated independently by the verifier V), the probability that a randomly selected predicate will pass t observed authentication sessions will be $1/2^{ln_rt}$. Since there are one correct p-predicate vector and $\binom{MH+k-1}{k}^p-1$ incorrect ones, with t observed authentication sessions the average size of the attack set will be $1+\left(\binom{MH+k-1}{k}^p-1\right)\Big/2^{ln_rt}$, which is much smaller than the original estimation in [3]. Note that the computational complexity of the brute force attack is still $O\left(\binom{MH+k-1}{k}^p\right)$, since all the possible predicates have to be checked one by one.

4.2 Security against Brute Force Attack Targeting Password

When the target of brute force attack is the password S, Bai et al. estimated the password space as $M^{pk}H^{p\cdot len}$, which is the number of all possible p-dimension vectors $(S_1,\ldots,S_p)$. However, due to the special design of the PAS scheme, a password S can be equivalently represented as $\frac{len!}{(len-k)!}$ distinct p-predicate vectors: $\mathrm{Pred}=(\mathrm{Pred}_i)_{i=1}^{p}$, where $\frac{len!}{(len-k)!}$ is the number of all possible values of the k-tuple predicate-index vector $(\hat{I}_{1,len},\ldots,\hat{I}_{k,len})$ and

$$\mathrm{Pred}_i=\left(c_{i,\hat{I}_{1,k}},\ldots,c_{i,\hat{I}_{k,k}},w_i[\hat{I}_{1,len}]\cdots w_i[\hat{I}_{k,len}]\right).$$

Note that any change in one predicate will not influence any other predicates, so they are independent of each other. As a result, the password space can be calculated as the union of all the predicate spaces. Then, we can estimate the size of the modified password space to be $\binom{MH+k-1}{k}^p\frac{len!}{(len-k)!}$, which may be much smaller than the size of the original password space in case $len>k$ and $H>len$. For the default parameters, Table 4 shows how the ratio $r=\log_{10}\left(M^{pk}H^{p\cdot len}\Big/\left(\binom{MH+k-1}{k}^p\frac{len!}{(len-k)!}\right)\right)$ changes as k increases from 1 to $len=10$. We can see r is always much larger than 1, which means the size of the re-represented password space is always much smaller than $M^{pk}H^{p\cdot len}$. This can be best demonstrated for the basic PAS scheme. In this case, each password can be represented as len independent predicates, and the password space is reduced to $(MH)^p\cdot len$, which is smaller than $M^pH^{len\cdot p}$ as long as $H^{len\cdot(p-1)}>len$. For the default setting of the basic scheme, we can calculate that the password space is only $(MH)^p\cdot len=(25\times26)^2\cdot10\approx2^{22}$, which is too small from a cryptographic point of view. Since the cell index for each predicate is always the same, we can separately store the p cell indices $c_1,\ldots,c_p$ and the len p-character words $\{W_j^*=w_1[j]\cdots w_p[j]\}_{j=1}^{len}$. Apparently, this is just a reorganization of different parts of the password, so no extra memory is needed.

After representing the password space as the union of $\frac{len!}{(len-k)!}$ predicate spaces, we can easily obtain the size of the attack set with t observed authentication sessions for each predicate based on the result we obtained in the last subsection. That is

Table 3. Re-evaluated security of PAS against three attacks.

	Password	Predicate	Response
Brute Force / SAT	$\left(1+\left(\binom{MH+k-1}{k}^p-1\right)\Big/2^{ln_rt}\right)\frac{len!}{(len-k)!}$	$1+\left(\binom{MH+k-1}{k}^p-1\right)\Big/2^{ln_rt}$	NA
Random Guess	$\frac{1}{1/2^{ln_r}+(2^{ln_r}-1)\Big/\left(2^{ln_r}\binom{MH+k-1}{k}^p\right)} < 2^{ln_r}$		2^{ln_r}

Table 4. The ratio between the size of the re-represented password space and that of the original password space.

k	1	2	3	4	5	6	7	8	9	10
r	24.470	21.286	18.505	16.030	13.814	11.835	10.085	8.5749	7.3418	6.499

$\left(1+\left(\binom{MH+k-1}{k}^p-1\right)\Big/2^{ln_rt}\right)\frac{len!}{(len-k)!}$.

In addition, it deserves mention that a dictionary attack can narrow down the password space even further, since human users have to choose the p secret words as something that can be easily recalled. This is an inherent drawback of any human authentication scheme based on textual characters. Changing $\mathbb{H}$ to a set of graphical objects (such as a set of some small icons) may help mitigate this problem to some extent.

4.3 Security against Random Guess Attack

In random guess attack one does not need to try *all* passwords/predicates/responses, but randomly pick one from the password/predicate/response space and see if he/she can pass the authentication session. For random guess attack, there is no attack set, but we can use the reciprocal of the success probability of passing the authentication session as an equivalent metric of the security measurement.

When an attacker chooses a random response, the original estimation in [3] is correct, since there are 2^l possible responses. But the attacker can get a higher success rate if he chooses a random predicate or a random password. It is because the attacker has a chance to guess the correct predicate, which will definitely lead to the correct response. For all the other incorrect predicates, the success rate is the same as that of randomly guessing the response. Thus, the overall success rate is

$$1\cdot\frac{1}{\binom{MH+k-1}{k}^p}+\frac{1}{2^{ln_r}}\cdot\frac{\binom{MH+k-1}{k}^p-1}{\binom{MH+k-1}{k}^p}=\frac{1}{2^{ln_r}}+\frac{2^{ln_r}-1}{2^{ln_r}\binom{MH+k-1}{k}^p}>\frac{1}{2^{ln_r}}. \quad (1)$$

4.4 Security against SAT Attack

Because of the space limit, [3] does not include details of the security results on the SAT attack shown in Table 2. Fortunately, an appendix was available upon request from the first author of [3], which includes the derivation process. After checking the derivation process, we noticed that Bai et al. actually had not considered any specific features of the SAT attack. The attack was not transformed to a typical SAT problem formatted in Conjunctive Normal Form (CNF), either. As a matter of fact, since the SAT problem is NP-complete and there are many specific SAT solving algorithms, an analytic estimation on the number of solutions (i.e., the size of the attack set) and the time complexity of a specific practical SAT problem like the one from the PAS scheme is often very difficult [25].

Bai et al. actually derived the equations shown in Table 2 as follows: 1) assume the SAT solver is capable of making full use of the information leakage as we did in Section 4.1; 2) estimate the number of predicates that pass all the t authentication sessions; 3) calculate the probability that each cell index appears in all the candidate predicates, and then replace M by the number of candidate cell indices $M^* = M\left(1-(1-1/M)^N\right)^{len/k}$. Unfortunately, this derivation process does not reflect the real security level against SAT attack. As a matter of fact, the process is not only correct for the SAT attack but also for the brute force attack. If a SAT solver can eliminate an incorrect predicate, then the brute force attack can do so, too. It is well known that SAT solvers work like an optimized brute force algorithm for searching the whole solution space. The main difference from a naive brute force searching algorithm and a SAT algorithm is the time complexity of finding one or more solutions. Since Bai et al. chose the size of the attack set (i.e., the number of solutions) as the security metric, the SAT attack should have the same "performance" as

the naive brute force attack. This means that the one we obtained for the brute force attack is a more reasonable upper bound of the SAT attack. Observing our result obtained for brute force attack and the one Bai et al. derived for SAT attack (when the attack target is the password), one can easily see the former is much smaller than the latter in most cases. For instance, for the basic PAS scheme with the default parameters and $t = 1$, the latter is as high as $2^{103.3}$, but the former is only about $2^{22} \ll 2^{103.3}$.

4.5 Usability

Although Bai et al. claimed that the usability of the (basic) PAS scheme is much better than some other solutions (see the last sentence of Section 5.1 of [3]), we doubt if it is a fair comparison. The main problem is the lack of a consistent security analysis of the solutions. The existence of multiple security factors also makes it difficult to find a reasonable parameter set of each solution to compare the usability. For instance, the Cognitive Authentication Scheme (CAS) proposed in [12] has a low-complexity variant, which has relatively good usability but a lower security level according to the cryptanalysis reported in [13]. Comparing the CAS solution with the default setting of the basic PAS scheme, we have the following results:

- average login time: CAS – 1.5 minutes = 90 seconds, PAS – 84.23 seconds;
- security against random guess attack: CAS – $2^{20} \sim 2^{25}$, PAS – 2^{10};
- maximal number of authentication sessions a password can be used: CAS – less than 12, PAS – around 10 (actually less according to our analysis given in the next section).

Based on the above data, it is obvious that the basic PAS scheme is worse than the low-complexity variant of CAS in terms of both security and usability. Actually, even the above comparison is not a fair one, either, since we do not consider all security and usability factors. In our opinion, comparing performance of different human authentication systems is not an easy task without a comprehensive security and usability study of all the systems involved. But one principle is undoubtedly clear: the comparison of usability should be made for the same level of security against various kinds of attacks, and vice versa. In other words, the performance comparison should be done by considering both security and usability simultaneously.

Another problem with the basic PAS scheme is that it requires, probably, too long passwords. For the default setting, each user has to remember two cell indices and two words of length 10. In total there are 4 digits and 20 characters to be remembered. Although Bai et el. discussed several ways to create easily memorable and still strong passwords, we doubt if they indeed work in reality for average users. In [3] it was not reported if the participants in the user study had difficulties choosing their passwords and how likely it was for them to forget their passwords. According to a large-scale user study on web password habits [26], the average password length is around 6 to 9 and passwords longer than 13 characters are rare. Hence, it remains a question if 4 digits plus 20 characters are indeed usable.

In case the usability of the basic PAS scheme may be a problem, the extended PAS scheme seems even more difficult for average users to handle. Even when $k = 2$, the average login time will be at least doubled, which is about 2×84 seconds ≈ 2.8 minutes, exceeding the upper bound of more than half of the average users according to the user study reported in [3]. In addition, if the value of *len* remains the same, the number of digits and characters to be remembered will also be doubled. By using a smaller value of *len*, the memorability problem can be relaxed, but it has no obvious influence on the average login time, which does not depend on the value of *len*. Further more, we expect the error rate will also significantly increase due to the added complexity of handling more terms in each predicate.

To sum up, although we cannot definitely say if the basic PAS scheme is usable or not, it is clear that the extended PAS scheme is not usable as an acceptable solution against passive adversaries for most average users. Because of this fact, in the next section we will focus our attention mainly on the basic PAS scheme.

5 A Probabilistic Attack

Our security analysis given in the previous section has shown that security of the PAS scheme is much weaker than claimed in [3]. In Section 4.1, we also showed that the number of candidate predicates decreases exponentially as t increases. For the default setting of the basic PAS scheme, the predicate pair used can be uniquely determined with high probability when $t = 2$, since $1 + ((25 \times 26)^2 - 1)/2^{2\times5\times2} \approx 1.4029 < 2$. This leads to partial breaking of the password. To avoid information leakage from the observed responses, Bai et al. proposed to set $t_{\max} = 1$. With this setting, on average one will get $1 + ((25 \times 26)^2 - 1)/2^{2\times5} \approx 413.6$ predicate pairs for each observed session. Since the predicate pairs used for different authentication sessions are different, it seems impossible to break any part of the password when $t_{\max} = 1$.

In this section, we propose a probabilistic attack

that is still able to partially break the password even when $t_{\max} = 1$ is used. The key point is that the same set of cell indices appear in the p-predicate vectors used for different authentication sessions. This makes it possible to further exploit the correlation among different p-predicate vectors to get more information about the secret cell indices, which can then be used to further refine the set of candidate p-predicate vectors obtained from each observed authentication session. When the number of observed authentication sessions is sufficiently large, we may be able to uniquely determine the cell indices. The probabilistic nature of the attack allows us to guess the cell indices even when the number of observed authentication sessions is not high enough. After determining the cell indices, some secret characters may also be uniquely determined or there are only a few candidates left.

The success rate of the attack smoothly increases as the number of observed authentication sessions increases. For the default setting of the basic PAS scheme, experimental results show that only 7 observed authentication sessions are enough to achieve a success rate higher than 50%, which refutes the claim that the password can be used for at least 10 times before renewal. Even with only two observed authentication sessions, the success rate is not negligible – around 3.5%. The probabilistic attack is also computationally efficient. Its maximal complexity is always strictly smaller than the complexity of the brute force attack.

In the following part of this section, we describe how the attack works, and give some theoretical analyses on the probabilities involved and the computational complexity of the attack. Experimental results are given to demonstrate the feasibility of the proposed attack on the default setting of the basic PAS scheme. Finally, we show the consequence of breaking the secret cell indices is that the PAS scheme is downgraded to a challenge-response protocol working like a one-time password (OTP) system but with worse usability and security.

5.1 Description of the Attack

To simplify the description of the probabilistic attack, we show how it works for the basic PAS scheme when the attacker knows the value of len. In this case, given $\hat{t} \geq 1$ observed authentication session(s), a step-by-step description of the probabilistic attack is as follows:

- *Step 1*: For each observed authentication session, obtain a set of p-predicate vectors agreeing with all the n_r challenge-response pairs. Denote all the $\hat{t}$ sets by $\mathbb{P}_i$, $i = 1, \ldots, \hat{t}$.
- *Step 2a*: For each p-predicate vector $(\text{Pred}_1, \ldots, \text{Pred}_p)$ in $\mathbb{P}_i$, extract the cell-index part to get a p-tuple cell-index vector $(c_1, \cdots, c_p)$. All the p-tuple cell-index vectors form a new set $\mathbb{C}_i$.
- *Step 2b*: Calculate $\mathbb{C}^* = \bigcap_{i=1}^{\hat{t}} \mathbb{C}_i$.
- *Step 2c*: Use $\mathbb{C}^*$ to refine each set $\mathbb{P}_i$ and get a new set as follows: $\mathbb{P}_i^* = \{x = (c_i, h_i) | x \in \mathbb{P}_i \wedge c_i \in \mathbb{C}^*\}$.
- *Step 3a*: If $|\mathbb{C}^*| = 1$, all the p secret cell indices can be immediately determined, and thus some candidates of those secret characters in $\mathbb{P}_i^*$ corresponding to the secret cell indices can also be obtained.
- *Step 3b*: If $|\mathbb{C}^*| > 1$, count the number of times each cell-index vector occurs in $\mathbb{P}_1^*, \ldots, \mathbb{P}_{\hat{t}}^*$ and rank the cell-index vectors in order of their occurrence. All cell-index vectors that are ranked first are the candidates for the secret cell-index vector. All characters in $\mathbb{P}_1^*, \ldots, \mathbb{P}_{\hat{t}}^*$ that correspond to these candidates cell-index vectors are then the candidates for the secret characters.

In the proposed attack, Step 1 corresponds to the brute force attack targeting each p-predicate vector, and Step 2 exploits the correlation existing between different p-predicate vectors (i.e., the static cell-index vector). Step 3 has two different cases, according to the cardinality of $\mathbb{C}^*$. The ranking based strategy in Step 3b is justified by the fact that the secret cell-index vector appears to occur most frequently, since it occurs at least once while others may never occur. A more detailed analysis on this ranking probability will be discussed in Section 5.2.2. Step 3b is the main part to make the attack work in a probabilistic manner.

The proposed probabilistic attack also works for the extended PAS scheme. It can be done by simply replacing c_i with $\{c_{i,1}, \ldots, c_{i,k}\}$. However, due to the usability problem with the extended PAS scheme (recall Section 4.5), we will only discuss the basic PAS scheme in the following theoretical and experimental analysis on the performance of the probabilistic attack.

5.2 Theoretical Analysis

In this subsection, we show some theoretical analyses on Steps 3a and 3b of the attack.

5.2.1 Number of observed authentication sessions to have $|\mathbb{C}^*| = 1$

First let us investigate how many observed authentication sessions will ensure that $|\mathbb{C}^*| = 1$ happens with high probability. According to our discussion in Section 4.1, the probability that each incorrect p-predicate vector will remain in $\mathbb{P}_i$ is $1/2^{ln_r}$. Then, we can derive $\Pr[|\mathbb{P}_i| = a + 1] = \binom{N_1}{a} \left(1/2^{ln_r}\right)^a \left(1 - 1/2^{ln_r}\right)^{N_1 - a}$,

where $0 \le a \le N_1$ and $N_1 = (MH)^p - 1$. Note that the correct p-predicate vector is always in $\mathbb{P}_i$, so $|\mathbb{P}_i| \ge 1$.

Given a set $\mathbb{P}_i$ of size $a+1$, let us estimate the probability that an incorrect p-tuple cell-index vector $(c_1,\ldots,c_p)$ belongs to $\mathbb{C}_i$ under the assumption that all incorrect p-predicate vectors appear in $\mathbb{P}_i$ with equal probability. To facilitate the following discussion, denote the probability by $\rho_0(a)$. When $a > N_1 - H^p$, we can see $\rho_0(a) = 1$, since there can be a maximum of $N_1 - H^p$ p-predicate vectors with other cell-index vectors. When $a \le N_1 - H^p$, the probability is $\rho_0(a) = 1 - \binom{N_1-H^p}{a} \Big/ \binom{N_1}{a} = 1 - \prod_{i=0}^{a-1}\left(1 - \frac{H^p}{N_1-i}\right)$.

Based on the above results, for a randomly generated set $\mathbb{P}_i$ whose size is unknown, the probability that an incorrect cell-index vector $(c_1,\ldots,c_p)$ belongs to $\mathbb{C}_i$ is as follows

$$\begin{aligned} \rho &= \Pr[(c_1,\ldots,c_p) \in \mathbb{C}_i] \\ &= \sum_{a=0}^{N_1} \rho_0(a) \cdot \Pr[|\mathbb{P}_i| = a+1]. \end{aligned} \quad (2)$$

Assuming the above probability ρ does not depend on the subscript i, we get $\Pr[(c_1,\ldots,c_p) \in \mathbb{C}^*] = \prod_{i=1}^{\hat{t}} \Pr[(c_1,\ldots,c_p) \in \mathbb{C}_i] = \rho^{\hat{t}}$. Then, we can further derive the probability that $|\mathbb{C}^*| = 1$ as the probability that none of the $M^p - 1$ incorrect cell-index vectors is in $\mathbb{C}^*$: $\Pr[|\mathbb{C}^*| = 1] = \left(1-\rho^{\hat{t}}\right)^{M^p-1}$. Let $\Pr[|\mathbb{C}^*| = 1] \ge q$, we can derive the following condition: $\hat{t} \ge \left\lceil \log_\rho \left(1 - q^{\frac{1}{M^p-1}}\right) \right\rceil$.

Once the parameters of the basic PAS scheme are all given, one can immediately estimate the value of ρ and then calculate the minimal value of $\hat{t}$ corresponding to any threshold probability q. For the default parameters, we can calculate $\rho = 0.4834$. With this value of ρ, Table 5 shows the minimal value of $\hat{t}$ that can ensure $|\mathbb{C}^*| = 1$ happens with different probabilities. We can see that on average 10 observed authentication sessions are enough to uniquely determine the secret cell indices with Step 3a.

5.2.2 Ranking Probability in Step 3b

The data in Table 5 show that Step 3a is not able to effectively reduce the number of observed authentication sessions. When $q = 0.5$, we need 10 observed authentication sessions, which is the maximal number before password renewal. This does not make too much sense. Although we may also be able to break the password with 7 observed authentication sessions, the probability is a bit too low. Step 3b can help the attack work with even less than 7 observed authentication sessions, and also with a nontrivial success rate.

To make a theoretical analysis on the ranking probability problem involved in Step 3b, we first need to estimate the size of $\mathbb{P}_i^*$. Assuming the number of incorrect p-predicate vectors in $\mathbb{P}_i$ decreases with the same rate as the number of incorrect cell-index vectors in $\mathbb{C}$, i.e., $(|\mathbb{P}_i^*| - 1)/(|\mathbb{P}_i| - 1) = (|\mathbb{C}^*| - 1)/(|\mathbb{C}| - 1) = \rho^{\hat{t}}$, we can have $|\mathbb{P}_i^*| = 1 + \rho^{\hat{t}}(|\mathbb{P}_i| - 1)$. Since $E(|\mathbb{P}_i|) = 1 + N_1/2^{ln_r}$, we get $E(|\mathbb{P}_i^*|) = 1 + \rho^{\hat{t}} N_1/2^{ln_r}$.

After we have the estimation of $|\mathbb{P}_i^*|$, we want to know the probability that in $\sum_{i=1}^{\hat{t}} |\mathbb{P}_i^*|$ p-predicate vectors the correct cell-index occurs not less frequently than any incorrect one. Thus, we need to solve the following mathematical problem.

> *There are $N = M^p$ types of objects. Type-1 objects occur with probability $q_1 = (H^p - 1)/N_1$, and all other objects occur with probability $q_0 = H^p/N_1$. Randomly pick $L = \sum_{i=1}^{\hat{t}}(|\mathbb{P}_i^*| - 1)$ objects with the above probabilities and add $\hat{t}$ more type-1 object(s), what is the probability that the number of type-1 object(s) is not less than the number of objects of any other type?*

Note that $q_1 + (N-1)q_0 = 1$ for the above problem. To facilitate our discussion, denote the number of type-i objects in the L objects by $\#(O_i)$. It is not easy to get an explicit solution to the above problem. Now let us try to derive a practical lower bound of the probability. When $L \le \hat{t}$, $\#(O_i) \le L \le \hat{t} \le \#(O_1) + \hat{t}$ always holds, so $\Pr\left[\max_{i=2}^{N}(\#(O_i)) \le \#(O_1) + \hat{t}\right] = 1$. When $L \ge \hat{t} + 1$, we have the following result:

$$\begin{aligned} &\Pr\left[\max_{i=2}^{N}(\#(O_i)) \le \#(O_1) + \hat{t}\right] \\ &= 1 - \Pr\left[\exists i \in \{2,\ldots,N\}, \#(O_i) > \#(O_1) + \hat{t}\right] \\ &\ge 1 - \Pr\left[\exists i \in \{2,\ldots,N\}, \#(O_i) \ge \hat{t} + 1\right] \\ &= 1 - \Pr\left[\bigvee_{i=2}^{N} (\#(O_i) \ge \hat{t} + 1)\right] \\ &\ge 1 - \min\left(1, \sum_{i=2}^{N} \Pr\left[\#(O_i) \ge \hat{t} + 1\right]\right) \\ &= 1 - \min\left(1, (N-1)\sum_{i=\hat{t}+1}^{L} \binom{L}{i} q_0^i (1-q_0)^{L-i}\right). \end{aligned} \quad (3)$$

When $\hat{t}$ is close to 1, the above lower bound is generally equal to 0, which does not make much sense. But as $\hat{t}$ becomes larger, the lower bound quickly converges to 1. Taking the default parameters of the basic PAS scheme and assuming $L = E\left(\sum_{i=1}^{\hat{t}}(|\mathbb{P}^*| - 1)\right) = \hat{t}\rho^{\hat{t}} N_1/2^{ln_r}$, we calculated the above lower bound for $\hat{t} = 1,\ldots,10$. For each value of $\hat{t}$, 10000 random experi-

Table 5. The minimal value of $\hat{t}$ to ensure $\Pr[|\mathbb{C}^*| = 1] \geq q$, with respect to different values of q.

q	0.01	0.05	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
$\hat{t} \geq$	7	8	8	9	9	9	10	10	11	11	12

ments were also made to see how large the probabilities are. Table 6 shows the results.

Following a similar argument, we can also estimate the following inequality:

$$\begin{aligned} &\Pr[\exists i \in \{2,\ldots,N\}, \#(O_i) \geq \#(O_1) + \hat{t}] \\ &\leq \Pr[\exists i \in \{2,\ldots,N\}, \#(O_i) \geq \hat{t}] \\ &= \Pr\left[\bigvee_{i=2}^{N} (\#(O_i) \geq \hat{t})\right] \\ &\leq \min\left(1, \sum_{i=2}^{N} \Pr[\#(O_i) \geq \hat{t}]\right) \\ &= \min\left(1, (N-1)\sum_{i=\hat{t}}^{L} \binom{L}{i} q_0^i (1-q_0)^{L-i}\right). \end{aligned} \tag{4}$$

Then, assuming there are $N_{\max}$ cell-index vectors occurring most often in $\mathbb{P}_{\hat{t}}^*, \ldots, \mathbb{P}_{\hat{t}}^*$, i.e., $N_{\max}$ is the cardinality of the set $\{i | \#(O_i) = \max_{j=1}^{M^p} \#(O_j)\}$, we can get an upper bound of its mean: $E(N_{\max}) \leq 1 + (M^p - 1) \cdot \min\left(1, (N-1)\sum_{i=\hat{t}}^{L} \binom{L}{i} q_0^i (1-q_0)^{L-i}\right)$. For the default setting of the PAS scheme and $\hat{t} = 1, \ldots, 10$, Table 7 shows the theoretical upper bound and the estimated value from 10000 random experiments.

The data in Tables 6 and 7 imply that one can recover the secret cell-index vector with high probability with only 3 observed authentication sessions.

5.3 Complexity Analysis

The computational complexity of the proposed probabilistic attack is the sum of the complexity of all the three steps. The complexity of Step 1 is $\hat{t}(MH)^p$, which is the maximal number of p-predicate vectors one has to check for all the $\hat{t}$ observed authentication sessions to get $\mathbb{P}_i$. After Step 1 is finished, the average size of each $\mathbb{P}_i$ is $1 + N_1/2^{ln_r}$, so the average complexity of Step 2 is $\hat{t}\left(1 + N_1/2^{ln_r}\right)$. The complexity of Step 3a is very small, so it can be omitted. The ranking done in Step 3b has a complexity $\sum_{i=1}^{\hat{t}} |\mathbb{P}_i^*| = \hat{t}\left(1 + \rho^{\hat{t}} N_1/2^{ln_r}\right)$. The worst-case complexities of Step 2 and 3b are always less than the complexity of Step 1. As a whole, we can see the overall complexity of the attack is determined by Step 1, which has an upper bound $O(\hat{t}(MH)^p)$. For the default setting of the PAS basic scheme and $\hat{t} = 4$, the complexity is $O(\hat{t}(MH)^p) = O(2^{20.7})$.

Recalling the size of the password space of the basic PAS scheme is $len \cdot (MH)^p$, we can see the complexity of the probabilistic attack is always strictly smaller (although not very much) than that of the brute force attack since $\hat{t} < len$ always holds. Note that when $\hat{t} = len$ one does not need to break the system, since all the predicate indices have been used up and the password has already been renewed.

5.4 Experimental Results

Based on the theoretical analysis and the complexity estimation of the probabilistic attack, we can see the attack is feasible as long as $(MH)^{pk}$ is not cryptographically large. This condition is satisfied for the default setting of the PAS scheme. In fact, it has to be so, because all the parameters involved (especially p and k) cannot be too large to ensure an acceptable level of usability.

We developed a MATLAB implementation of the basic PAS scheme with $p = 2$, and tested the real performance of the proposed probabilistic attack. On a PC equipped with a 2.4GHz Intel Core2 Duo CPU, 2GB memory and 32-bit Windows Vista Business OS, one successful attack with $\hat{t}$ observed authentication sessions consumes only around $5\hat{t}$ seconds. The MATLAB code is available as a zip file at http://www.hooklee.com/Papers/Data/PAS.zip. Run SimulateAttacks to perform a certain number of simulated attacks.

The statistical results of 1000 real attacks targeting the default setting of the basic PAS scheme are shown in Table 8. It turned out that the real performance is worse than the theoretical analysis obtained in Section 5.2.2. We attribute this to the deviation of real attacks from some of the theoretical assumptions we made in the theoretical analysis in Section 5.2.2. For instance, we calculate the values in Table 6 by assuming $E(|\mathbb{P}_i^*|) = 1 + \rho^{\hat{t}} N_1/2^{ln_r}$ and $L = \hat{t}\rho^{\hat{t}} N_1/2^{ln_r}$, but in practice their values vary in a wide range around the means. Despite the mismatch between Table 8 and Table 6, we can see the success rate of breaking the secret cell-index pair and the average number of candidates follow the same pattern as the data in Table 7.

The experimental data in Table 8 clearly show that with 7 observed authentication sessions one can break the secret cell-index pair with probability greater than 50%. Even with only two observed authentication sessions, the success rate is high enough (3.5%) to threaten a considerable percentage of users.

Table 6. A theoretical lower bound of $\Pr\left[\max_{i=2}^{N}(\#(O_i)) \leq \#(O_1)+\hat{t}\right]$ and the estimated value obtained from 10000 random experiments.

$\hat{t}$	1	2	3	4	5	6	7	8	9	10
The theoretical lower bound	0	0	0.9473	0.9997	1	1	1	1	1	1
Experimental result	0.0504	0.2915	0.9604	0.9999	1	1	1	1	1	1

Table 7. A theoretical upper bound of $E(N_{\max})$ and the estimated value obtained from 10000 random experiments.

$\hat{t}$	1	2	3	4	5	6	7	8	9	10
The theoretical upper bound	625	625	607.1	6.842	1.012	1	1	1	1	1
Experimental result	3.6846	3.6184	1.7168	1.0086	1	1	1	1	1	1

Table 8. The success rate of breaking the secret cell-index vector and the number of candidates estimated from 1000 real attacks to the default setting of the basic PAS scheme.

$\hat{t}$	1	2	3	4	5	6	7	8	9	10
Success rate	0.012	0.035	0.071	0.13	0.24	0.41	0.60	0.76	0.86	0.94
Number of candidates	3.01	2.51	2.02	1.73	1.51	1.36	1.23	1.10	1.03	1.01

5.5 Consequences of the Probabilistic Attack

Note that it is impossible and unnecessary to break the whole password with the probabilistic attack, since some secret characters will never occur until the last authentication session. In fact, the main consequence of breaking the secret cell indices is the following: the password becomes a set of *len* words $\{W_j^* = w_1[j]\cdots w_p[j]\}_{j=1}^{len}$, each of which is used for exactly one authentication session. After all the *len* words $\{W_j^*\}_{j=1}^{len}$ are used up, a new password (i.e., a new set of *len* words) have to be issued to the user. Clearly, this means the PAS scheme now works essentially like a one-time password (OTP) system, where each word W_j^* is the OTP used for each authentication session and expires immediately after being used.

The degradation of the PAS scheme to an OTP-like system has several consequences. First, this fact disqualifies the PAS scheme as a better solution over common OTP systems against the targeted passive adversaries. Second, the downgraded PAS scheme is still a challenge-response protocol, which asks the user to go through the same process as in the original PAS scheme. In comparison, common OTP systems are not based on a challenge-response structure and the user is simply asked to input the dynamic password in an input box, so the usability is much better. Third, the downgraded PAS scheme offers a lower security against random guess attack. Recalling our analysis given in Section 4.3, we can derive that the success rate of randomly guessing the predicate (which is reduced to be the word W_j^*) is

$$1\cdot\frac{1}{\binom{H+k-1}{k}^p}+\frac{1}{2^{ln_r}}\cdot\frac{\binom{H+k-1}{k}^p-1}{\binom{H+k-1}{k}^p} = \frac{1}{2^{ln_r}}+\frac{2^{ln_r}-1}{2^{ln_r}\binom{H+k-1}{k}^p} \quad (5)$$

Comparing the above equation with Eq. (1), we can see the success rate becomes larger due to the lack of M in the denominator of the second term. For the default setting of the PAS scheme, Eq. (1) equals to $\frac{1}{2^{10}}+\frac{2^{10}-1}{2^{10}\times(25\times 26)^2}\approx 9.7893\times 10^{-4}$, but Eq. (5) equals to $\frac{1}{2^{10}}+\frac{2^{10}-1}{2^{10}\times 26^2}\approx 2.4544\times 10^{-3}$, nearly 2.5 times larger. To maintain the same level of security against random guess attack, the parameters have to be increased accordingly, which will make usability even worse.

6 Conclusion

In this paper, we re-evaluate the security of the predicate-based authentication service (PAS) presented by Bai et al. [3]. We show that the PAS scheme is insecure against both brute force attack and a probabilistic attack. The probabilistic attack can break part of the password even with a small number of observed authentication sessions. The breaking of part of the password downgrades the PAS scheme to an OTP-like system, thus nullifying its main advantages over common OTP systems.

It is possible to enhance security of the PAS scheme

by increasing the values of some parameters, unfortunately, which will definitely decrease the usability and make the system not useful as a practical solution. This problem about curse of usability is the main reason why it is very difficult to design a both secure and usable authentication system secure against passive adversaries who can observe all authentication sessions.

In future, we plan to work on how to better compare performance of different human authentication systems. The main focus will be on how to define an acceptable tradeoff between security and usability.

Acknowledgments

Shujun Li was supported by a fellowship from the Zukunftskolleg of the Universität Konstanz, Germany, which is part of the "Excellence Initiative" Program of the DFG (German Research Foundation). Hassan Jameel Asghar was supported by an MQRES (Macquarie University Research Excellence Scholarships) International PhD Scholarship. Josef Pieprzyk was supported by the Australia Research Council under Grant DP0987734. Ahmad-Reza Sadeghi was supported by the EU project CACE (Computer Aided Cryptography Engineering, http://www.cace-project.eu). Huaxiong Wang was supported by the National Research Foundation of Singapore under Research Grant NRF-CRP2-2007-03 and the Singapore Ministry of Education under Research Grant T206B2204.

References

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

[2] Shujun Li and Heung-Yeung Shum. Secure human-computer identification against peeping attacks (SecHCI): A survey. Technical report available online at http://www.hooklee.com/Papers/SecHCI-Survey.pdf, 2003.

[3] Xiaole Bai, Wenjun Gu, S. Chellappan, Xun Wang, Dong Xuan, and Bin Ma. PAS: predicate-based authentication services against powerful passive adversaries. In *Proc. Annual Computer Security Applications Conference (ACSAC'2008)*, pages 433–442. IEEE Computer Society, 2008.

[4] C.-H. Wang, T. Hwang, and J.-J. Tsai. On the Matsumoto and Imai's human identification scheme. In *Advances in Cryptology – EUROCRYPT'95*, volume 921 of *Lecture Notes in Computer Science*, pages 382–392. Springer, 1995.

[5] C.-H. Wang, T. Hwang, and J.-J. Tsai. On the Matsumoto and Imai human identification scheme. *IEE Proceedings - Computers and Digital Techniques*, 142(5):313–317, 1995.

[6] Tsutomu Matsumoto. Human-computer cryptography: An attempt. In *Proc. 3rd ACM Conference on Computer and Communications Security (CCS'96)*, pages 68–75. ACM, 1996.

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

[8] Xiang-Yang Li and Shang-Hua Teng. Practical human-machine identification over insecure channels. *Journal of Combinatorial Optimization*, 3(4):347–361, 1999.

[9] Shujun Li and Heung-Yeung Shum. Secure human-computer identification (interface) systems against peeping attacks: SecHCI. IACR's Cryptology ePrint Archive: Report 2005/268, August 2005.

[10] Hassan Jameel, Riaz Shaikh, Heejo Lee, and Sungyoung Lee. Human identification through image evaluation using secret predicates. In *Topics in Cryptology – CT-RSA 2007*, volume 4377 of *Lecture Notes in Computer Science*, pages 67–84. Springer, 2007.

[11] Hassan Jameel, Riaz Shaikh, Le Hung, Yuan Wei, Syed Raazi, Ngo Canh, Sungyoung Lee, Heejo Lee, Yuseung Son, and Miguel Fernandes. Image-feature based human identification protocols on limited display devices. In *Information Security Applications (WISA'2008)*, volume 5379 of *Lecture Notes in Computer Science*, pages 211–224. Springer, 2009.

[12] Daphna Weinshall. Cognitive authentication schemes safe against spyware. In *Proc. IEEE Symposium on Security and Privacy (S&P'2006)*, pages 295–300. IEEE Computer Society, 2006.

[13] Philippe Golle and David Wagner. Cryptanalysis of a cognitive authentication scheme. In *Proc. IEEE Symposium on Security and Privacy (S&P'2007)*, pages 66–70. IEEE Computer Society, 2007.

[14] Rachna Dhamija and Adrian Perrig. Déjà Vu: A user study using images for authentication. In *Proc. 9th USENIX Security Symposium*, pages 45–58. USENIX Association, 2000.

[15] Volker Roth, Kai Richter, and Rene Freidinger. A PIN-entry method resilient against shoulder surfing. In *Proc. 11th ACM Conference on Computer and Communications Security (CCS'2004)*, pages 236–245. ACM, 2004.

[16] Zhi Li, Qibin Sun, Yong Lian, and Daniele D. Giusto. An association-based graphical password design resistant to shoulder-surfing attack. In *Proceedings of the 2005 IEEE International Conference on Multimedia and Expo (ICME'2005)*, pages 245–248. IEEE, 2005.

[17] Furkan Tari, A. Ant Ozok, and Stephen H. Holden. A comparison of perceived and real shoulder-surfing risks between alphanumeric and graphical passwords. In *Proc. 2nd Symposium on Usable Privacy and Security (SOUPS'2006)*, pages 56–66. ACM, 2006.

[18] Atsushi Harada, Takeo Isarida, Tadanori Mizuno, and

Masakatsu Nishigaki. A user authentication system using schema of visual memory. In *Biologically Inspired Approaches to Advanced Information Technology (BioADIT'2006)*, volume 3853 of *Lecture Notes in Computer Science*, pages 338–345. Springer, 2006.

[19] Susan Wiedenbeck, Jim Waters, Leonardo Sobrado, and Jean-Camille Birget. Design and evaluation of a shoulder-surfing resistant graphical password scheme. In *Proc. Working Conference on Advanced Visual Interfaces (AVI'2006)*, pages 177–184. ACM, 2006.

[20] Di Lin, Paul Dunphy, Patrick Olivier, and Jeff Yan. Graphical passwords & qualitative spatial relations. In *Proc. 3rd Symposium on Usable Privacy and Security (SOUPS'2007)*, pages 161–162. ACM, 2007.

[21] Huanyu Zhao and Xiaolin Li. S3PAS: A scalable shoulder-surfing resistant textual-graphical password authentication scheme. In *Proc. 21st International Conference on Advanced Information Networking and Applications Workshops (AINAW'2007)*, volume 2, pages 467–472. IEEE Computer Society, 2007.

[22] Eiji Hayashi, Rachna Dhamija, Nicolas Christin, and Adrian Perrig. Use Your Illusion: Secure authentication usable anywhere. In *Proc. 4th Symposium on Usable Privacy and Security (SOUPS'2008)*, pages 35–45. ACM, 2008.

[23] Alexander De Luca and Bernhard Frauendienst. A privacy-respectful input method for public terminals. In *Proc. 5th Nordic Conference on Human-Computer Interaction: Building Bridges (NordiCHI'2008)*, pages 455–458. ACM, 2008.

[24] Hirokazu Sasamoto, Nicolas Christin, and Eiji Hayashi. Undercover: Authentication usable in front of prying eyes. In *Proc. 26th Annual SIGCHI Conference on Human Factors in Computing Systems (CHI'2008)*, pages 183–192. ACM, 2008.

[25] Jun Gu, Paul W. Purdom, John Franco, and Benjamin W. Wah. Algorithms for the satisfiability (SAT) problem: A survey. In *Satisfiability Problem: Theory and Applications*, volume 35 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, chapter 2, pages 19–152. American Mathematical Society, 1996.

[26] Dinei Florêncio and Cormac Herley. A large-scale study of web password habits. In *Proc. 16th International Conference on World Wide Web (WWW'2007)*, pages 657–665. ACM, 2007.