We can prove formally that the above is a zero-knowledge interactive proof system; for details refer to [3].

Security of protocol: The security of the protocol relies on the following:

(i) The general decoding problem for linear codes is NP-complete [1]. The proposed protocol is based on the syndrome decoding problem for rank distance codes, which appears even more difficult than for Hamming distance codes.

(ii) The number of the vectors of rank r is

$$N_r = \Pi_{i=1}^{r} \frac{(q^n - q^{i-1})(q^m - q^{i-1})}{q^r - q^{i-1}} \simeq q^{r(n+m)-r^2}$$

(iii) There are $N(n) = \Pi_{i=1}^{n} (q^n - q^{i-1}) \simeq q^{n(n-1)}$ regular $n \times n$ matrices.

As a numerical example, consider the case where $n = 32$, $k = 16$, $m = 8$, $4 \leq r \leq 8$ and $q = 2$. There are three attacks to this scheme:

(i) From $\mathbf{i} = \mathbf{sH}^t$, find the solution of rank r. Because it deals with rank distance codes, none of the known decoding algorithms can be used. The equation $\mathbf{i} = \mathbf{sH}^t$ has $q^{m(n-k)} = 2^{128}$ solutions, which is too many for an exhaustive search.

(ii) Test all vectors of rank r. There are $N_r \geq 2^{144}$ such vectors, therefore it is not practical to test them all.

(iii) By means of the knowledge of $\mathbf{sP}^{-1}$, find $\mathbf{s}$. The number of regular matrices ($N(n) \simeq 2^{992}$) is so large that it will be infeasible to find the secret key.

If we choose suitable small values of r, e.g. $r \leq$ the error correction capability of the code, the possible collisions (two different keys may have the same identification by $\mathbf{H}$) could be avoided. In addition, the improved scheme has many advantages in comparison with the original scheme:

(i) Without loss of generality we can assume that the system matrix $\mathbf{H}$ is given in the block form $\mathbf{H} = (\mathbf{I}, \mathbf{H}_1)$, where $\mathbf{I}$ is the $k \times k$ identity matrix and H_1 is a $k \times (n - k)$ matrix over $GF(q^m)$. It requires 2^{11} memory bits.

(ii) The secret key $\mathbf{s}$ can be stored in 256 bits, and the public key $\mathbf{i}$ can be stored in 128 bits.

(iii) The communications complexity of the protocol is (average) 4992 bits. Because the user must send vector $\mathbf{i}'$ (128 bits) and matrix $\mathbf{H}'$ (4096 bits) and either $\mathbf{P}$ (1024 bits), $\mathbf{Q}$ (256 bits) or s' (256 bits) per round.

(iv) The main computing task of the protocol is that the prover performs one matrix-matrix multiplication and the verifier performs one matrix-matrix multiplication every two interactions (on average). It is equivalent to 2^{17} and 2^{16} bit multiplications, respectively.

(v) The new protocol does not need to use any cryptographical hash function.

Acknowledgment: I would like to thank an anonymous colleague who offered me [3], which is very hard to find. Without his help, the presented scheme would have been almost impossible to develop.

4 June 1994
Electronics Letters Online No: 19941106

K. Chen (*Mathematisches Institut, Arndt Strasse 2, D-35392 Giessen, Germany*)

References

1 BERLEKAMP, E.R., MCELIECE, R.J., and VAN TILBORG, H.C.A.: 'On the inherent intractability of certain coding problems', *IEEE Trans.*, 1978, **IT-24**, (3), pp. 384–387
2 GABIDULIN, E.M.: 'Theory of codes with maximum rank distance', *Problems of Information Transmission*, 1985, **21**, pp. 1–12
3 GIRAULT, M.: 'A (non-practical) three-pass identification protocol using coding theory'. Proc. Auscrypt'90, 1990, Vol. 453, pp. 265–272 (LNCS)
4 HARARI, S.: 'A new authentication algorithm', *Coding Theory and Applications*, 1989, **388**, pp. 204–211 (LNCS)
5 STERN, J.: 'An alternative to the Fiat-Shamir protocol'. Proc. Eurocrypt'89, 1990, Vol. 434, pp. 173–180 (LNCS)
6 STERN, J.: 'A new identification scheme based on syndrome decoding'. Proc. Crypto'93, 1994, Vol. 773, pp. 13–21 (LNCS)

Multistage secret sharing based on one-way function

J. He and E. Dawson

Indexing term: Cryptography

The concept of multistage secret sharing (MSS) is proposed and a general implementation of MSS schemes given. In such a scheme, many secrets are shared in such a way that all secrets can be reconstructed separately. Each share is of the same size as that of any single shared secret.

Introduction: Secret sharing is a very useful tool in modern cryptography [1, 2]. Informally, a secret sharing scheme gives a method for distributing a secret s among a set P of participants in such a way that only qualified subsets of P can reconstruct s whereas any other (nonqualified) subset of P cannot determine anything about the value of s. Usually, s is divided by a dealer into $n = |P|$ pieces called shares, and each share will be delivered privately to a particular participant. We call this a sharing of the secret s. Later on, all members in any qualified subset of P could pool their shares together and reconstruct the secret.

One common drawback of almost all known secret sharing schemes is that they are one-time schemes in the following sense. Once a qualified group of participants reconstructs the secret s by pooling their shares, both the secret s and all shares become known to everyone within the group or even everyone else, and there is no further secret. In other words, each share kept by each participant can be used to reconstruct only one secret.

In this Letter we propose the concept of multistage secret sharing (MSS). In an MSS scheme, many secrets are shared but only one share is kept by each user. The share has the same size as that of any single secret. The secrets will be reconstructed stage by stage in a specified order, but the reconstruction of secrets at earlier stages does not reveal or weaken the secrecy of the remaining secrets. In other words, each participant can use his/her secret share many times.

A general method is presented for implementing an MSS scheme based on any traditional secret sharing scheme which is not necessarily a threshold scheme. This method is elaborated using the well known Shamir scheme [2] as an example. The implementation assumes the existence of a one-way function [3]. Note that any kind of unconditionally secure multistage secret sharing is impossible due to the information theory lower bound of [4].

Basic technique: We now explain the basic technique with which most traditional secret sharing schemes can be extended to an MSS scheme. In this Letter we always work in the finite field Z_p where p is a fixed odd prime. All numbers are elements of Z_p.

For any polynomial f and any a, b, if $f(a) = b$, then we say that the point (a, b) lies on f, or, a and b are consistent with f (and vice versa). A set of points $\{(x_i, y_i) : i = 1, 2, \ldots, n\}$ is t-consistent if there exists some polynomial f of degree at most t such that all these points lie on f. Such a set is consistent if it is t-consistent for some t.

Consider the following problem. Let n,t be fixed integer parameters where it is assumed that $n \geq t$. Let s be the secret to be shared. Let $x_1, x_2, \ldots, x_n$ be n distinct numbers which are publicly known to everyone. Now suppose the dealer arbitrarily chooses n

numbers $y_1, y_2, \ldots, y_n$ (not necessarily distinct), and delivers each y_i privately to participant P_i. It is highly probable that the set of n points $\{(x_i, y_i) : i = 1, 2, \ldots, n\}$ is not $(t-1)$-consistent. However, it is possible to modify the n values $y_1, y_2, \ldots, y_n$ by adding to them some necessary shift values so that the resulting values become $(t-1)$-consistent. Furthermore, all shift values can be made public without revealing any information about s.

The method is as follows:

(i) First, the dealer chooses $a_1, a_2, \ldots, a_{t-1}$ randomly from Z_p and forms the polynomial

$$f(x) = s + a_1 x + a_2 x^2 + \cdots + a_{t-1} x^{t-1}$$

(ii) Secondly, the dealer computes the values

$$z_i = f(x_i),\ d_i = z_i - y_i,\ i = 1, 2, \ldots, n$$

(iii) Finally, the dealer publishes all shift values d_i, $i = 1, 2, \ldots, n$.

Each participant P_i can obtain his true share z_i by adding the public shift value d_i to his secret share y_i, and use z_i to reconstruct the polynomial f. Because d_i is used with y_i to obtain the true share z_i, we say that d_i matches y_i.

Theorem 1: The public shift values $d_i (i = 1, 2, \ldots, n)$ reveal nothing about the secret s.

Proof of theorem 1: We consider the worst case. Suppose $(t-1)$ participants $P_1, P_2, \ldots, P_{t-1}$ work together and try to reconstruct s. They can use their $(t-1)$ true shares $z_1, z_2, \ldots, z_{t-1}$. In addition, they know all public shift values.

For every s', there is a uniquely determined polynomial g of degree $(t-1)$ such that g is consistent with the t points $(0, s')$, (x_1, z_1), (x_2, z_2), $\ldots$, $x_{t-1}, z_{t-1})$ and is not in contradiction with the remaining $(n-t)$ shift values because we can always match each of them with some corresponding value. In fact, d_i matches $g(x_i) - y_i$, $i = t+1, 2, \ldots, n$.

So to any group of $(t-1)$ participants, every s' is an equally possible candidate for the shared secret, and the unconditional security of the scheme is maintained.

For convenience of subsequent discussion, we call the above method the public shift technique. The arbitrarily chosen $y_1, y_2, \ldots, y_n$ are called secret shares, and $z_1, z_2, \ldots, z_n$ are called true shares.

Note that the public shift technique is very similar to the public broadcasting technique used in the implementation of threshold schemes with disenrollment capability [5, 6]. The difference is that in [5], some messages are broadcast only when some share is disclosed and the existing shares need to be updated. In our scheme, all shift values are published in the first place. Also, the goals are different. In [5], to achieve an L-fold disenrollment capability and to maintain the unconditional security, each participant needs to keep $L+1$ secret shares (plus using all public broadcasting messages). In our scheme, each participant just keeps one secret share and the computational security is guaranteed.

MSS scheme: Let $f : Z_p \to Z_p$ be any one-way function. For any x and any non-negative integer k, $f_k(x)$ denotes k successive applications of f to x, i.e., $f^0(x) = x$, $f^k(x) = f(f_{k-1}(x))$ for $k > 0$.

Let $s_i (i = 1, 2, \ldots, k)$ be k secrets to be shared. Let $x_1, x_2, \ldots, x_n$ be n distinct numbers which are publicly known to everyone. The dealer will do the following:

(i) Choose n numbers $y_1, y_2, \ldots, y_n$ arbitrarily.

(ii) For $i = 1, 2, \ldots, k$, execute the following step: share s_i by using the public shift technique, with $f^{i-1}(y_j) (j = 1, 2, \ldots, n)$ as the secret shares. Let the resulting shift values be d_{ij}, $j = 1, 2, \ldots, n$.

(iii) Deliver y_i privately to participant P_i, and publish all shift values.

The reconstruction of the k secrets will be conducted in k stages, with s_{k-i+1} reconstructed at the ith stage, $i = 1, 2, \ldots, k$. When trying to reconstruct the ith secret s_i, an involved participant P_j should submit his ith secret share $f^{i-1}(y_j)$. So each participant P_j will use his initial secret y_j in the form and order $f^{k-1}(y_j)$, $f^{k-2}(y_j)$, $\ldots$, $f^0(y_j)$, each of which is used as a secret share at the corresponding stage. In this way, his pooled secret shares would not compromise his not yet used secret shares, because (e.g.) $f^{i-1}(y_j)$ will in no way enable anyone (except P_j himself) to obtain $f^{i-2}(y_j)$, unless he can invert the one-way function f.

At the $(k-i+1)$st stage, the corresponding secret s_i can be reconstructed if and only if at least t participants, say $P_1, P_2, \ldots, P_t$, have submitted their secret shares $f^{h_j}(y_j)$ for some $h_j \le i-1$, $j = 1, 2, \ldots, t$. Even if up to $(t-1)$ participants pool their initial secrets, they still cannot reconstruct a secret if none of the remaining participants agrees to co-operate and submit the necessry secret share for the appropriate stage.

Complexity: For k secrets shared among n participants, each participant has to keep only one secret element in his hand. This is of the same size as any shared secret. However, there are a total of kn public messages (public shift values).

2 August 1994
Electronics Letters Online No: 19941076

J. He and E. Dawson (*Information Security Research Center, Queensland University of Technology, Gardens Point, GPO Box 2434, Brisbane, QLD 4001, Australia*)

References

1 BLAKLEY, G.R.: 'Safeguarding cryptographic keys'. Proc. AFIPS 1979 National Computer Conf., 1979, Vol. 48, pp. 313-317
2 SHAMIR, A.: 'How to share a secret', *Commun. ACM*, 1979, **22**, (11), pp. 612-613
3 DIFFIE, W., and HELLMAN, M.E.: 'New directions in cryptography', *IEEE Trans.*, 1976, **IT-22**, (6), pp. 644-654
4 KARNIN, E.D., GREENE, J.W., and HELLMAN, M.E.: 'On secret sharing system', *IEEE Trans.*, 1983, **IT-29**, (1), pp. 35-41
5 BLAKLEY, B., BLAKLEY, G.R., CHAN, A.H., and MASSEY, J.L.: 'Threshold schemes with disenrollment'. Advances in Cryptology-CRYPTO'92, BRICKELL, E.F., (Ed.), pp. 540-548
6 DAWSON, E., and DONOVON, D.: 'Shamir's scheme says it all', *IFIP Trans.*, 1993, **A-37**, pp. 91-102

Security of interactive DSA batch verification

C.H. Lim and P.J. Lee

Indexing terms: Cryptography, Data privacy, Security of data, Smart cards

The authors show that the interactive DSA batch transaction protocol proposed by Naccache *et al.* at Eurocrypt'94 is not secure.

Introduction: The DSA (digital signature algorithm) of NIST [1] is receiving widespread attention. At Eurocrypt'94, Naccache *et al.* [2] proposed various ways to optimise the DSA for smartcard applications. These include batch verification, inversionless signing and signer-aided compression, etc. Naccache *et al.* presented two kinds of batch transaction protocol for fast DSA verification : the interactive batch and probabilistic batch. This Letter investigates the security of the interactive batch verification of DSA. More precisely, we show that either the signer or the verifier can easily generate a collection of bogus signatures that satisfies the batch verification criterion. Of course, each of the faked signatures does not satisfy the DSA verification condition individually. Consequently the interactive batch verification method is not secure at all.

Description of DSA: Let p, q and g be the DSA public parameters : p a large prime, q a large prime divisor of $p-1$ and g an element in Z_p of order q. Let H denote the secure hash algorithm. The signer has a secret key $x \in Z_q$ and a public key $y = g^x \bmod p$. To generate a DSA signature for message m, the signer randomly picks $k \in Z_q$ and computes $\{r, s\}$ as

$$r = (g^k \bmod p) \bmod q,\ s = k^{-1}(H(m) + rx) \bmod q$$