Designs, Codes and Cryptography, 38, 237–257, 2006

A Fuzzy Vault Scheme

ARI JUELS ajuels@rsasecurity.com
RSA Laboratories, 174 Middlesex Turnpike, Bedford, MA 01730, USA

MADHU SUDAN
Massachusetts Institute of Technology, 32 Vassar street, Cambridge, MA 02139, USA

Communicated by: P. Wild

Received November 27, 2002; Revised January 12, 2005; Accepted February 16, 2005

Abstract. We describe a simple and novel cryptographic construction that we refer to as a *fuzzy vault*. A player Alice may place a secret value κ in a fuzzy vault and "lock" it using a set A of elements from some public universe U. If Bob tries to "unlock" the vault using a set B of similar length, he obtains κ only if B is close to A, i.e., only if A and B overlap substantially. In constrast to previous constructions of this flavor, ours possesses the useful feature of *order invariance*, meaning that the ordering of A and B is immaterial to the functioning of the vault. As we show, our scheme enjoys provable security against a computationally unbounded attacker. Fuzzy vaults have potential application to the problem of protecting data in a number of real-world, error-prone environments. These include systems in which personal information serves to authenticate users for, e.g., the purposes of password recovery, and also to biometric authentication systems, in which readings are inherently noisy as a result of the refractory nature of image capture and processing.

Keywords: authentication, cryptography, error-correting codes

1. Introduction

Alice is a movie lover. She is looking to find someone who shares her taste in movies, but does not want to reveal information about her preferences indiscriminately to other people. One approach she might take is to compile a set A of her favorite movies and publish it in a concealed form. For instance, Alice might post to a Web newsgroup a ciphertext C_A representing an encryption of her telephone number tel_A under the set (here, key) A. In this case, if another person, say Bob, comes along with a set B of his own favorites that is identical to A, then he can decrypt C_A and obtain Alice's phone number. If Bob tries to decrypt C_A with a set different than Alice's, he will fail to obtain her telephone number. A drawback to this approach is its exactitude, or lack of error-tolerance. If Bob's interests are very similar to Alice's, e.g., if he likes two or three films that Alice doesn't, then he will not learn tel_A. It seems very likely in this case, though, that Alice would still like Bob to obtain her telephone number, as their tastes are quite similar.

In this paper, we introduce the notion of a *fuzzy vault*. This is a cryptographic construction whereby Alice can *lock* her telephone number tel_A using the set A, yielding a *vault* denoted by V_A. If Bob tries to *unlock* the vault V_A using his own set B, he will succeed provided that B overlaps largely with A. On the other hand, anyone who tries to unlock V_A with a set of favorite movies differing substantially from Alice's will fail, helping to ensure that Alice's set of favorites remains private. Thus, a fuzzy vault may be thought of as a form of error-tolerant encryption operation where keys consist of sets. Our fuzzy vault proposal has two important features that distinguish it over similar, prior work. First, the sets A and B may be arbitrarily ordered, i.e., true sets rather than sequences. Second, in contrast to previous work, we are able to prove information-theoretic security bounds over some natural non-uniform distributions on the set A.

Error-tolerant cryptographic algorithms are useful in many circumstances in which security depends on human factors, and thus exactitude represents a drawback. We offer just a few examples here, all of which might benefit from use of our fuzzy vault scheme:

1. *Privacy-protected matching*: As an extension of our movie lover's example above, we might consider a business scenario. Asco Corp. is looking to sell routers possessing a set $A = \{a_1, a_2, \ldots, a_k\}$ of specifications. It might publish a fuzzy vault V_A with its identity κ, locked under A. If Bisco Corp. is looking for routers with a set B of similar specifications, then it will be able to open the vault. Anyone who tries to unlock the vault with a dissimilar set will not learn κ. (We address this idea in detail later in the paper, and decribe an important security enhancement using on-line throttling mechanisms.)

2. *Personal entropy systems*: Proposed by Ellison et al. [12], this is a system that enables users to recover passwords by answering a series of questions. In recognition of the unreliability of human memory, the system permits users to answer some of these questions incorrectly. A serious vulnerability in this system is exposed in [5], who show more broadly that the underlying hardness assumption is weak. Our fuzzy vault scheme offers an alternative implementation that is provably secure in an information-theoretic sense and that may involve use of sets, and not just fixed-order answers.

3. *Biometrics*: Alice authenticates to her server using fingerprint information. Her system administrator wishes to store her fingerprint on the

server or, more precisely, a set A of features characterizing her fingerprint. (Such sets are known as biometric *templates*.) If an attacker breaks into the server, however, Alice does not want her template A compromised. An additional complication is that biometric systems are error-prone: When Alice tries to authenticate herself, her fingerprint reader is likely to produce a template A' that is similar to, but not identical to A (with bit errors and random permutation and erasure of elements). Alice might store a PIN locked in a fuzzy vault on a set A of features describing her fingerprint, thereby achieving both error-tolerance and privacy. Note that order-invariance is critical here. It is usually not possible to impose an order effectively on biometric features because of the problem of erasures. For this reason, previous schemes like that of Juels and Wattenberg [19] described below are unlikely to work well for this problem.

1.1. Previous Work

A somewhat less naïve approach to a fuzzy vault construction than straightforward encryption might be achieved through use of Shamir secret sharing techniques [27]. Alice partitions her secret value κ into shares $s_1, s_2, \ldots, s_n$, and encrypt these shares respectively under each of the elements $a_1, a_2, \ldots, a_n$ in her set A. This would yield a set of ciphertexts $e_1, e_2, \ldots, e_n$. Given use of a (t, n)-secret sharing scheme, Bob would only need to decrypt t shares successfully in order to unlock Alice's secret κ. The problem with this approach is twofold. First, suppose that Bob's set B consists of elements $b_1, b_2, \ldots, b_n$. Because A and B are unordered sets, Bob has no way of knowing which of the ciphertexts e_i to try to decrypt with a given set element b_j. Even if Bob tries all n^2 possible decryption operations, i.e., tries to decrypt each e_i with each b_j, there is a second problem: He still does not know which decryptions were successful. Straightforward mechanisms to reveal this information to Bob, e.g., a checksum to indicate a correct plaintext, can leak substantial information about A. Indeed, this may be regarded as the source of the weakness in, e.g., the Ellison et al. construction [12]. It is also possible for Bob to try to deduce by means of a brute-force search which elements of B do not overlap with those of A. This strategy is inflexible and likely to be prohibitively slow in many practical scenarios, as the computational requirements grow exponentially in the size of $|A \cap B|$.

Another idea that does not work well is that of imposing a common ordering on the sets A and B and then using a fuzzy vault construction or similar technique that does not have the property of order invariance, e.g., [19]. This approach fails because a small difference between sets can produce large differences in an ordered element-by-element comparison. Suppose, for example, that A and B again represent respective lists of Alice and Bob's favorite movies. If Alice and Bob's favorites include all Oscar winners, except that Alice does not like *Antonia's Line*, then a movie-by-movie comparison of these lists in alphabetical order will yield almost no matches, while in fact A and B overlap considerably. This problem also applies to attempts to impose ordering on features in biometric systems.

To overcome these difficulties, we invoke *error-correcting codes* as the basis for our construction. Given the strong technical and historical affinities between the two, it is not surprising that error-correcting codes appear in many areas of cryptography. Applications include public-key cryptography (via the well known McEliece cryptosystem) [22], identification schemes [30], digital signature schemes [1], and cryptanalytic techniques [16], just to name a few examples. This previous work uniting cryptography and error-correcting codes has not taken tolerance of error in messages or established keys as a goal. Rather, the objective has been to exploit the error-correction capabilities or the hardness properties of problems of selected codes in order to build cryptographic primitives of a standard nature, i.e., primitives for use in environments in which data is generally not subject to corruption. For example, in the McEliece cryptosystem, the addition of noise during the encryption process serves the desired function of concealing a plaintext. Error-correcting codes also play an important rôle in non-standard cryptographic models. They serve to compensate for "dark counts" and other apparatus faults in quantum cryptographic key distribution protocols (see, e.g., [2]) and play rôles in oblivious transfer over both quantum [3,8] and noisy channels (see, e.g., [7]). A central feature of all of these cryptographic system designs, in both standard and non-standard models, is the aim of "exactness". In particular, the goal has been to manipulate messages with bit-for-bit integrity or to establish a key between two players in which all bits are identical.

Exactness in cryptographic systems is obviously of critical importance in many situations. Permitting an attacker to alter even a single bit in a sensitive message—such as a funds transfer or patient record—can be quite dangerous. At the interface between cryptographic systems and their human users, however, the ability to achieve exactitude breaks down. Human beings are rather unobliging in this regard: They misremember private information, make typos in passwords, and present fingers to fingerprint readers a slightly different way each time. Nor is the inexact nature of human participation in computing confined strictly to error. Programs that answer Web queries or match profiles of prospective business partners, for example, must handle high levels of uncertainty and imprecision.

With these issues in mind, some recent research efforts in cryptography and data security have aimed at accommodating what might be dubbed "fuzziness" in user input. One example is provided by graphical password systems, as in [18], where a user sketches a secret picture in lieu of entering a text password. Users generally sketch a given picture a slightly different way each time. Thus graphical password systems must be tolerant of error, while not sacrificing security. Another example is the "personal entropy" proposal of Ellison et al.[1] That system enables users to recover passwords by answering a series of questions [12]. In recognition of the unreliability of human memory, the system permits users to answer some of these questions incorrectly. The investigation of "fuzzy" cryptographic systems in this vein was initiated in [9,10], work that incorporated error-correcting codes into a cryptographic scheme with the stated aim of achieving security in biometric applications.

The starting point for our fuzzy vault construction in this paper is the *fuzzy commitment* scheme of Juels and Wattenberg [19], which may be viewed as an extention of and improvement over [9,10] in terms of the level of error-tolerance it achieves with provable security guarantees. Fuzzy commitment is a cryptographic primitive whereby a user commits to a secret value κ under a key x. The user may decommit using any key x' that is "close" to x under some suitable metric, such as Hamming distance. An attacker without any knowledge of x, however, cannot feasibly decommit κ. When applied to a biometric system, for example, an enrolled fingerprint template might be viewed as a key x. The user tries to authenticate using another, slightly different image of the same finger, which we may denote by x'. Authentication is successful if and only if x' is "close" to x.

As the fuzzy commitment scheme in [19] is a close conceptual antecedent to our own, it is worth briefly sketching the details. Let F_q be a field, and C be the set of codewords for some error-correcting code; assume that codewords lie in ${F_q}^n$. To commit to a value $x \in {F_q}^n$, the user selects a codeword c uniformly at random from C and computes an offset of the form $\delta = c - x \in {F_q}^n$, i.e., the difference over individual field elements. The commitment then consists of the pair (δ, y), where $y = h(c)$ for some suitable one-way function h. To decommit using key x', the user computes $\delta + x'$ and, if possible, decodes to the nearest codeword c'. The decommitment is successful iff $h(c') = y$.

The construction in [19] has the advantageous features of conceptual simplicity and the ability to make use of any underlying error-correcting code. Moreover, provided that x is drawn uniformly at random from ${F_q}^n$, the scheme enjoys rigorously proveable security linear in the cardinality of C. Suppose that the attacker gains no information about c or x from y, as would be the case under a random oracle assumption on h given sufficiently large security parameters. It is easy to see then that the task of the attacker is to guess c uniformly over C. A similar, less resilient antecedent scheme is proposed in [9, 10], while another system with similar goals but no rigorously provable security characteristics is proposed in [28, 29].

Note that if the value $h(c)$ is removed from the Juels and Wattenberg scheme, i.e., if we no longer think of it as a commitment scheme, then we obtain a kind of fuzzy vault in which the vault itself is equal to δ. If x is uniformly distributed, then the scheme enjoys easily provable information-theoretic security, i.e., security against a computationally unbounded attacker (also proportional to the cardinality of C). Like our own fuzzy vault construction, this one can also be applied to any of the three practical scenarios described above, i.e., privacy-protected matching, personal entropy systems, or biometrics.

As a fuzzy vault variant, though, the scheme of Juels and Wattenberg has two shortcomings. First, while it tolerates some number of errors in the information symbols in x, it does not tolerate substantial re-ordering of these symbols. Given that translation and rotation errors are common in, e.g., biometric systems, it is reasonable to expect that the image x' may consist of a permutation of symbols in x. The property of *order-invariance* is thus likely to be desirable in a fuzzy commitment scheme. A second shortcoming of [19] is the difficulty of proving rigorous results about security over non-uniform distributions. Our proposed scheme addresses these two shortcomings, and may be thought of as an order-invariant version of the Juels–Wattenberg scheme.

The present work has appeared previously in the form of a one-page abstract [20].

1.2. Our Scheme

Like the scheme of Juels and Wattenberg, ours is conceptually simple, and can be implemented using any underlying error-correcting code (although we focus on Reed–Solomon codes in our exposition here). While possessing the advantages of order-invariance and easier analysis on non-uniform distributions, our scheme does have a couple of drawbacks that are important to note from the outset. First, it typically has substantially greater—though still quite practical—memory requirements than the Juels–Wattenberg scheme. Second, it is somewhat less flexible in terms of available parameter choices at a given security level, as we shall see.

Let us briefly sketch the intuition behind our scheme. Suppose that Alice aims to lock a secret κ under set A. She selects a polynomial p in a single variable x such that p encodes κ in some way (e.g., has an embedding of κ in its coefficients). Treating the elements of A as distinct x-coordinate values, she computes evaluations of p on the elements of A. We may think of Alice as projecting the elements of A onto points lying on the polynomial p. Alice then creates a number of random *chaff* points that do not lie on p, i.e., points that constitute random noise. The entire collection of points, both those that lie on p and the chaff points, together constitute a commitment of p (that is, κ). Call this collection of points R. The set A may be viewed as identifying those points in R that lie on p, and thus specifying p (and κ). As random noise, the chaff points have the effect of concealing p from an attacker. They provide the security of the scheme.

Suppose now that Bob wishes to unlock κ by means of a set B. If B overlaps substantially with A, then B identifies many points in R that lie on p, so that Bob is able to recover a set of points that is largely correct, but perhaps contains a small amount of noise. Using error correction, he is able to reconstruct p exactly, and thereby κ. If B does not overlap substantially with A, then it is infeasible for Bob to learn κ, because of the presence of many chaff points. (If B overlaps "somewhat", then he may still be able to recover κ. The gap between feasible recovery and infeasible is fairly small, however, as we discuss below.) We present details and analysis in what follows.

The hardness of our scheme is based on the *polynomial reconstruction* problem, a special case of the Reed–Solomon list decoding problem [5]. Other schemes making use of this problem include, for example, the scheme proposed by Monrose, Reiter, and Wetzel for hardening passwords using keystroke data [23]. An important difference between our scheme and previous ones of this flavor is our range of parameter choices. The [23] scheme bases its security on the computational hardness of small polynomial reconstruction instances, while we select parameters enabling us to achieve information theoretic security guarantees for the same problem.

1.3. Organization

We sketch protocol and security definitions for our scheme in Section 2. In Section 3, we present protocol details for our fuzzy vault scheme. We offer security analysis with proofs in Section 4. We conclude with some discussion around the practical application of fuzzy vaults in Section 5, and conclude briefly in Section 6 with some remarks on further research.

2. Definitions and Background

We work over a field F_q and a universe $\mathcal{U}$; for convenience, we assume in our exposition that $\mathcal{U} = F_q$, although this need not be the case in general. Our aim is to lock a secret value $\kappa \in {F_q}^k$ under a secret set $A \in \mathcal{U}^t = {F_q}^t$, for protocol parameters k and t. We consider a fuzzy vault algorithm Lock that takes as input a secret κ and set A and outputs a vault $V_A \in {F_q}^r$ for some security parameter r. The algorithm Lock may be (and for our purposes will be) probabilistic.

A corresponding decryption algorithm Unlock takes as input a vault $V_A \in {F_q}^r$ and a decryption set $B \in \mathcal{U}^t$. The output of this algorithm is a plaintext value $\kappa' \in {F_q}^k$, or else '*null*' if the algorithm is unable to extract a plaintext.

Our goal is to come up with a pair of vault locking/unlocking algorithms Lock/Unlock that allows reconstruction of the plaintext κ when the decryption set B is close to the encryption set A. At the same time, we want the vault V_A not to reveal κ. Recall from above that we are interested in algorithms that are order invariant. In other words, the ordering on the sets A and B should have no real impact on the locking and unlocking procedures.

2.1. Requirements

The next three definitions formalize the requirements of a good pair (Lock, Unlock) of algorithms for our fuzzy vault scheme. We say that a probability is *negligible* if it is asymptotically smaller than any positive polynomial in t and r. We say that a probability is *overwhelming* if it is larger than $1-\zeta$ for some negligible quantity ζ. Our first definition outlines the completeness condition for our algorithms, i.e., what should happen when the players are honest.

Definition 1. An locking/unlocking pair (Lock, Unlock) with parameter set (k, t, r) is *complete* with ϵ-fuzziness if the following holds. For every $\kappa \in {F_q}^k$ and every pair of sets $A, B \in \mathcal{U}^t$ such that $|A - B| \leq \epsilon$ for an integer ϵ, it is the case that Unlock$(B, \text{Lock}(A, \kappa)) = \kappa$ with overwhelming probability.

We now formalize the security, and in particular the soundness of the algorithmic pair (Lock, Unlock) in an information-theoretic sense. Assume that A is selected according to some potentially non-uniform distribution d. We seek to characterize the ability of an attacker with unlimited computational power to determine κ from Lock(A, κ). We assume that this attacker is given knowledge of a uniformly random δ-fraction of A, i.e., a uniform random subset A' of at most δt elements in A (where we assume δt to be an integer). This assumption that the adversary has knowledge of part of the secret key A is slightly unorthodox. In a "fuzzy" system, however, it is natural to consider such notions of partial adversarial knowledge, as highlighted in our examples below. Of course, other security assumptions are possible.

We characterize security in terms of the following experiment with a computationally unbounded adversary Adv for a given parameter set. This adversary Adv takes as input a set of δt elements of A, the parameters t and k, and a vault V_A on A, and outputs a guess at κ. Formally, Adv is an algorithm Adv: $\mathcal{U}^{\delta t} \times Z^2 \times {F_q}^r \rightarrow {F_q}^k$ with no bound on computational complexity. Let $\in_d$ denote selection from probability distribution d, and $\in_U$ denote uniform random selection. Here, and in all analysis that follows, we assume that κ is generated uniformly at random, as κ is typically used as a key for some independent ciphertext or cryptographic protocol. Let $\{A\}_i$ denote the set of subsets of A of cardinality i. The experiment is as follows.

```
**Experiment** Attack$(\text{Lock}, Adv)$
  $\kappa \in_U {F_q}^k$;  $A \in_d \mathcal{U}^t$;  $A' \in_U \{A\}_{\delta t}$;
  if$Adv(A', t, k, \text{Lock}(A, \kappa)) = \kappa$
      Output$'1'$;
  else
      Output$'0'$;
```

This leads to the following definition.

Definition 2. A locking/unlocking pair (Lock, Unlock) is information theoretically secure with parameter pair (δ, μ) if pr[Attack$(\text{Lock}, Adv) = 1] \leq \mu$ for any computationally unbounded adversary Adv.

Let d' be the probability distribution d restricted to sets A such that $A' \subset A$. Observe that given vault V_A, the best strategy a (computationally unbounded) adversary can adopt is to output a plaintext κ' such that the expression

$$w(\kappa', V_A) = \text{pr}_{A \in_{d'} \mathcal{U}^t}[\text{Lock}(A, \kappa') = V_A]$$

is maximized. For a given vault $V_A = \text{Lock}(A, \kappa)$, the probability of success of this strategy is easily seen to be $w(\kappa, V_A) / \sum_{\kappa' \in {F_q}^k} w(\kappa', V_A)$.

Remark. Note that our definition of information theoretic security does not necessarily imply that the secret κ is *fully* protected in an information theoretically secure manner. In particular, we may have mutual information $I(\text{Lock}, \kappa) > 0$. This is to say that our scheme may offer information theoretic hiding of κ over a set of possible values smaller than ${F_q}^k$.

2.2. Reed–Solomon Codes

It is possible to construct a fuzzy vault scheme based on essentially any type of linear error-correcting code. To sharpen our conceptual focus and analysis, however, we restrict our attention to Reed–Solomon (R–S) codes. We are interested primarily in (k, t)-codes, i.e., those in which codewords consist of t information symbols, i.e., field elements, where $q \geq t$. Each codeword corresponds to a unique polynomial p of degree less than k over F_q; thus there are q^k codewords in total. For example, when q is prime and thus F_q is the field of residue classes $\bmod q$, we may simply embody an R–S code as the sequence $\{y_1 = p(1), y_2 = p(2), \ldots, y_t = p(t)\}$, where $1, 2, \ldots, t$.

If $t > k$, then a codeword may be seen to contain some redundancy. The presence of such redundancy is what permits the code to be used for error correction. Suppose that $c' = \{y'_1, y'_2, \ldots, y'_t\}$ is a *corruption* of the codeword $c = \{y_1, y_2, \ldots, y_t\}$. In other words, we have $y'_i \neq y_i$ for some ϵ-fraction of the information symbols in c', for $0 \leq \epsilon \leq 1$. Provided that ϵ is small enough, the redundancy of the code is such that given just the corrupted codeword c', we can recover the original codeword c. For this, we use a *decoding algorithm* that we denote by RSDECODE. The algorithm RSDECODE takes c' as input, and provided that too much corruption has not occurred, outputs c.

The most common application of a Reed–Solomon or other error-correcting code is to message transmission over a noisy channel. For this, the procedure is as follows. The sender takes a message $\kappa \in {F_q}^k$ and encodes it as a polynomial of degree at most k. The sender computes the corresponding codeword c and transmits it over a noisy channel. The noise on the channel causes a corrupted codeword c' to be obtained by the receiver. The receiver applies RSDECODE to c', obtains c, and recovers the original polynomial p and thus the message κ. As we shall see, in our scheme we may think of the noise on the channel as arising from differences between the sets A and B. By guessing A inaccurately, Bob introduces noise into the channel transmitting κ. (In contrast, the fuzzy commitment scheme in [19] never actually makes explicit use of the message space.)

2.3. Our Special Use of Reed–Solomon Codes

For our constructions, it is convenient to consider a generalization of Reed–Solomon codes. We think of a codeword as consisting of an evaluation of a polynomial p over *any* set of t distinct points in F_q. In other words, we think of a codeword as consisting of a set of pairs $\{(x_i, y_i)\}_{i=1}^t$, where $x_i \in F_q$, all of the x_i are distinct, and $y_i = p(x_i)$. (We are considering a punctured extended Reed–Solomon code.)

In this generalized view, the decoding algorithm RSDECODE takes as input a collection of points which are presumed to lie preponderantly on a single polynomial of pre-specified degree at most $k-1$. The RSDECODE algorithm, if successful, outputs a polynomial p intersecting the large majority of input points.[2] Otherwise, the algorithm outputs '*null*'. This will happen, for instance, if no polynomial of the right degree matches the inputs adequately, or if computation of such a polynomial is too hard because of too much corruption of the associated codeword. The following are parameter specifics for the algorithm RSDECODE.

Public parameters: A field F_q.

Input: A degree parameter $k \leq q$ and a set of points $Q = \{(x_i, y_i)\}_{i=1}^t$ such that $x_i, y_i \in F_q$ for $1 \leq i \leq t$.

Output: A polynomial p of degree less than k over F_q, or else '*null*'. We write RSDECODE(k, Q) to denote the output on inputs k and Q.

For our (practical) purposes, the best choice for RSDECODE is generally the classical algorithm of Peterson–Berlekamp–Massey [4,21,25]. This algorithm decodes successfully if at least $\frac{k+t}{2}$ points in Q share a common polynomial. The best version of RSDECODE to date, i.e., the one most likely to recover p successfully, is that of Guruswami and Sudan [15]. This algorithm successfully determines p provided that the number of points in Q that lie on p is at least $\sqrt{kt}$. Our preference for the classical algorithm is based on the fact that this algorithm is in general much more efficient than the Guruswami–Sudan, and has the advantage of being well studied and widely implemented. Moreover, for many of the parameter choices we are likely to encounter in practice, $\frac{k+t}{2}$ is fairly close to $\sqrt{kt}$.

3. The Fuzzy Vault Algorithms

We are now ready to detail our locking and unlocking algorithms for our fuzzy vault scheme. We first present the algorithm LOCK. The basic idea here is to create a generalized Reed–Solomon codeword representing the secret κ (as a corresponding polynomial p). This codeword is computed over x-coordinates corresponding to elements in the set A. To conceal the codeword, we add chaff points, i.e., random noise in the form of random (x_i, y_i) pairs.

In our exposition here, we assume some straightforward, publicly agreed-upon method for representing the secret κ as a polynomial (e.g., taking the information symbols in κ to be the coefficients of the polynomial). We simply write $p \leftarrow \kappa$ to represent this conversion. We let $\in_U$ denote uniformly random selection from a set.

Public parameters: A field F_q, a Reed–Solomon decoding algorithm RSDECODE.

Input: Parameters k, t, and r such that $k \leq t \leq r \leq q$. A secret $\kappa \in {F_q}^k$. A set $A = \{a_i\}_{i=1}^t$, where $a_i \in F_q$, a_i's are distinct.

Output: A set R of points $\{(x_i, y_i)\}_{i=1}^r$ such that $x_i, y_i \in F_q$ and x_i's are distinct.

Algorithm LOCK

```
X, R ← φ;
p ← κ;
for i = 1 to t do
        (x_i, y_i) ← (a_i, p(a_i));
        X ← X ∪ {x_i};
        R ← R ∪ {(x_i, y_i)};
for i = t+1 to r do
        x_i ∈_U F_q \ X;
        X ← X ∪ {x_i};
        y_i ∈_U F_q \ {p(x_i)};
        R ← R ∪ {(x_i, y_i)};
output R;
```

So as not to leak information about the order in which the x_i are chosen, the set R may be output in a pre-determined order, e.g., points in order of ascending x-coordinates, or else in a random order. Note that chaff points in LOCK are selected so as to intersect neither the set A nor the polynomial p. This is for technical reasons, namely to simplify our security proofs. We refer to the set R and the parameter triple (k, t, r) together as a fuzzy vault, denoted by V_A.

As explained above, to unlock a vault V_A created by Alice, Bob tries to determine the codeword that encodes the secret κ. Recall that the set A specifies the x-coordinates of "correct" points in R, i.e., those that lie on the polynomial p. Thus, if B is close to A, then B will identify a large majority of these "correct' points. Any divergence between B and A will introduce a certain amount of error. Provided that there is sufficient overlap, however, this noise may be removed by means of a Reed–Solomon decoding algorithm. We write $\kappa' \leftarrow p$ to denote conversion of a polynomial of degree at most k to a secret in ${F_q}^k$, i.e., the reverse of the procedure employed in LOCK. We let $(x_i, y_i) \overset{(b_i,\circ)}{\longleftarrow} R$ denote projection of R onto the x-coordinate b_i. By "projection," we mean that if there is a pair $(b_i, y) \in R$ for any y, then $(x_i, y_i) = (b_i, y)$; otherwise a null element is assigned to the pair (x_i, y_i). Our unlocking algorithm is now as follows.

Public parameters: A field F_q, a Reed–Solomon decoding algorithm RSDECODE.

Input: A fuzzy vault V_A comprising a parameter triple (k, t, r) such that $k \le t \le r \le q$ and a set R of points $\{(x_i, y_i)\}_{i=1}^{r}$ such that $x_i, y_i \in F_q$. A set $B = \{b_i\}_{i=1}^{t}$, where $b_i \in F_q$.

Output: A value $\kappa' \in {F_q}^k \cup \{\text{'}null\text{'}\}$.

Algorithm Unlock

```
$Q \leftarrow \phi$;
for $i = 1$ to $t$ do
        $(x_i, y_i) \overset{(b_i, \circ)}{\longleftarrow} R$;
        $Q \leftarrow Q \cup \{(x_i, y_i)\}$;
$\kappa' \leftarrow \text{RSDECODE}(k, Q)$;
output $\kappa'$;
```

If the final decoding operation is successful, then the algorithm outputs a secret κ' which should be equal to κ if the set B is close to the original set A. If the decoding operation fails, then the algorithm outputs '*null*'.

By employing the Peterson–Berlekamp–Massey algorithm in the obvious manner as the underlying error-correction in RSDecode, enabling correction of $\frac{t-k}{2}$ errors, we obtain the following proposition characterizing the completeness of our fuzzy vault scheme.

Proposition 3. *Given use of the Peterson–Berlekamp–Massey algorithm for* RSDecode, *the algorithm pair* (Lock, Unlock) *above with parameter triple* (k, t, r) *is complete with* $(\frac{t-k}{2})$*-fuzziness.*

As an example of how the above algorithms might be applied, we briefly consider a parameterization of k and t in what we call the movie lover's problem, i.e., the problem described above in which Alice is seeking someone with similar taste in movies. We defer discussion of security parameters for the next section.

Example 1 (The movie lover's problem). *Let us consider the movie lover's problem with a total set of* 10^4 *titles in which Alice selects a set A of* $t = 22$ *different favorites.*[3] *We might choose* $k = 14$. *Since* $\frac{k+t}{2} = 18$, *another movie lover with a set B of 22 favorite titles will be able to decrypt the vault via the Peterson–Berlekamp–Massey algorithm provided that the original set A and the new set B intersect on at least 18 titles. Notice that for this choice of parameters, it is feasible to compute all possible subsets of size* 18 *from the set of size* 22, *and try interpolating from each subset. This would result, however, in an average of 3657.5 trials, while the cost of one decoding step is easily within an order of magnitude of one interpolation step. Thus the use of* RSdecode *speeds up*[4] *the decommitment step by at least a factor of 300.*

4. Security

The security of our fuzzy vault construction depends on the number of chaff points $r-t$ in the target set R. The greater the number of such points, the more "noise" there is to conceal p from an attacker. As many chaff points are added to R, there begins to emerge a set of spurious polynomials that look like p, i.e., polynomials that have degree less than k and agree with exactly t points in R. Briefly stated, the more chaff points there are, the greater the probability that some set of t of these chaff points (and/or real points) align themselves by chance on some polynomial of the desired degree. In the absence of additional information, an attacker cannot distinguish between the correct polynomial p and all of the spurious ones. Thus, p is hidden in an information-theoretically secure fashion in R, with security proportional to the number of spurious polynomials. Note that the security of the vault V_A depends exclusively on the number of such polynomials, and not on the length of the secret key κ; the vault is often weaker than the secret κ it protects (which is acceptable for the applications we describe). The following lemma proves that with high probability many polynomials of degree less than k agree with the target set R in t places, i.e., that there are many spurious polynomials. This lemma and its proof are based on similar results of Dumer et al. [11].

Recall that the locking algorithm Lock picks t points according to a given p of degree less than k and $r-t$ random points (x_i, y_i) in $F_q \times F_q$ and outputs this set in random order as a vault hiding p (i.e., κ). The following lemma is parameterized by r, k, and t and a small real number μ:

Lemma 4. *For every μ, where $0<\mu<1$, with probability at least $1-\mu$ the target set R generated by the algorithm* Lock *on polynomial p and locking set A satisfies the following condition: There exist at least $\mu\binom{r}{t}q^{k-r}(q-1)^{r-t}$ polynomials p' of degree less than k such that R includes exactly t points of the form $(x, p'(x)) \in F_q \times F_q$.*

Proof. We first analyze a modified algorithm whose output is just a collection of r random points, say (x_i, z_i) in $F_q \times F_q$, where the x_i values are distinct. We claim that for every fixed choice of $x_1, \ldots, x_r$, the expected number of polynomials (over the choices of the z_i's) that agree with this set of points in t places, denoted N, is large. To see this, fix a polynomial p and consider the probability that p agrees with the set of points in exactly t places. This probability is given by the expression:

$$\binom{r}{t}q^{-t}(1-1/q)^{r-t}.$$

Thus the expected number of polynomials of degree less than k that agree with t of the r random points is $N = \binom{r}{t} q^{k-r}(q-1)^{r-t}$.

We now revert to our original problem, where the algorithm Lock is not outputting a random set of points, but rather a set of points t of which are selected to be from a fixed polynomial p. We will show that in this case the probability that the number of polynomials in agreement with the output set in t points is less than μN, is at most μ. (Once again we prove this for every fixed choice of $x_1, \ldots, x_r$.) This yields the lemma as desired. To prove this part, we construct a (huge) bipartite graph $\mathcal{G}$. The left vertices correspond to polynomials of degree less than k (i.e., there is one vertex for each such polynomial). The right vertices correspond to vectors from ${F_q}^r$ (i.e., there is one vertex for vector from ${F_q}^r$). Two vertices p and $y = (y_1, \ldots, y_r)$ are adjacent if p agrees with this vector in exactly t points, i.e., $p(x_i) = y_i$ for exactly t choices of i. The analysis of the modification of Lock showed that the average degree of a right vertex is N. Notice that the algorithm Lock, on the other hand, picks a fixed vertex p on the left and outputs a random neighbor of this vertex. Our goal is to show that this output vertex has high degree (say μN) with high probability. Towards this end, we first notice that the graph $\mathcal{G}$ is symmetric about left vertices, i.e., for every pair of vertices on the left p and p' there is an automorphism of $\mathcal{G}$ that maps p to p'. (Specifically, the automorphism maps a left vertex f to the vertex $f + p' - p$ and a right vertex $y = (y_1, \ldots, y_r)$ is mapped to the vertex $z = (z_1, \ldots, z_r)$ where $z_i = y_i + p'(x_i) - p(x_i)$.) Thus, it would suffice to consider the right endpoint of a random edge of $\mathcal{G}$ and argue that its degree is at least μN. But this is obviously true since vertices with degree $< \mu N$ can only account for a μ fraction of the edges in the graph $\mathcal{G}$. The lemma is thus proved. ■

Example 2. *As an example, consider the following choice of parameters. Suppose we pick a field size q of approximately 10^4, and set $r = q$. Now let $t = 22$, i.e., the movie lovers pick 22 of their favorite movies out of a choice of q, and we chaff the data with $q - 22$ random points. Suppose we use this information to encrypt a polynomial of degree less than 14 (as in our earlier example). Then we expect to see about 2^{86} polynomials of degree less than 14 agreeing with 22 out of the roughly 10^4 points in R. In particular, with probability at least $1 - 2^{-43}$, there will be 2^{43} polynomials exhibiting this behavior. (Thus, we achieve what may be roughly characterized as a 43-bit security level.)*

The example above suffers from a significant loss in security due to a naïve transformation of expected values to high probability results in the proof of Lemma 4. We believe that this loss in security is just an artifact of the proof, and that the true answer is perhaps more along the lines "With probability at least $1-2^{-83}$, there are 2^{83} polynomials agreeing with the given data on 22 points." (Thus, we get roughly 83-bit security.) Again, this conjecture remains open at this stage. For the moment, however, we try a different choice of parameters to strengthen our security analysis.

Example 3. *Again, we pick $r=q\approx 10^4$ and $t=22$. This time we use this information to encrypt a polynomial of degree less than* 18. *The decommitment works correctly with* 20 *agreements, and the running time is faster than a brute-force search by a factor of at least* 10. *Then we expect to see about* 2^{139} *polynomials of degree less than 18 agreeing with* 22 *out of the approximately* 10^4 *points in Q. In particular, with probability at least $1-2^{-70}$, there will be 2^{70} polynomials exhibiting this behavior. (Thus, we achieve what may be roughly characterized as a 70-bit security level.)*

As stated above, we believe our scheme more amenable to analysis over non-uniform distributions than that in [19]. As an example, we note that the above lemma naturally adapts itself to the case where the set of locking sets A are not all considered equally likely. For simplicity we consider the case where A is equally likely to come from some family of sets $\mathcal{E}\subset 2^{\mathcal{U}}=2^{F_q}$, i.e., a family of sets over $\mathcal{U}=F_q$. This is reflected in the following lemma.

Lemma 5. *For every $\mu>0$, with probability at least $1-\mu$, the target set R generated by the algorithm* Lock *to commit to a polynomial p with locking set A satisfies the following condition: There exist at least $\mu|\mathcal{E}|\left(\binom{r}{t}/\binom{q}{t}\right)q^{k-r}(q-1)^{r-t}$ polynomials $p'\in\mathcal{P}$ such that R agrees with p' on some subset of t points in the family $\mathcal{E}$.*

Proof. This lemma is proved in exactly the same way as the previous one. The only change is that the probability that a given polynomial p' agrees with a random set of r points in one of the subsets from $\mathcal{E}$ reduces to

$$|\mathcal{E}|\left(\binom{r}{t}\Big/\binom{q}{t}\right)(1/q)^t(1-1/q)^{r-t},$$

and this change percolates through the rest of the proof. ■

Example 4. *Consider a variant of the movie lover's problem where the movie lover is expected to choose 2 movies each from 10 categories, and each category consists of 1000 movies. In this case, the distribution on movies has* *support on only* $\left(\binom{10^3}{2}\right)^{10}$ *sets. The above lemma shows that with* $r = 10^4$*,* $t = 20$ *and* $k = 16$*, one expects to find* 2^{106} *polynomials of degree at most* 15 *agreeing with the data on* 20 *points, with two agreements each from each of* 10 *categories. As usual, this can be converted to the following probability statement: With probability at least* $1 - 2^{-53}$ *there exist* 2^{53} *polynomials of degree at most* 15 *that agree with the given data points on two points each in each of the* 10 *categories.* (*Thus, we achieve roughly a* 53*-bit security level.*)

Finally, we give a general characterization of the information-theoretic security of LOCK according to Definition 2.

THEOREM 6. *For every* $\delta > 0$*, the algorithm* LOCK *is* (δ, p)*-information theoretically secure for* $p = 2\sqrt{\frac{1}{3}q^{k-(1+\delta)t}(r/t)^{(1-\delta)t}}$

Proof. As in the proofs of Lemma 4 and Lemma 5, we first argue that given any $\mu > 0$, $E \subseteq \mathcal{U}$ of size t and $E' \subseteq E$ of size δt, the data points R have at least $\frac{\mu}{3}q^{k-(1+\delta)t}(r/t)^{(1-\delta)t}$ polynomials agreeing with the data points on t points that include E' as a subset.

Picking $\mu = \sqrt{\frac{1}{3}q^{k-(1+\delta)t}(r/t)^{(1-\delta)t}}$ we get that with probability at least $1 - \mu$, there are at least $1/\mu$ polynomials exhibiting such behavior. By symmetry, each of these polynomials is equally likely to be the polynomial p that is being encrypted. Thus the Attack algorithm has at most a 2μ chance of finding p—success probability of μ, if our encryption is unlucky and there are not too many polynomials agreeing with the data; and μ is the probability that the encryption is correct, but the attacker manages to guess it by luck. Thus we get the security as claimed. ■

Remark. It is possible to make a substantially stronger security claim under reasonable computational assumptions on the hardness of Reed–Solomon decoding, as done in, e.g., [23]. We do not explore this possibility here, as it is not essential to our achieving good security results, and there is no general consensus about appropriate hardness assumptions for this problem.

5. Application of Fuzzy Vaults

In the examples we have given above, the security parameterization is slightly weak from a cryptographic standpoint. For instance, in example 4, we achieve roughly a 53-bit security level, slightly weaker than DES, and thus vulnerable to intensive off-line attack [13]. In many cases, we may address this problem simply by re-parameterizing the vault, in particular, raising the size t of the set A. For example, to construct a strong personal-entropy system, we might require the user to answer 29 questions correctly out of 32 (with $t = 32$ and $k = 25$), thereby achieving a fuzzy vault with 85-bit security, which offers good cryptographic security for most purposes.

For applications involving privacy-protected matching, such as the movie-lover's problem, this approach will not work. The problem here lies not in the fuzzy vault scheme, but is inherent in the problem of matching people or other entities. In particular, given that there are only six billion or so people in the world, if Alice wants to have a good chance of finding a compatible movie lover, she can at best select a set that can be guessed by a well-informed attacker with a probability of one in six billion or so. As six billion is equal roughly to $2^{32.3}$, this means that Alice's vault can at best be expected to have less than 33-bit security[5]—entirely inadequate for cryptographic purposes. In most scenarios involving matching, the pool of participants, and thus the best achievable off-line security, is likely to be even smaller.

For this reason, we propose that fuzzy vaults in such privacy-protecting matching scenarios are best employed in an on-line setting. Here is a rough, first sketch of an idea for an on-line version of the movie-lover's problem. Alice publishes a fuzzy vault V_A encrypting a secret value κ under a set A of her favorite movies. If Bob wishes to obtain Alice's telephone number, he tries to open V_A using his set of favorite movies B. If he decodes successfully, he obtains a secret value κ'. Alice and Bob now invoke a *password-authenticated key-agreement protocol* (see [6] for a recent example). They use their respective secrets κ and κ' as passwords for this protocol.[6] If $\kappa = \kappa'$, then Alice and Bob will successfully establish a private channel. Otherwise, they will learn no information about the secret value of the other party, aside from the fact that $\kappa \neq \kappa'$. By employing an appropriate throttling mechanism, Alice can restrict the number of overall queries or queries from a single source so as to restrict information leakage.

This strategy effectively protects Alice's set A, but enables Alice to attract inappropriate matches. To illustrate this problem, consider the following simple strategy. Alice creates a vault V_A in which *all* x-coordinates agree with some polynomial p. Now, no matter what Bob inserts into his set B, he will think that Alice has the same favorite movies. This example highlights the important fact that a fuzzy vault is not, strictly speaking a commitment. In particular, it is not uniquely *binding*: Alice may embed multiple sets $A_1, A_2, \ldots, A_l$ in a single vault V_A.

There are several ways to avoid this difficulty. One way is for Alice to include in her vault a cryptographically binding commitment c_A to her secret value κ using, such as, e.g., a Pedersen commitment [24]. Now, Alice participates in the key agreement protocol with Bob in a manner that binds her to c_A (through a straightforward modification of existing algorithms). This does not ensure that V_A and c_A represent the same secret

κ, but this condition is not required for secure matching. A more general, mutually binding protocol is the following.

1. Alice publishes V_A on secret κ_A.
2. Bob publishes V_B on secret κ_B.
3. Alice applies UNLOCK to V_B using her set A. If successful, she obtains a value κ'_B. Otherwise, she aborts.
4. Bob applies UNLOCK to V_A using his set B. If successful, he obtains a value κ'_A. Otherwise, he aborts.
5. Alice and Bob do a password-authenticated key agreement using respective passwords $(\kappa_A \,\|\, \kappa'_B)$ and $(\kappa_B \,\|\, \kappa'_A)$. (Here, $\|$ is some suitable conjunctive operator.)

This protocol binds Alice and Bob to use of their respective vaults V_A and V_B. Omitting details, we remark that given use of a secure password-authenticated key-agreement algorithm (e.g., [6]), the above protocol reveals to Alice and Bob only the information contained in V_A and V_B and also whether $\kappa_A = \kappa_B$.

6. Conclusion: Further Research

Fuzzy vaults represent a new cryptographic primitive with the special property of error-tolerance in the presence of limited noise and arbitrary re-ordering of the symbols in a decomittment key. As explained above, one of the possible most promising applications of this idea is the protection of biometric templates, particularly for fingerprints. A major class of fingerprint matching algorithms are based on the comparison of registered with freshly presented features known as minutia points. (These are, roughly speaking, points in a fingerprint where ridges either end or intersect.) Because of the difficulty of achieving consistency in fingerprint imaging processes, and the consequent problems of erasure and re-ordering of minutia points in readings, measurement of set overlap is a natural metric to apply in this environment. The vanguard of research into fingerprint-matching algorithms is largely proprietary, making details difficult to obtain. It is hoped that such obstacles will not greatly hinder investigation into the application of fuzzy vaults to this promising avenue of research.

Another important area for further research is the analysis centered on Lemma 4. Our belief is that the transformation from expected values to high-probability bounds may be considerably improved. This might be achieved through a more refined understanding of the probability distribution of polynomials in agreement with random point sets—a basic question with larger ramifications in coding theory.

Acknowledgments

The authors wish to extend special thanks to Burt Kaliski for his ideas regarding Section 5. Thanks also to Daniel Bleichenbacher and Markus Jakobsson.

Notes

1. A serious vulnerability in this system is exposed in [5]. A fix with rigorously provable security properties is proposed in [14], based on [19]. Our own proposed scheme here offers an alternative with different properties.
2. So-called *set decoding* algorithms may in fact produce a set of candidate polynomials. We assume that a successful algorithm outputs one of these selected uniformly at random from the entire set.
3. We consider 22 titles, as this is the number of password questions used in [12], which seems a good example application for our ideas.
4. Another way of viewing this is that the fuzzy vault algorithm can be enhanced by additional use of brute-force search, thereby improving the security threshold. This improvement can be made substantial without a loss of speed relative to the pure brute-force algorithm.
5. It is possible to slow the algorithm Unlock so as to impose a higher computational burden on an attacker, but this approach still doesn't offer adequate security here.
6. Alternatively, as a practical alternative more compatible with existing infrastructure, they can employ the Secure Socket-Layer (SSL) protocol to establish a private, authenticated channel, and then employ a *socialet millionaires'* or similar protocol to test the condition $\kappa = \kappa'$ in zero knowledge [17,26]. This method depends upon one player having an appropriately signed certificate.

References

1. M. Alabbadi and S. B. Wicker, A digital signature scheme based on linear error-correcting block codes. In Josef Pieprzyk and Reihanah Safavi-Naini (eds.), *Asiacrypt '94*, Springer-Verlag (1994) LNCS no. 917, pp. 238–248.
2. C. H. Bennett, F. Bessette, G. Brassard, G. Savail and J. Smolin, Experimental quantum cryptography, *J. Cryptol.* Vol. 5, no. 1 (1992), pp. 3–28.
3. C. H. Bennett, G. Brassard, C. Crépeau and M.-H. Skubiszewska, Practical quantum oblivious transfer protocols. In J. Feigenbaum (ed.), *Crypto '91*, Springer-Verlag (1991). LNCS no. 576, pp. 351–366.
4. E. R. Berlekamp, *Algebraic Coding Theory*, McGraw Hill, New York (1968).
5. D. Bleichenbacher and P. Nyuyen, Noisy polynomial interpolation and noisy chinese remaindering. In B. Preneel (ed.), *Eurocrypt '00*, (2000) LNCS no. 1807, pp. 53–69.
6. V. Boyko, P. MacKenzie, and S. Patel, Provably secure password-authenticated key exchange using Diffie-Hellman. In B. Preneel (ed.), *Eurocrypt '00*, Springer-Verlag (2000) LNCS no. 1807, pp. 156–171.
7. C. Crépeau, Efficient cryptographic protocols based on noisy channels. In W. Fumy (ed.), *Eurocrypt '97*, Springer-Verlag, (1997) LNCS no. 1233, pp. 306–317.
8. C. Crépeau and J. Kilian, Achieving oblivious transfer using weakened security assumptions. In *Proceedings of the 29th IEEE Symposium on the Foundations of Computer Science* (1988), pp. 42–52.

9. G. I. Davida, Y. Frankel and B. J. Matt, On enabling secure applications through off-line biometric identification. In *IEEE Symposium on Privacy and Security* (1998).
10. G. I. Davida, Y. Frankel and B. J. Matt, On the relation of error correction and cryptography to an offline biometric based identification scheme. In *Proceedings of WCC99, Workshop on Coding and Cryptography* (1999).
11. I. Dumer, D. Micciancio and M. Sudan. Hardness of approximating the minimum distance of a linear code. In *Proceedings of the 40th Annual Symposium on Foundations of Computer Science (FOCS)*, (1999), pp. 475–484.
12. C. Ellison, C. Hall, R. Milbert and B. Schneier, Protecting Secret Keys with Personal Entropy, *J. Fut. Comput. Sys.* Vol. 16, no. 4 (2000, February) pp. 311–318.
13. Electronic Frontier Foundation, Cracking DES: Secrets of encryption research, wiretap politics & chip design. O'Reilly (1998).
14. N. Frykholm and A. Juels, An error-tolerant password recovery scheme. In P. Samarati (ed.), *Eighth ACM Conference on Computer and Communications Security*, ACM Press (2001) pp. 1–8.
15. V. Guruswami and M. Sudan, Improved decoding of Reed–Solomon and algebraic-geometric codes, In *FOCS '98*, IEEE Computer Society (1998), pp. 28–39.
16. T. Jakobsen, Cryptanalysis of block ciphers with probabilistic non-linear relations of low degree, In H. Krawczyk (ed.), *Crypto '98*, Springer-Verlag (1998) LNCS no. 1462, pp. 212–222.
17. M. Jakobsson and M. Yung, Proving with knowing: On oblivious, agnostic, and blindfolded provers, In N. Koblitz (ed.), *Crypto '96*, Springer-Verlag (1996), LNCS no. 1109, pp. 186–200.
18. I. Jermyn, A. Mayer, F. Monrose, M. K. Reiter and A. D. Rubin, The design and analysis of graphical passwords, In *Proceedings of the 8th USENIX Security Symposium* (1999), pp. 1–14.
19. A. Juels and M. Wattenberg, A fuzzy commitment scheme, In G. Tsudik, (ed), *Sixth ACM Conference on Computer and Communications Security*, ACM Press (1999), pp. 28–36.
20. A. Juels and M. Sudan, A fuzzy vault scheme, In *International Symposium on Information Theory (ISIT)*, IEEE Pressm, (2002), p. 408.
21. J. L. Massey, Shift register synthesis and BCH decoding. *IEEE Trans. Inform. Theory*, Vol. 15, no. 1 (1969) pp. 122–127.
22. R. J. McEliece, A public-key cryptosystem based on algebraic coding theory, Technical Report DSN progress report 42–44, Jet Propulsion Laboratory, Pasadena (1978).
23. F. Monrose, M. K. Reiter and S. Wetzel, Password hardening based on keystroke dynamics, In G. Tsudik (ed.), *Sixth ACM Conference on Computer and Communications Security*, ACM Press (1999), pp. 73–82.
24. T. Pedersen, Non-interactive and information-theoretic secure verifiable secret sharing. In J. Feigenbaum (ed.), *Crypto '91*, Springer-Verlag (1991), LNCS no. 576, pp. 129–140.
25. W. W. Peterson, Encoding and error-correction procedures for Bose-Chaudhuri codes, *IEEE Trans. Inform. Theory*, Vol. IT-60 (1960) pp. 459–470.
26. B. Schoenmakers, F. Boudot and J .Traoré, A fair and efficient solution to the sociaset millionaires' problem, *Discrete Appl. Math.* Vol. 111 (2001, July) pp. 23–36.
27. A. Shamir, How to share a secret, *Commun. ACM*, Vol. 22 (1979) pp. 612–613.
28. C. Soutar, Biometric encryption for secure key generation, January 1998, *Presentation at the 1998 RSA Data Security Conference*.
29. C. Soutar and G. J. Tomko, Secure private key generation using a fingerprint, In *Card-Tech/SecurTech Conference Proceedings*, Vol. 1, (May 1996) pp. 245–252.
30. J. Stern, A new identification scheme based on syndrome decoding, In D.R. Stinson (ed.), *Crypto '93*, Springer-Verlag (1993), LNCS no. 773, pp. 13–21.