Evaluation of Security Level of Cryptography: MY-ELLTY Signature Scheme

Alfred Menezes, Minghua Qu, Doug Stinson, Yongge Wang
Certicom Research
Contact: amenezes@certicom.com

January 15, 2001

1 Summary

MY-ELLTY is an elliptic curve digital signature scheme that provides partial message recovery. If parameters are selected carefully, then the scheme appears to be secure, although no proof of this is known. The main drawback of the proposal is that the choice of the hash functions for the ECMR-192-h, ECMR-OEF-h and ECMR-160-h specifications of MY-ELLTY has output bit lengths that are too small. As a result, the hash function is not collision-resistant and consequently the proposed ECMR-192-h, ECMR-OEF-h and ECMR-160-h schemes are not existentially unforgeable against chosen-message attacks. While there is an obvious way to fix this weakness in the MY-ELLTY signature scheme (increase the number of bits from the hash output), one then has to reduce the size of the recoverable message part.

We also observed that the security proof provided is wrong because it incorrectly applies the forking lemma of Pointcheval and Stern.

2 Protocol specification

The submission proposed the MY-ELLTY scheme with three different parameters: ECMR-192-h, ECMR-OEF-h, and ECMR-160-h. We mainly concentrate on the scheme ECMR-192-h which offers higher security than ECMR-160-h. The ECMR-OEF-h scheme is analogous to ECMR-192-h, except that ECMR-OEF-h is based on elliptic curves over fields $GF(p^m)$ for some prime p and integer m.

2.1 EC Domain parameters

Elliptic Curve domain parameters are comprised of:

1. a field size p, 192-bit prime;
2. two field elements a and b in $GF(p)$ which define the equation of the elliptic curve E over $GF(p)$ (i.e., $y^2 = x^3 + ax + b$);
3. two field elements x_G and y_G in $GF(p)$ which define a finite point $G = (x_G, y_G)$ of prime order in $E(GF(p))$;
4. the order q of the point G.

2.2 MY-ELLTY key pairs

An entity A's key pair is associated with a particular set of EC domain parameters $D = (p, a, b, q, G)$.

MY-ELLTY KEY PAIR GENERATION. Each entity A does the following:

1. Select a random or pseudorandom integer x in the interval $[1, q-1]$.
2. Compute $Y = xG$.
3. A's public key is Y; A's private key is x.

2.3 MY-ELLTY signature generation

To sign a message m, an entity A with domain parameters $D = (p, a, b, q, G)$ and associated key pair (x, Y) does the following:

Input: The elliptic-curve parameters (p, a, b, q, G), the signer's secret key x $(0 < x < q)$, and a message m of length at least 96 bits.

Output: A signed message $r\|s\|m_{nr}$, where $r \in \{0,1\}^{192}$, $s \in GF(q)$, and m_{nr} is the m with its 96 leftmost bits deleted.

Procedure:

1. Let the first 96 bits of m be denoted by m_{re}, and the remainder of m be m_{nr}.
2. Compute $h = H(m)$, where H is the SHA-1 hash function with outputs truncated to 96 bits.
3. Concatenate m_{re} and h, and obtain the 192-bit value d $(d = m_{re}\|h)$.
4. Generate a random integer k such that $0 < k < q$.
5. Compute the point $(x_1, y_1) = kG$ on E (using affine coordinates for point representation).
6. Compute $r = d \oplus x_1$.
7. Compute $r' = r \bmod q$. If $r' = 0$, then return to step 4.
8. Compute $s = (r'k - r' - 1)(x+1)^{-1} \bmod q$. If $s = 0$, then return to step 4.
9. (r, s) is the signature.
10. Concatenate the signature (r, s) and m_{nr}, and output $r\|s\|m_{nr}$ as the signed message.

2.4 MY-ELLTY signature verification

To verify A's signed message (r, s, m_{nr}), B obtains an authentic copy of A's domain parameters $D = (p, a, b, q, G)$ and associated public key Y. B then does the following:

Input: The elliptic-curve parameters (p, a, b, q, G), the signer's public key Y, the signed message $r\|s\|m_{nr}$, where $r \in \{0,1\}^{192}$, $s \in GF(q)$, and m_{nr} is a bit string of arbitrary length.

Output: Recovered 192-bit message d or "invalid".

Procedure:

1. Divide the signature message into the first 384-bit signature (r, s) and the remainder m_{nr}.
2. Compute $r' = r \bmod q$.
3. Verify that $r' \neq 0$ and $0 < s < q$; if not, then output "invalid".
4. Compute the point $(x_2, y_2) = ((1 + r' + s)/r')G + (s/r')Y$ on the elliptic curve (using affine coordinates for point representation).
5. Compute $d = r \oplus x_2$.
6. d is the recovered message.
7. Let m_{re} be the first 96 bits of the message d, and h be the remainder.
8. Concatenate m_{re} and m_{nr}, and obtain the value m $(m = m_{re}\|m_{nr})$.
9. Compute $h' = H(m)$.
10. If $h = h'$ output the message m. Otherwise, output "invalid".

3 Security level of cryptographic techniques

The security objective of any signature scheme is to be existentially unforgeable against a chosen-message attack. The goal of an adversary who launches such an attack against a legitimate entity A is to obtain a valid signature on a single message m, after having obtained A's signature on a collection of messages (not including m) of the adversary's choice.

The submitters have provided a proof that the general MY-ELLTY signature scheme is existentially unforgeable against chosen-message attack in the random oracle model, i.e., when the hash function is modelled by a random function. However, the proof is wrong because it invokes the forking lemma incorrectly. In order to apply the forking lemma, it is required that the commitment kG cannot be modified after receiving the challenge. This is guaranteed by hashing both kG and the message m, and not only m.

In the submitted version of ECMR-192-h signature scheme, only 96 bits (80 bits in ECMR-160-h) of the hash output of SHA-1 are used in the signature generation process. Thus, since finding a collision of the truncated hash function takes about 2^{48} steps, the security of the signature scheme is at most 48 bits. Hence MY-ELLTY ECMR-192-h (and also MY-ELLTY ECMR-160-h and MY-ELLTY ECMR-OEF-h) is not secure against existential forgery attacks. For further discussion of why collision-resistance of the hash function is crucial for security, see §4.1.

The obvious way to fix this weakness in the MY-ELLTY signature scheme is to increase the number of bits from the hash output. However, one then has to reduce the size of the recoverable message part m_{re}.

4 Security level of cryptographic primitive functions

4.1 Attacks on the hash function

DEFINITION. A *(cryptographic) hash function H* is a function that maps bit strings of arbitrary lengths to bit strings of a fixed length t such that:

1. H can be computed efficiently;
2. (*preimage resistance*) For y selected uniformly at random from $\in \{0,1\}^t$ it is computationally infeasible to find a bit string x such that $H(x) = y$.
3. (*second preimage resistance*) Given x_1, it is computationally infeasible to find a different bit string x_2 such that $H(x_1) = H(x_2)$.
4. (*collision resistance*) It is computationally infeasible to find distinct bit strings x_1 and x_2 such that $H(x_1) = H(x_2)$.

SECURITY REQUIREMENTS. The following explains how attacks on MY-ELLTY can be successfully launched if the hash function H is not collision resistant, or second preimage resistant.

1. If H is not collision resistant, then an entity A may be able to repudiate signatures as follows. A first generates two messages m and m' such that the first 96 bits of m and m' are equal, and $H(m) = H(m')$; such a pair of messages is called a *collision* for H. She then signs m, and later claims to have signed m' (note that every signature for m is also a signature for m').
2. If H is not collision resistant, then an entity B could use the birthday attack to swindle A as follows. B prepares two versions (m_1 and m_2) of a contract, where m_1 is favorable to A and m_2 bankrupts A, and also where the first 96 bits of m_1 and m_2 are equal. B makes

several subtle changes to each document and compares whether $H(m'_1) = H(m'_2)$, where m'_1 (m'_2) is a subtle variant of m_1 (m_2), and where the first 96 bits of m'_1 and m'_2 are equal. When B finds two such variants, B obtains the signature (r,s) of m'_1 from A. Then (r,s) is also a signature of m'_2.

3. If H is not second preimage resistant, then an entity B may be able to forge signatures as follows. B generates a message m, find another message m' such that $H(m) = H(m')$, and obtains the signature (r,s) of m' from A. Then (r,s) is also a signature of m.

IDEAL SECURITY. A t-bit hash function is said to be have *ideal security* [17] if both: (i) given a hash output, producing a preimage (or a second preimage) requires approximately 2^t operations; and (ii) producing a collision requires approximately $2^{t/2}$ operations (this is the best that can be hoped for, in view of the birthday attacks).

Thus, for the MY-ELLTY signature schemes, even for the ideal hash function, an attacker can produce a collision in approximately 2^{48} operations.

5 Security level of cryptographic primitive problem: the elliptic curve discrete logarithm problem

One way in which an adversary can succeed is to compute A's private key d from A's domain parameters (p,a,b,q,G) and public key Q. The adversary can subsequently forge A's signature on any message of its choice.

PROBLEM DEFINITION. The *elliptic curve discrete logarithm problem (ECDLP)* is the following: given an elliptic curve E defined over a finite field $GF(p)$, a point $P \in E(GF(p))$ of order n, and a point $Q = lP$ where $0 \leq l \leq n-1$, determine l.

5.1 Known Attacks

This subsection overviews the algorithms known for solving the ECDLP and discusses how they can be avoided in practice.

1. NAIVE EXHAUSTIVE SEARCH. In this method, one simply computes successive multiples of P: P, $2P$, $3P$, $4P, \ldots$ until Q is obtained. This method can take up to n steps in the worst case.

2. POHLIG-HELLMAN ALGORITHM. This algorithm, due to Pohlig and Hellman [21], exploits the factorization of n, the order of the point P. The algorithm reduces the problem of recovering l to the problem of recovering l modulo each of the prime factors of n; the desired number l can then be recovered by using the Chinese Remainder Theorem.

The implications of this algorithm are the following. To construct the most difficult instance of the ECDLP, one must select an elliptic curve whose order is divisible by a large prime n. Preferably, this order should be a prime or almost a prime (i.e. a large prime n times a small integer h). For the remainder of this section, we shall assume that the order n of P is prime.

3. BABY-STEP GIANT-STEP ALGORITHM. This algorithm is a time-memory trade-off of the method of exhaustive search. It requires storage for about $\sqrt{n}$ points, and its running time is roughly $\sqrt{n}$ steps in the worst case.

4. POLLARD'S RHO ALGORITHM. This algorithm, due to Pollard [22], is a randomized version of the baby-step giant-step algorithm. It has roughly the same expected running time ($\sqrt{\pi n/2}$ steps) as the baby-step giant-step algorithm, but is superior in that it requires a negligible amount of storage.

   Gallant, Lambert and Vanstone [11], and Wiener and Zuccherato [32] showed how Pollard's rho algorithm can be sped up by a factor of $\sqrt{2}$. Thus the expected running time of Pollard's rho method with this speedup is $(\sqrt{\pi n})/2$ steps.

5. PARALLELIZED POLLARD'S RHO ALGORITHM. Van Oorschot and Wiener [20] showed how Pollard's rho algorithm can be parallelized so that when the algorithm is run in parallel on r processors, the expected running time of the algorithm is roughly $(\sqrt{\pi n})/(2r)$ steps. That is, using r processors results in an r-fold speed-up.

6. POLLARD'S LAMBDA METHOD. This is another randomized algorithm due to Pollard [22]. Like Pollard's rho method, the lambda method can also be parallelized with a linear speedup. The parallelized lambda-method is slightly slower than the parallelized rho-method [20]. The lambda-method is, however, faster in situations when the logarithm being sought is known to lie in a subinterval $[0,b]$ of $[0,n-1]$, where $b < 0.39n$ [20].

7. MULTIPLE LOGARITHMS. R. Silverman and Stapleton [26] observed that if a single instance of the ECDLP (for a given elliptic curve E and base point P) is solved using (parallelized) Pollard's rho method, then the work done in solving this instance can be used to speed up the solution of other instances of the ECDLP (for the same curve E and base point P). More precisely, if the first instance takes expected time t, then the second instance takes expected time $(\sqrt{2}-1)t \approx 0.41t$. Having solved these two instances, the third instance takes expected time $(\sqrt{3}-\sqrt{2})t \approx 0.32t$. Having solved these three instances, the fourth instance takes expected time $(\sqrt{4}-\sqrt{3})t \approx 0.27t$. And so on. Thus subsequent instances of the ECDLP for a particular elliptic curve become progressively easier. Another way of looking at this is that solving k instances of the ECDLP (for the same curve E and base point P) takes only $\sqrt{k}$ as much work as it does to solve one instance of the ECDLP. This analysis does not take into account storage requirements.

   Concerns that successive logarithms become easier can be addressed by ensuring that the elliptic parameters are chosen so that the first instance is infeasible to solve.

8. SUPERSINGULAR ELLIPTIC CURVES. Menezes, Okamoto and Vanstone [16, 15] and Frey and Rück [9] showed how, under mild assumptions, the ECDLP in an elliptic curve E defined over a finite field $GF(p)$ can be reduced to the ordinary DLP in the multiplicative group of some extension field $GF(q^k)$ for some $k \geq 1$, where the number field sieve algorithm applies. The reduction algorithm is only practical if k is small — this is not the case for most elliptic curves, as shown by Balasubramanian and Koblitz [4]. To ensure that the reduction algorithm does not apply to a particular curve, one only needs to check that n, the order of the point P, does not divide $q^k - 1$ for all small k for which the DLP in $GF(q^k)$ is tractable — in practice, when $n > 2^{160}$ then $1 \leq k \leq 20$ suffices [2].

   An elliptic curve E over $GF(p)$ is said to be *supersingular* if the trace t of E is divisible by the characteristic p of $GF(p)$. For this very special class of elliptic curves, it is known that $k \leq 6$. It follows that the reduction algorithm yields a subexponential-time algorithm for the ECDLP in supersingular curves. For this reason, supersingular curves are explicitly excluded from use in the MY-ELLTY by the above divisibility check.

   More generally, the divisibility check rules out all elliptic curves for which the ECDLP can be efficiently reduced to the DLP in some small extension of $GF(p)$. These include the supersingular elliptic curves and elliptic curves of trace 2 (elliptic curves E over $GF(p)$ for which $\#E(GF(p)) = q - 1$).

9. PRIME-FIELD ANOMALOUS CURVES. An elliptic curve E over $GF(p)$ is said to be *prime-field-anomalous* if $\#E(GF(p)) = p$. Semaev [24], Smart [28], and Satoh and Araki [23] showed how to efficiently solve the ECDLP for these curves. The attack does not extend to any other classes of elliptic curves. Consequently, by verifying that the number of points on an elliptic curve is not equal to the cardinality of the underlying field, one can easily ensure that the Semaev-Smart-Satoh-Araki attack does not apply.

10. CURVES DEFINED OVER A SMALL FIELD. Suppose that E is an elliptic curve defined over the finite field $GF(2^e)$. Gallant, Lambert and Vanstone [11], and Wiener and Zuccherato [32] showed how Pollard's rho algorithm for computing elliptic curve logarithms in $E(GF(2^{ed}))$ can be further sped up by a factor of $\sqrt{d}$ — thus the expected running time of Pollard's rho method for these curves is $(\sqrt{\pi n/d})/2$ steps. For example, if E is a Koblitz curve (see the submission), then Pollard's rho algorithm for computing elliptic curve logarithms in $E(GF(2^m))$ can be sped up by a factor of $\sqrt{m}$. This speedup should be considered when doing a security analysis of elliptic curves whose coefficients lie in a small subfield.

11. CURVES DEFINED OVER $GF(2^m)$, m COMPOSITE. Galbraith and Smart [10], expanding on earlier work of Frey [7, 8], discuss how the Weil descent might be used to solve the ECDLP for elliptic curves defined over $GF(2^m)$ where m is composite (such fields are sometimes called *composite* fields). More recently, Gaudry, Hess and Smart [12] refined these ideas to provide some evidence that when m has a small divisor l, e.g. $l = 4$, the ECDLP for elliptic curves defined over $GF(2^m)$ can be solved faster than with Pollard's

rho algorithm. See also Menezes and Qu [18] for an analysis of the Weil descent attack. In light of these results, it seems prudent to not use elliptic curves over composite fields.

It should be noted that some ECC standards, including the draft ANSI X9.63 [3], explicitly exclude the use of elliptic curves over composite fields. The ANSI X9F1 committee also agreed in January 1999 to exclude the use of such curves in a forthcoming revision of ANSI X9.62.

12. NON-APPLICABILITY OF INDEX-CALCULUS METHODS. Whether or not there exists a general subexponential-time algorithm for the ECDLP is an important unsettled question, and one of great relevance to the security of MY-ELLTY. It is extremely unlikely that anyone will ever be able to *prove* that no subexponential-time algorithm exists for the ECDLP. However, much work has been done on the DLP over the past 24 years, and more specifically on the ECDLP over the past 16 years, and no subexponential-time algorithm has been discovered for the ECDLP. Miller [19] and J. Silverman and Suzuki [27] have given convincing arguments for why the most natural way in which the index-calculus algorithms can be applied to the ECDLP is most likely to fail.

13. XEDNI-CALCULUS ATTACKS. A very interesting line of attack on the ECDLP, called the *xedni-calculus attack* was recently proposed by J. Silverman [25]. One intriguing aspect of the xedni-calculus is that it can be adapted to solve both the ordinary discrete logarithm and the integer factorization problems. However, it was subsequently shown by a team of researchers including J. Silverman (see Jacobson et al. [13]) that the attack is virtually certain to fail in practice.

14. HYPERELLIPTIC CURVES. Hyperelliptic curves are a family of algebraic curves of arbitrary genus that includes elliptic curves. Hence, an elliptic curve can be viewed as a hyperelliptic curve of genus 1. Adleman, DeMarrais and Huang [1] (see also Stein, Müller and Thiel [30]) presented a subexponential-time algorithm for the discrete logarithm problem in the jacobian of a large genus hyperelliptic curve over a finite field. However, in the case of elliptic curves, the algorithm is worse than naive exhaustive search.

15. EQUIVALENCE TO OTHER DISCRETE LOGARITHM PROBLEMS. Stein [29] and Zuccherato [33] showed that the discrete logarithm problem in real quadratic congruence function fields of genus 1 is equivalent to the ECDLP. Since no subexponential-time algorithm is known for the former problem, this may provide further evidence for the hardness of the ECDLP.

5.2 Experimental Results

The best general-purpose algorithm known for the ECDLP is the parallelized version of Pollard's rho algorithm which has an expected running time of $(\sqrt{\pi n})/(2r)$ steps, where n is the (prime) order of the base point P, and r is the number of processors utilized.

CERTICOM'S ECC CHALLENGE. Certicom initiated an ECC challenge [5] in November 1997 in order to encourage and stimulate research on the ECDLP. Their challenges consist of instances of the ECDLP on a selection of elliptic curves. The challenge curves are divided into three categories listed below. In the following, ECCp-*k* denotes a random curve over a field $GF(p)$, ECC2-*k* denotes a random curve over a field $GF(2^m)$, and ECC2K-*k* denotes a Koblitz curve (see the submission) over $GF(2^m)$; *k* is the bitlength of *n*. In all cases, the bitsize of the order of the underlying finite field is equal or slightly greater than *k* (so curves have either prime order or almost prime order).

1. Randomly generated curves over $GF(p)$, where *p* is prime: ECCp-79, ECCp-89, ECCp-97, ECCp-109, ECCp-131, ECCp-163, ECCp-191, ECCp-239, and ECCp-359.
2. Randomly generated curves over $GF(2^m)$, where *m* is prime: ECC2-79, ECC2-89, ECC2-97, ECC2-109, ECC2-131, ECC2-163, ECC2-191, ECC2-238, and ECC2-353.
3. Koblitz curves over $GF(2^m)$, where *m* is prime: ECC2K-95, ECC2-108, ECC2-130, ECC2-163, ECC2-238, and ECC2-358.

RESULTS OF THE CHALLENGE. Escott et al. [6] report on their 1998 implementation of the parallelized Pollard's rho algorithm which incorporates some improvements of Teske [31]. The hardest instance of the ECDLP they solved was the Certicom ECCp-97 challenge. For this task they utilized over 1200 machines from at least 16 countries, and found the answer in 53 days. The total number of steps executed was about 2×10^{14} elliptic curve additions which is close to the expected time ($(\sqrt{\pi n})/2 \approx 3.5 \times 10^{14}$, where $n \approx 2^{97}$). Escott et al. [6] conclude that the running time of Pollard's rho algorithm in practice fits well with the theoretical predictions. They estimate that the ECCp-109 challenge could be solved by a network of 50,000 Pentium Pro 200MHz machines in about 3 months.

6 Recommended parameters

The submission recommended three set of parameters for the scheme. The elliptic curve for ECMR-192-h is over a 192-bit prime field, the elliptic curve for ECMR-160-h is over a 160-bit prime field, and the elliptic curve ECMR-OEF-h is over the 160-bit field $GF(p^5)$ where $p = 2^{32} - 185$.

In Table 1, the security levels of the MY-ELLTY with parameters recommended in the submission are compared with other submitted schemes, with RSA, and with symmetric schemes (e.g., the Advanced Encryption Standard–AES). Some of the comparisons among symmetric key cryptography, RSA security, and ECDSA security are adapted from Lenstra and Verheul [14].

Table 1: Rough Comparison of Security Levels

Symmetric Key Size	RSA Modulus Size	ESIGN Modulus Size	ACE Modulus Size	ECDSA Key Size	MY-ELLTY Key Size
40					160
48					192
76	960	960	960	152	
80 (SKIPJACK)	1024	1024	1024	160	
112 (Triple-DES)	2048	2048	2048	224	
128 (128-bit AES)	3072	3072	3072	256	
192 (192-bit AES)	7680	7680	7680	384	
256 (256-bit AES)	15360	15360	15360	512	

7 Performance comparison

Table 1 presented comparable key lengths of several schemes. Generally, the ESIGN signature scheme is faster than both ECDSA and RSA signature schemes with comparable key lengths. ACE, RSA, and ESIGN have roughly the same public key size for comparable security level. ACE and RSA have roughly the same private key size for comparable security level. ESIGN's private key size is roughly $2/3$ of the ACE (or RSA) private key size for comparable security level. ECDSA and MY-ELLTY have significantly smaller key size for comparable security level. However, the hash function used in MY-ELLTY is not collision resistant, thus it is not secure against attacks on the hash function.

References

[1] L. Adleman, J. DeMarrais and M. Huang, "A subexponential algorithm for discrete logarithms over the rational subgroup of the jacobians of large genus hyperelliptic curves over finite fields", *Algorithmic Number Theory*, Lecture Notes in Computer Science, **877** (1994), Springer-Verlag, 28-40.

[2] ANSI X9.62, *Public Key Cryptography for the Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA)*, 1999.

[3] ANSI X9.63, *Public Key Cryptography for the Financial Services Industry: Elliptic Curve Key Agreement and Key Transport Protocols*, working draft, October 2000.

[4] R. Balasubramanian and N. Koblitz, "The improbability that an elliptic curve has subexponential discrete log problem under the Menezes–Okamoto–Vanstone algorithm", *Journal of Cryptology*, **11** (1998), 141-145.

[5] Certicom ECC Challenge, November 1997, http://www.certicom.com

[6] A. Escott, J. Sager, A. Selkirk and D. Tsapakidis, "Attacking elliptic curve cryptosystems using the parallel Pollard rho method", *CryptoBytes – The Technical Newsletter of RSA Laboratories*, volume 4, number 2, Winter 1999, 15-19. Also available at http://www.rsasecurity.com

[7] G. Frey, "How to disguise an elliptic curve (Weil descent)", talk at ECC '98. Slides available at http://www.cacr.math.uwaterloo.ca

[8] G. Frey, "Applications of arithmetical geometry to cryptographic constructions", *Proceedings of the Fifth International Conference on Finite Fields and Applications*, to appear.

[9] G. Frey and H. Rück, "A remark concerning *m*-divisibility and the discrete logarithm in the divisor class group of curves", *Mathematics of Computation*, **62** (1994), 865-874.

[10] S. Galbraith and N. Smart, "A cryptographic application of Weil descent", *Codes and Cryptography*, Lecture Notes in Computer Science, **1746** (1999), Springer-Verlag, 191-200.

[11] R. Gallant, R. Lambert and S. Vanstone, "Improving the parallelized Pollard lambda search on binary anomalous curves", to appear in *Mathematics of Computation*.

[12] P. Gaudry, F. Hess and N. Smart, "Constructive and destructive facets of Weil descent on elliptic curves", preprint, January 2000. Available from http://www.hpl.hp.com/techreports/2000/HPL-2000-10.html

[13] M. Jacobson, N. Koblitz, J. Silverman, A. Stein and E. Teske, "Analysis of the xedni calculus attack", *Designs, Codes and Cryptography*, **20** (2000), 41-64.

[14] A. Lenstra and E. Verheul. Selecting cryptographic key sizes. Distributed in the *3rd workshop on Elliptic Curve Cryptography (ECC 99)*. Available from http://www.cryptosavvy.com/

[15] A. Menezes, *Elliptic Curve Public Key Cryptosystems*, Kluwer Academic Publishers, Boston, 1993.

[16] A. Menezes, T. Okamoto and S. Vanstone, "Reducing elliptic curve logarithms to logarithms in a finite field", *IEEE Transactions on Information Theory*, **39** (1993), 1639-1646.

[17] A. Menezes, P. van Oorschot and S. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1997.

[18] A. Menezes and M. Qu, "Analysis of the Weil descent attack of Gaudry, Hess and Smart", *Proceedings of RSA 2001*, 2001, to appear.

[19] V. Miller, "Uses of elliptic curves in cryptography", *Advances in Cryptology – Crypto '85*, Lecture Notes in Computer Science, **218** (1986), Springer-Verlag, 417-426.

[20] P. van Oorschot and M. Wiener, "Parallel collision search with cryptanalytic applications", *Journal of Cryptology*, **12** (1999), 1-28.

[21] S. Pohlig and M. Hellman, "An improved algorithm for computing logarithms over $GF(p)$ and its cryptographic significance", *IEEE Transactions on Information Theory*, **24** (1978), 106-110.

[22] J. Pollard, "Monte Carlo methods for index computation mod p", *Mathematics of Computation*, **32** (1978), 918-924.

[23] T. Satoh and K. Araki, "Fermat quotients and the polynomial time discrete log algorithm for anomalous elliptic curves", *Commentarii Mathematici Universitatis Sancti Pauli*, **47** (1998), 81-92.

[24] I. Semaev, "Evaluation of discrete logarithms in a group of p-torsion points of an elliptic curve in characteristic p", *Mathematics of Computation*, **67** (1998), 353-356.

[25] J. Silverman, "The xedni calculus and the elliptic curve discrete logarithm problem", *Designs, Codes and Cryptography*, **20** (2000), 5-40.

[26] R. Silverman and J. Stapleton, Contribution to ANSI X9F1 working group, 1997.

[27] J. Silverman and J. Suzuki, "Elliptic curve discrete logarithms and the index calculus", *Advances in Cryptology – Asiacrypt '98*, Lecture Notes in Computer Science, **1514** (1999), Springer-Verlag, 110-125.

[28] N. Smart, "The discrete logarithm problem on elliptic curves of trace one", *Journal of Cryptology*, **12** (1999), 193-196.

[29] A. Stein, "Equivalences between elliptic curves and real quadratic congruence function fields", *Journal de Théorie des Nombres de Bordeaux*, **9** (1997), 75-95.

[30] A. Stein, V. Müller and C. Thiel, "Computing discrete logarithms in real quadratic congruence function fields of large genus", *Mathematics of Computation*, **68** (1999), 807-822.

[31] E. Teske, "Speeding up Pollard's rho method for computing discrete logarithms", *Algorithmic Number Theory*, Lecture Notes in Computer Science, **1423** (1998), Springer-Verlag, 541-554.

[32] M. Wiener and R. Zuccherato, “Faster attacks on elliptic curve cryptosystems”, *Selected Areas in Cryptography*, Lecture Notes in Computer Science, **1556** (1999), Springer-Verlag, 190-200.

[33] R. Zuccherato, “The equivalence between elliptic curve and quadratic function field discrete logarithms in characteristic 2”, *Algorithmic Number Theory*, Lecture Notes in Computer Science, **1423** (1998), Springer-Verlag, 621-638.