Cryptanalysis of MQV with partially known nonces.

P.J. Leadbitter, N.P. Smart

Computer Science Department
University of Bristol
Woodland Road,
Bristol BS8 1UB,
United Kingdom.
{peterl, nigel}@cs.bris.ac.uk

Abstract. In this paper we present the first lattice attack on an authenticated key agreement protocol, which does not use a digital signature algorithm to produce the authentication. We present a two stage attack on MQV in which one party may recover the other party's static private key from partial knowledge of the nonces from several runs of the protocol. The first stage reduces the attack to a hidden number problem which is partially solved by considering a closest vector problem and using Babai's algorithm. This stage is closely related to the attack of Nguyen and Shparlinski on DSA but is complicated by a non-uniform distribution of multipliers. The second stage recovers the rest of the key using the baby-step/giant-step algorithm or Pollard's Lambda algorithm and runs in time $O(q^{1/4})$. The attack has been proven to work with high probability and validated experimentally. We have thus reduced the security from $O(q^{1/2})$ down to $O(q^{1/4})$ when partial knowledge of the nonces is given.

1 Introduction

An authenticated key agreement protocol is a key agreement protocol in which both sides are guaranteed the identity of the party with which they are agreeing a key. This is usually done through both parties having a static public/private key pair, for which the other party has the certified public key. To break the authenticity of such schemes it is clearly sufficient to obtain the victims private key. There are essentially two types of such protocols, those in which the authenticity is produced via a digital signature and those in which the authenticity is produced as an integral part of the protocol.

The MQV [7] protocol is an authenticated key agreement protocol, proposed by Law, Menezes, Qu, Solinas, and Vanstone, based on the standard un-authenticated Diffie-Hellman key agreement protocol. It is believed to have known key security, forward secrecy and key-compromise impersonation resilience under the assumption of the intractability of the elliptic curve discrete log problem and the elliptic curve Diffie-Hellman problem. There is not yet however any formal security proof to support these claims. The MQV protocol has the same message flows as the standard Diffie-Hellman protocol, but obtains an authentic shared key using a non-standard method for obtaining the shared secret. Hence, MQV achieves better bandwidth performance than simply signing the standard Diffie-Hellman message flows.

In this paper we introduce a scenario not discussed in [7]. Suppose one party A involved in the key agreement process can, illegitimately or otherwise, recover several of the most significant bits of some of the secret ephemeral nonces that the second party B calculates each time A and B generate a shared secret. This partial information can come from many sources, for example side channel analysis or a poor implementation of a random number generator. We do not assume A to have any knowledge of B's static private key. After sufficiently many runs of the protocol, we show A is able to recover the whole of B's static private key. Since this is the key that uniquely identifies the user B, the user A is clearly able to impersonate B, possibly without B's knowledge.

Lattices have played a key role in cryptanalysis since Coppersmith proposed an LLL based technique of finding small roots of modular polynomials, with applications to low exponent RSA [5]. The recurring theme is to construct a lattice that holds some vital secret in one of its lattice points. The characteristic we use to identify the point we are looking for is its distance from a known non-lattice point, or its distance from zero, (in which case we are searching for a small lattice point). There is no efficient algorithm to find the closest lattice point, or indeed to find the smallest lattice point. There are however polynomial time algorithms which provably find close vectors and small lattice vectors, namely Babai's algorithm [1] and the LLL algorithm of Lenstra, Lenstra and Lovasz [8]. Since such small (close) vectors are rare, the vectors these algorithms recover, tend to be, or be closely related to, the lattice vector we are looking for. In practice these algorithms work remarkably well.

One problem with which we are left, is how to ensure our hidden vector is small. The common approach, as exploited in this paper, is to consider the case where we are given some information, either through poor implementation or other means. The lattice attacks on RSA [2], [5] were limited by the stringent bounds imposed on the size of the decryption exponent or the number of known bits of the factors or the secret key. This is a consequence of their only being able to use the information from the encryption of one message in a given lattice. The scenario introduced by Howgrave-Graham and Smart [6] and continued by Nguyen and Shparlinkski [9] was favourable to the attacker, since the gathering of information could be spread throughout several signatures, making the attack more realistic. It also lets the attacker construct a lattice which takes full advantage of the properties of the l_∞ norm. This style of attack needs one run of the scheme to be different to the next run (even for the same input data). In DSA, since each of the static and ephemeral keys can be take any value modulo q the associated lattice attacks are successful. In the paper, we explain how in MQV the variety between successive runs is more limited making it more secure against this style of attack but still not completely secure. In addition we still need to solve a discrete logarithm problem to break the scheme, although one which is considerably easier than the original discrete logarithm. For the attacks on DSA and EC-DSA the lattice attacks recover the secret key without the need for this expensive auxillary step.

The paper is structured as follows: In section 2 we set up the notation and describe the MQV protocol and outline our attack. In section 3 we theoretically analyse the first (lattice-based) stage of our attack. In section4 we describe the second stage of our attack which runs in time $O(q^{1/4})$. In section 5 we present our experimental results using a lattice slightly different to that used in section 3 but which is better suited to practical situations. Finally in section 6 we present some conclusions.

2 Notation

For any $m, n \in \mathbb{Z}$, we define $\lfloor m \rfloor_n$ be the unique $r \in \mathbb{Z}$ such that $0 \leq r < n$ and for some $q \in \mathbb{Z}, r + qn = m$. In other words $\lfloor m \rfloor_n$ returns the positive remainder on division of m by n. We let

$$|m|_q = \min_{k \in \mathbb{Z}} |m - kq| = \min\{\lfloor m \rfloor_q, q - \lfloor m \rfloor_q\}.$$

Let $\mathtt{MSB}_{l,q}(x)$ denote any integer u such that

$$|x - u| \leq q/2^{l+1}.$$

In other words the function $\mathtt{MSB}_{l,q}(x)$ returns an integer which agrees with x on at least the l-th most significant digits. The hidden number problem is the task of recovering the hidden number b when we are given

$$\mathtt{MSB}_{l,q}(\lfloor bt_i \rfloor_q)$$

for many random but known $t_i \in \mathbb{F}_q$. The hidden number problem has been studied quite extensively since first being introduced in [4] by Boneh and Venkatesan. We shall present an overview of the basic lattice strategy for solving such problems in the next section.

We now discuss the MQV protocol. We let $\mathbb{F}$ be a finite field and E be an elliptic curve over $\mathbb{F}$ whose order is divisible by a prime q of size $\approx 2^f$. For any $Q \in E(\mathbb{F})$, let

$$\overline{Q} = \lfloor x \rfloor_{2^{\lceil f/2 \rceil}} + 2^{\lceil f/2 \rceil}$$

where f is the bitlength of q and x is the integer obtained by considering the binary representation of the first coordinate of Q.

We can now describe the MQV protocol in the elliptic curve setting as follows: Let P be a point of order q in $E(\mathbb{F})$. Two entities A and B possess a static private keys

$$w_A, w_B \in \{1, \dots, q-1\}$$

and corresponding static public keys

$$W_A = w_A P,$$
$$W_B = w_B P.$$

For each run of the protocol, the following steps are executed:

1. A generates the private ephemeral key
$$r_A \in \{1, \ldots, q-1\},$$
and computes the public ephemeral key $R_A = r_A P$. The ephemeral key R_A is then passed to B.
2. B generates the private ephemeral key
$$r_B \in \{1, \ldots, q-1\},$$
and computes public ephemeral key $R_B = r_B P$. The ephemeral key R_B is then passed to A.
3. A computes
$$s_A = \lfloor r_A + \overline{R}_A w_A \rfloor_q$$
and
$$R_B + \overline{R}_B W_B = s_B P.$$
4. B computes
$$s_B = \lfloor r_B + \overline{R}_B w_B \rfloor_q$$
and
$$R_A + \overline{R}_A W_A = s_A P.$$
5. A and B both calculate the shared secret
$$K = \frac{\#E(\mathbb{F})}{q} s_A s_B P.$$

In our scenario we assume that the user A wishes to obtain the static private key w_B of the user B. They will aim to do this by performing a number of valid protocol runs with the user B and recording the message flows, their own ephemeral secrets and in addition determining the l most significant bits of the integer

$$s_B - r_B \pmod{q},$$

where we assume l is small. We do not discuss how such bits are determined but simply note that performing a power analysis on a naive implementation of the point multiplications $r_B P$ and $s_B P$ which are performed by user B may be all that is required.

We assume that user A interacts in this manner with user B a total of n times. For the ith run of the protocol, we have that by assumption $\lfloor s_B^{(i)} - r_B^{(i)} \rfloor_q = x_i + u_i$ for some unknown but small $x_i \in \mathbb{Z}$ and known $u_i = \mathtt{MSB}_{l,q}(s_B^{(i)} - r_B^{(i)})$.

The calculations of the nonces

$$s_B = \lfloor r_B + \overline{R}_B w_B \rfloor_q$$

then give rise to n equations:

$$x_i - t_i b + u_i = 0 \pmod{q} \tag{1}$$

where $t_i = \overline{R}_B^{(i)}$ and $b = w_B$.

Note that:

- The x_i are all less than $q/2^l$ in absolute value and are unknown.
- The integer b lies in the range $[0, q]$ and corresponds to user B's static private key and is completely unknown.
- The integer u_i lies in $[0, q]$ and is given.
- Finally the given integers t_i satisfy

$$2^{\lceil f/2 \rceil} \leq t_i < 2^{\lceil f/2 \rceil + 1}.$$

We then have that $\lfloor t_i b - u_i \rfloor_q < q/2^l$ for $1 \leq i \leq n$, which gives us our hidden number problem.

Our attack now proceeds in two stages:

- **Stage One:** We recover the $f/2$ most significant bits of b, using the algorithms of Babai and LLL.
- **Stage Two:** We recover the rest of the bits of b using an analogue of the baby-step/giant-step algorithm, or using Pollard's Lambda method.

3 Theoretical Analyse of the Lattice Stage

Here we outline the Boneh and Venkatesan algorithm proposed in [4] as presented in [9]. The only modification being we do not choose our multipliers t_i uniformly in $\mathbb{F}_q$. We present our analyse quite generally, and not just to the MQV specific problem, in that we assume that the multipliers t_i are of the form $\gamma \alpha_i$ where α_i are chosen uniformly in

$$A = \{a \in \mathbb{F}_q : a_1 \leq \lfloor a \rfloor_q < a_2\}.$$

where $0 \leq a_1 < a_2 \leq q$ and $\gamma \in \mathbb{F}_q$. Our hidden number problem is then given by $u_i = \mathtt{MSB}_{l,q}(\lfloor b t_i \rfloor_q)$, where b is the only unknown.

Note that Nguyen and Shparlinski consider the case

$$(\gamma, a_1, a_2) = (1, 0, q-1)$$

in their analysis [9]. The MQV case as discussed above is

$$(\gamma, a_1, a_2) = (1, 2^{\lceil f/2 \rceil}, 2^{\lceil f/2 \rceil + 1} - 1).$$

Consider the lattice $L(q, l, t_1 \dots t_n)$ generated by the rows of:

$$M = \begin{pmatrix} q & 0 & \cdots & 0 & 0 \\ 0 & q & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & 0 & \vdots \\ 0 & \cdots & 0 & q & 0 \\ t_1 & \cdots & \cdots & t_n & 2^{-(l+1)} \end{pmatrix} \tag{2}$$

Since there exist $k_i \in \mathbb{Z}$ such that $\lfloor bt_i \rfloor_q = bt_i + k_i q$ and $b \in \mathbb{Z}$, there is a "hidden" lattice point

$$\boldsymbol{h} = (k_1, \dots, k_n, b) \cdot M = \left(\lfloor bt_1 \rfloor_q, \dots, \lfloor bt_n \rfloor_q, b/2^{l+1} \right)$$

which is close to the known vector

$$\boldsymbol{u} = (u_1, \dots, u_n, 0).$$

By the definition of u_i and the size of b, we have that

$$\|\boldsymbol{h} - \boldsymbol{u}\|_\infty < q/2^{l+1}$$

Given the lattice $L(q, l, t_1 \dots t_n)$ and $\boldsymbol{u}$, Babai's nearest plane algorithm [1] recovers a lattice point $\boldsymbol{v}$, which is close to $\boldsymbol{u}$. We have that

$$\begin{aligned} \|\boldsymbol{u} - \boldsymbol{v}\| &\le 2^{(n+1)/4} \|\boldsymbol{u} - \boldsymbol{h}\| \\ &\le (n+1)^{1/2} 2^{(n+1)/4} q/2^{l+1} \end{aligned}$$

and so

$$\|\boldsymbol{h} - \boldsymbol{v}\|_\infty \le \|\boldsymbol{h} - \boldsymbol{u}\|_\infty + \|\boldsymbol{u} - \boldsymbol{v}\| \le \frac{q(1 + (n+1)^{1/2} 2^{(n+1)/4})}{2^{l+1}}. \tag{3}$$

So for sufficiently small l, we can guarantee that the hidden vector and the recovered vector differ by an extremely short lattice point.

Consider the case where $\gamma = 1$, so for $i = 1 \dots n$, the t_i are chosen independently and uniformly in $[a_1, a_2] \cap Z$. Let $\beta \in \mathbb{Z} \setminus \{0\}$, $Q \in \mathbb{R}$ and let $E(\beta)$ be the event that for all i,

$$|\beta t_i|_q \le Q.$$

Note that $E(\beta) = E(\beta + kq)$ for any $k \in \mathbb{Z}$, so we need only consider $-q/2 < \beta \le q/2$. Also if $-q/2 < \beta < 0$, the problem is then equivalent to considering $-\beta$ and t_i is chosen independently and uniformly in $[-a_2, -a_1] \cap \mathbb{Z}$.

So first consider $0 < \beta \le q/2$. We then have that

$$\#\{t_i \in [a_1, a_2] \cap \mathbb{Z} : |\beta t_i|_q \le Q\} = \sum_{k=a_1'}^{a_2'} \left(\left\lfloor \frac{kq+Q}{\beta} \right\rfloor - \left\lceil \frac{kq-Q}{\beta} \right\rceil + 1 \right)$$
$$+ \max(\lfloor (a_1' - 1 + Q)/\beta \rfloor - \lceil a_1 \rceil + 1, 0)$$
$$+ \max(\lfloor a_2 \rfloor - \lceil (a_2' + 1 - Q)/\beta \rceil + 1, 0)$$

where $a_1' = \lceil \beta a_1/q \rceil$ and $a_2' = \lfloor \beta a_2/q \rfloor$. Hence,

$$P(E(\beta)) = \prod_{i=1}^{n} P(|\beta t_i|_q \le Q) \le \left(\frac{(\frac{\beta(a_2-a_1)}{q} + 1)(\frac{2Q+1}{\beta})}{(a_2 - a_1) + 2} \right)^n. \tag{4}$$

We now consider $\beta(a_2 - a_1) \ge q$, in which case we have

$$P(E(\beta)) \le \left(\frac{4Q+2}{q} \right)^n \le \frac{1}{2^{\nu n}} \tag{5}$$

taking Q of the form $q/(2^{\nu+2}) - 1/2$. Notice that by the comments above, this result holds for all $\beta \in \mathbb{Z}$. Furthermore the event that for all i, $|\beta t_i|_q \le Q$ with $t_i = \alpha_i \gamma$ and α_i chosen independently and uniformly in A is simply $E(\beta\gamma)$.

Consider a lattice point

$$\boldsymbol{w}(\boldsymbol{z}, \beta) = (z_1, \ldots, z_n, \beta) \cdot M$$

and let $T = \{\beta \in \mathbb{Z} : |\gamma\beta|_q \ge q/(a_2 - a_1)\}$. It is now easy to see that the probability that there exists a $\boldsymbol{z} \in \mathbb{Z}^n$ and a $\beta \in T$ such that $\|\boldsymbol{w}(\boldsymbol{z}, \beta)\|_\infty \le Q$, is bounded by $q/2^{\nu n}$.

We can then deduce that with high probability a short vector $\boldsymbol{w} \in L$, is of the form $\boldsymbol{w}(\boldsymbol{z}, \beta)$ where $|\gamma\beta|_q < q/(a_2 - a_1)$.

Applying this method to MQV will yield the upper half of the bits of b. However we do not expect to recover the lower bits since there are at least $\left\lfloor \frac{Q}{\max(a_2, -a_1)} \right\rfloor$ values of $\beta \in \mathbb{F}_q^*$ for which $\boldsymbol{w}(\boldsymbol{0}, \beta) \le Q$. Recall that we recover b completely if and only if $\beta = 0 \pmod q$. The difficulty of recovering the actual hidden vector provides additional security against this style of attack. It substantially slows down the process of recovering the key and if the keylength is too large, this method becomes infeasible.

4 Baby-Step/Giant-Step Stage

We are now faced with a discrete log problem, $W = bP$, where the upper half of the bits of b are known. There are two ways of solving this problem in time $O(q^{1/4})$, the first is deterministic and requires a large storage requirement, the second has minimal storage requirement and produces an expected run time of $O(q^{1/4})$.

The first deterministic method is Shanks' Baby-Step/Giant-Step method, applied to this restricted discrete logarithm problem. Let d denote the known bits of b, then by the Euclidean division algorithm there exists λ and $\mu \in \mathbb{Z}$ such that $d = \lambda \lceil q^{1/4} \rceil + \mu$. Note that $W - (b-d)P - \mu P = \lambda \lceil q^{1/4} \rceil P$.

1. Calculate and store $U = W - (b-d)P$ and $V = \lceil q^{1/4} \rceil P$
2. Calculate and store the "baby steps", $\mathtt{Baby}_\mu = U - \mu P$ in order of size of the integer interpretation of the binary representation.
3. `DO` calculate $G = \lambda V$ `WHILE` G does not equal any of the stored baby steps. (Using a binary search)
4. Recover d from μ and λ such that $\mathtt{Baby}_\mu = \lambda V$ and hence recover b.

The algorithm requires storage of $O(q^{1/4})$ large integers, but is guaranteed to recover b in time $O(q^{1/4})$.

Using the Pollard Lambda method one can obtain the same effect with constant storage but only in expected run time $O(q^{1/4})$, see [11] for details.

5 Experiments

In this section we outline the process of recovering the static private key w_B in the case of MQV, both in theory and in practice. In each case, we provide evidence of success through probability calculation and experiment respectively.

The following calculation does not rely on any idealistic model; it comes from results on Babai's algorithm. However it hugely underestimates the effectiveness of the algorithm in practice. We have that combining (3) and (5), it is sufficient that

$$2^l > 2^{(p+f)/n+1}(1 + (n+1)^{1/2} 2^{(n+1)/4})$$

to recover a vector of the form $h + w(z, \beta)$ where $|\beta|_q \leq 2^{\lceil f/2 \rceil}$ with probability greater than $1 - 2^{-p}$.

Alternatively, using a $\mathtt{CVP}_\infty$ oracle model, we can recover a vector $\boldsymbol{v}$ which differs from the hidden vector $\boldsymbol{h}$ by no more than $q/2^l$ in the infinity norm. Applying (5), we expect a lattice attack with $\left\lceil \frac{f+p}{n} \right\rceil + 2$ bits known to recover a vector of the form $\boldsymbol{h} + \boldsymbol{w}(\boldsymbol{z}, \beta)$ where $|\beta|_q \leq 2^{\lceil f/2 \rceil}$ with probability greater than $1 - 2^{-p}$.

Since, $|\beta| \leq 2^{\lceil f/2 \rceil}$, we have that $\lfloor 2^{l+1} h_{n+1} \rfloor_q$ and b agree on the upper half of their bits. Consequently we can use the BSGS/Pollard Lambda algorithm as described above to recover b. Under the $\mathtt{CVP}_\infty$ oracle model, we can then break the protocol given 3 bits per nonce for 170 messages using a 160 bit prime with high probability.

However, no such $\mathtt{CVP}_\infty$ oracle exists, hence in practice we need to use a method which is not guaranteed to find a closest vector with respect to the ∞-norm. In practice such methods are based on the LLL algorithm [8]. To see how successful we can actually be we implemented the following experiment in

C++/LiDIA. For each experiment, we used a newly generated 160 bit prime q and as an alternative to Babai's algorithm, the close vectors were found by embedding the matrix M and the vector $\boldsymbol{u}$ in the square matrix:

$$N = \begin{pmatrix} M & 0 \\ \boldsymbol{u} & 1 \end{pmatrix}. \tag{6}$$

This is a standard technique, see [9] and [10]. The vector $(\boldsymbol{u} - \boldsymbol{h}, 1)$ is then a short vector in the lattice spanned by the rows of N. To recover this vector we used two successive LLL reductions. The first being the Schnorr-Euchner implementation [12] and the second being the de Weger implementation [13]. It was found that applying Schnorr-Euchner on its own did not often produce a suitably reduced basis, whilst applying the de Weger implementation on its own was often too costly. The following table describes our experiments and the resulting success rate on a set of twenty runs.

Bits Known	Number of Protocol Runs	Success Rate
4	30	0.60
4	40	0.50
4	50	0.90
4	60	0.90
5	20	0.55
5	25	0.85
5	30	1.00
6	16	0.40
6	18	0.90
6	20	1.00

A similar attack can also be applied if the bits known are the l least significant of $s_B^{(i)} - r_B^{(i)}$. Say $s_B^{(i)} - r_B^{(i)} = 2^l y_i + c_i$ where y_i is the only unknown and $|y_i| < q/2^l$. Let $\tau = (2^l)^{-1} \pmod q$. We then have a hidden number problem as above with $(t_i, u_i) = (\tau \overline{R}_B^{(i)}, \tau c_i)$ and $(\gamma, a_1, a_2) = (\tau, 2^{\lceil f/2 \rceil}, 2^{\lceil f/2 \rceil + 1} - 1)$. From our theory, we expect this to be as successful as our initial case.

6 Conclusion

We have shown MQV to be insecure with keys of size less than 2^{320} if the attacker is able to obtain a small number of bits of each ephemeral secret generated by his victim. For a standard keylength of 2^{160}, the 4 most significant bits were sufficient to recover the key. Using the $\texttt{CVP}_\infty$ model as a guide, we expect 3 bits may be sufficient with a more suitable l_∞-close vector finding algorithm. Although we cannot improve upon the $\texttt{CVP}_\infty$ oracle, we should recall that we calculated this taking upper bounds. In an ideal model where the lattice reduction uses all the information it is given perfectly, we heuristically expect $l = \lceil f/n \rceil$ to be necessary and sufficient, making $l = 1$ possible for a 160 bit key.

We note that our recovery of only half the bits in the first stage of our attack is not unique to our hidden number problem scenario. It is a fundamental problem to the nature of our equation set (1). We choose the HNP scenario since it it requires less bits known to function partially.

The following modification optimises MQV against this attack:

1. Choose C of minimal size in $\mathbb{Z}$ such that $\overline{R}_A, \overline{R}_B$ can be chosen in $[2^C, 2^{C+1}]$ without comprimising security against other attacks.
2. Choose D of minimal size in $\mathbb{Z}$ such that an elliptic curve group of order D is considered secure against the elliptic curve discrete logarithm problem.
3. Choose the base point P of prime order $q \geq 2^C D$.
4. Instead of defining $\overline{Q}$ for an elliptic curve point $Q = (x, y)$ to mean
$$\lfloor x \rfloor_{2^{\lceil f/2 \rceil}} + 2^{\lceil f/2 \rceil},$$
define it to mean
$$\lfloor x \rfloor_{2^C} + 2^C.$$

In their paper [7] Law, Menezes, Qu, Solinas and Vanstone claimed, in reference to their taking only half the bits of the x coordinate of R_A in $\overline{R}_A$,

"The modification does not appear to affect to security of the protocol."

Ironically, in this setting the modification made the protocol *more* secure. For this reason the idea of choosing some multipliers in a restricted interval is worth considering the future design of protocols that may be vulnerable to lattice based attacks.

References

1. L.Babai. On Lovàsz lattice reduction and the nearest lattice point problem. *Combinatoria*, **6**, 1–13, 1986.
2. D. Boneh and Glenn Durfee. Cryptanalysis of RSA with private key d less than $N^{0.292}$, In *Advances in Cryptology – EuroCrypt '99*, Springer LNCS 1592, 1–11, 1999.
3. D. Boneh, S. Halevi and N. Howgrave-Graham. The Modular Inversion Hidden Number Problem. In *Advances in Cryptology – ASIACrypt 2001*, LNCS 2248, 36-51, 2001
4. D. Boneh and R. Venkatesan. Hardness of computing the most significant bits of secret keys in Diffie-Hellman and related schemes. In *Advances in Cryptology – CRYPTO '96*, Springer LNCS 1109, 129–142, 1996.
5. D. Coppersmith. Small solutions to polynomial equations, and low exponent RSA vulnerabilities. *J. of Cryptology*, **10**, 233–260, 1997.
6. N. Howgrave-Graham and N.P. Smart. Lattice attacks on digital signature schemes. *Designs, Codes and Cryptography*, **23**, 283–290, 2001.
7. L. Law, A. Menezes, M. Qu, J. Solinas and S. Vanstone. An efficient protocol for authenticated key agreement. To appear *Designs, Codes and Cryptography*

8. A.L. Lenstra, H.W. Lenstra, Jr., and L. Lovász. Factoring polynomials with rational coefficients. *Math Ann.*, **261**, 515–534, 1982.
9. P.Q. Nguyen and I.E. Shparlinski. The insecurity of the Digital Signature Algorithm with partially known nonces. To appear *J. Cryptology.*
10. P.Q. Nguyen and J. Stern. Lattice reduction in cryptology: An update. In *Algorithmic Number Theory – ANTS-IV*, Springer LNCS 1838, 85–112, 2000.
11. J.M. Pollard. Monte Carlo methods for index computation $(\mathrm{mod}\, p)$. *Math. Comp.*, **32**, 918–924, 1978.
12. C.P. Schnorr and M. Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. *Math. Programming*, **66**, 181–199, 1994.
13. B.M.M. de Weger. Solving exponential diophantine equations using lattice basis reduction algorithms. *J. Number Theory*, **26**, 325–367, 1987.