POINT COMPRESSION FOR KOBLITZ ELLIPTIC CURVES

P. N. J. EAGLE AND S. D. GALBRAITH

ABSTRACT. Elliptic curves over finite fields have applications in public key cryptography. A Koblitz curve is an elliptic curve E over $\mathbb{F}_2$; the group $E(\mathbb{F}_{2^n})$ has convenient features for efficient implementation of elliptic curve cryptography.

Wiener and Zuccherato and Gallant, Lambert and Vanstone showed that one can accelerate the Pollard rho algorithm for the discrete logarithm problem on Koblitz curves. This implies that when using Koblitz curves, one has a lower security per bit than when using general elliptic curves defined over the same field. Hence for a fixed security level, systems using Koblitz curves require slightly more bandwidth.

We present a method to reduce this bandwidth. Our method is appropriate for applications such as Diffie-Hellman key exchange or Elgamal encryption. We show that, with a low probability of failure, our method gives the expected bandwidth for a given security level.

1. INTRODUCTION

Let E be an elliptic curve over $\mathbb{F}_q$. The elliptic curve discrete logarithm problem (ECDLP) is: Given a point $P \in E(\mathbb{F}_q)$ of large prime order ℓ and $Q \in \langle P \rangle$, find $a \in [0, \ell - 1] \subseteq \mathbb{Z}$ such that $Q = [a]P$. Pollard [5] gave algorithms to solve the DLP in a generic group of prime order ℓ using pseudorandom walks. Van Oorschot and Wiener [8] showed how to use distinguished points so that the DLP can be solved in close to the expected $\sqrt{\pi\ell/2}$ group operations.

A Koblitz curve is an ordinary elliptic curve $E : y^2 + xy = x^3 + ax^2 + 1$ over $\mathbb{F}_2$. We consider the group $E(\mathbb{F}_{2^n})$ where n is prime. For certain values of n one obtains group orders of the form $c \cdot \ell$ where c is a small even cofactor and ℓ is a large prime. Koblitz in [4] demonstrated the performance benefits of using the group $E(\mathbb{F}_{2^n})$. The advantage is that point multiplication can be accelerated using the 2-power Frobenius map $\psi(x, y) = (x^2, y^2)$ (see Solinas [7] for more details). Throughout the paper we assume that $\ell^2 \nmid \#E(\mathbb{F}_{2^n})$ so that if $P \in E(\mathbb{F}_{2^n})[\ell]$ then $\psi(P) \in \langle P \rangle$.

Wiener and Zuccherato [9] and Gallant, Lambert and Vanstone [2] showed that one can accelerate the Pollard rho method using equivalence classes. For general elliptic curves one can always use the equivalence relation $P \equiv -P$ (i.e., the equivalence classes $\{P, -P\}$) and therefore solve the ECDLP in a group of order ℓ in expected $\sqrt{\pi\ell/4}$ group operations. For Koblitz curves one can define the equivalence relation

$$P \equiv \pm\psi^i(P)$$

for $0 \leq i < n$. The equivalence classes are of size $2n$. It follows that one can solve the ECDLP in a subgroup of $E(\mathbb{F}_{2^n})$ of order ℓ in expected $\sqrt{\pi\ell/4n}$ group operations.

Definition 1. *An elliptic curve E over $\mathbb{F}_q$ has a k–*bit security level *if the expected running time required by Pollard's methods to solve the ECDLP is greater than 2^k group operations*[1].

When working modulo a large prime q one can find elliptic curves E over $\mathbb{F}_q$ such that $\#E(\mathbb{F}_q) = \ell$ is prime. It follows that one has k-bit security level when $\sqrt{\pi l/4} \approx 0.886\sqrt{q} > 2^k$. This certainly holds if $q \geq 2^{2k+1}$. When working over $\mathbb{F}_{2^n}$ (this is the setting of our paper) the group order is always even so one hopes for $\#E(\mathbb{F}_{2^n}) = 2 \cdot \ell$ where ℓ is prime. It is not proven that there exist elliptic curves over every field $\mathbb{F}_{2^n}$ whose number of points is twice a prime, but this conjecture is widely believed. For k-bit security, by the argument above, one therefore has $n \geq 2k + 2$.

For Koblitz curves over $\mathbb{F}_{2^n}$, due to the equivalence classes, one needs $n \geq 2k + 2 + \log_2(n) \geq 2k + 2 + \log_2(2k)$. It follows that when using Koblitz curves one needs more bits of storage and communication compared with general elliptic curves. An alternative way to view this is that we have lower security per bit

[1]Some authors might define this to be 2^k bit operations, or 2^k operations of the AES function.

than when using general curves over the same field. A natural problem is to achieve bandwidth for Koblitz curves which matches the case of more general curves.

For a given security level k there may not be a prime n close to $2k+2+\log_2(n)$ such that $\#E(\mathbb{F}_{2^n}) = 2 \cdot \ell$ for a large prime ℓ. If this is the case one would probably not use Koblitz curves and our methods give no improvement (though one might be able to use subfield curves over a different field, e.g., $\mathbb{F}_{2^2}$).

Given a point $P = (x_P, y_P) \in E(\mathbb{F}_q)$ one can transmit P by sending x_P and a single bit to determine y_P. In many applications it is possible to ignore y_P altogether and perform cryptography using the equivalence classes $\{P, -P\}$. This often goes under the name of elliptic curve cryptography using x-coordinates only. In other words, when using elliptic curves over prime fields one usually expects $2k+1$ bits or $\lceil (2k+1)/8 \rceil$ bytes bandwidth for k bits of security.

When using elliptic curves over $\mathbb{F}_{2^n}$ there is further potential for compression. Seroussi [6] gave a method to save one bit of the x-coordinate (we recall the details in §4.1). It follows that one can transmit $P \in E(\mathbb{F}_{2^n})$ using n bits, or $n-1$ bits if one ignores y_P and works with the equivalence classes $\{P, -P\}$. The minimal bandwidth for elliptic curve cryptography over $\mathbb{F}_{2^n}$ for security level k is therefore $2k+1$ bits or $\lceil (2k+1)/8 \rceil$ bytes. When using Koblitz curves this becomes $n = 2k+1+\log_2(n)$ bits or $\lceil (2k+1+\log_2(n))/8 \rceil$ bytes.

The main result of this paper is to give a method to reduce the bandwidth when using Koblitz curves. Our method is a generalisation of working with the equivalence class $\{P, -P\}$. Recall that the reason for the overhead is that one can attack the system using the Pollard rho algorithm on a set of equivalence classes

$$[P] = \{\pm\psi^i(P) : 0 \leq i < n\}.$$

Hence it is natural to do cryptography using these equivalence classes. We show with a low probability of failure, one can obtain the desired bandwidth for certain applications. We deal with the $\pm$ by discarding y_P. The equivalence class is then determined by the n-bit string x_P up to rotation (we represent $\mathbb{F}_{2^n}$ using a normal basis). The idea is to rotate x_P so that a certain pattern of bits appears at one end of the string (our proposal looks for a pattern of the form $\texttt{011}\cdots\texttt{110}$). This pattern can then be deleted and just the remaining string sent. An extra bit is saved by using the Seroussi trick.

The plan of the paper is as follows. Section 2 makes the notion of bandwidth overhead more precise and gives some targets to achieve. Section 3 explains how to perform Diffie-Hellman key exchange on equivalence classes. Section 4 gives a technical description and justification of the method. Section 5 analyses how well the method is expected to work in practice.

2. Overhead in Koblitz Curve Cryptosystems

To make the problem precise we need to consider how elliptic curve points are transmitted. We consider three communication models for transmitting binary data.

- (Fixed length bitstring) The receiver expects to get an m–bit string;
- (Fixed length bytestring) The receiver expects to get an m–byte string (this is a special case of the previous one);
- (Variable length bitstring) The receiver expects to get a bit–string of variable length $\leq m$ where the receiver knows that the string has ended by some end–of–transmission (EOT) symbol.

The additional bandwidth required to send a point using a Koblitz curve system, compared with using general curves over $\mathbb{F}_{2^n}$, for an equivalent security parameter k, is called the *overhead*. We denote by r_{bit} the number of additional bits required to be sent (this is the *overhead in a bitstring communication model*), and r_{byte} the number of additional bytes (the *overhead in a bytestring communication model*). When it is clear from the context we will simply refer to these as *bit and byte redundancies*.

Table 1 lists the values of n for which there is an elliptic curve over $\mathbb{F}_{2^n}$ such that $\#E(\mathbb{F}_{2^n}) = c\ell$ where ℓ is a large prime and $c \in \{2, 4\}$. The security level is $k = [\log_2(\sqrt{\pi 2^n/4cn})]$ (where $[x]$ means round to the nearest integer). One expects to achieve this security level using a general elliptic curve over $\mathbb{F}_{2^m}$ with $m = 2k+2$ (ignoring the possibility of Weil descent attacks for this value of m [3]). We therefore have

$$r_{\text{bit}} = n - (2k+2)$$

(for example, with $n = 163$ we have $k = 77$ and so $m = 2 \cdot 77 + 2 = 156$ and $r_{\text{bit}} = 7$). Applying the Seroussi trick in both situations gives $r_{\text{bit}} = (n-1) - (2k+1)$ which is the same overhead. Similarly, the number of

Parameters	n	c	Security level k	r_{bit}	r_{byte}
sect163k1	163	2	77	7	1
sect233k1	233	4	111	9	1
sect239k1	239	4	114	9	1
sect283k1	283	4	136	9	1
sect409k1	409	4	199	9	1
sect571k1	571	4	280	9	1

TABLE 1. Bandwidth overhead when using Koblitz curves.

extra bytes to be transmitted (when using the Seroussi trick) is

$$r_{\text{byte}} = \lceil (n-1)/8 \rceil - \lceil (2k+1)/8 \rceil$$

(for example, with $n = 163$, we have $\lceil (n-1)/8 \rceil = 21$ while $\lceil (2 \cdot 77 + 1)/8 \rceil = 20$ so $r_{\text{byte}} = 1$).

The reason why 9 bits of redundancy can mean only one byte of redundancy is that $2k+1$ is not necessarily a multiple of 8 and so there are already some spare bits in the byte representation.

3. Cryptography using Equivalence Classes

It is already well-known that one can perform elliptic curve cryptography using x-coordinates only (i.e., using equivalence classes of the form $\{P, -P\}$). Our proposal is to extend this idea to equivalence classes of the form $\{\pm\psi^i(P)\}$ for $0 \le i < n$. The goal of this section is to explain, using Diffie-Hellman key exchange as an example, that one can do cryptography with these equivalence classes. The crucial fact is that one can define point multiplication on equivalence classes: For $a \in \mathbb{N}$, one defines $[a][P] = [[a]P]$.

Lemma 1. *The operation $[a][P]$ is well-defined.*

Proof. Let $P_1 \in [P]$ so that $P_1 = \pm\psi^i(P)$ for some i. Since $\pm\psi^i$ is a group homomorphism we have $[a]P_1 = \pm\psi^i([a]P)$ and so $[[a]P_1] = [[a]P]$. □

Definition 2. *Let $\mathcal{S}$ be the set of equivalence classes of points in $E(\mathbb{F}_{2^n})[\ell]$. Let $r > 0$ be an integer. Let $\mathcal{C} : \mathcal{S} \to \{0,1\}^{n-r}$ and $\mathcal{D} : \{0,1\}^{n-r} \to \mathcal{S}$ be functions.*

Let $P \in E(\mathbb{F}_{2^n})[\ell]$ and write $[P]$ for the equivalence class of P. Let $\mathtt{x}' \in \{0,1\}^{n-r}$. If

$$\mathcal{D}\big(\mathcal{C}\big([P]\big)\big) = [P]$$

then we call $\mathcal{C}$ and $\mathcal{D}$ compression and decompression *functions.*

We now show that one can use compression and decompression functions to obtain a compressed Diffie-Hellman key exchange protocol. We assume the system parameters include a Koblitz curve $E(\mathbb{F}_{2^n})$ with point $P \in E(\mathbb{F}_{2^n})$ of prime order ℓ.

Alice picks a random $1 \le a < \ell$, computes $Q_A = [a]P$ and sends $\mathtt{x}'_A = \mathcal{C}\big([Q_A]\big)$ to Bob. Similarly, Bob picks $1 \le b < \ell$, computes $Q_B = [b]P$ and sends $\mathtt{x}'_B = \mathcal{C}\big([Q_B]\big)$ to Alice. Alice computes

$$k_A = \mathcal{C}\big([a]\mathcal{D}(\mathtt{x}'_B)\big)$$

and Bob computes

$$k_B = \mathcal{C}\big([b]\mathcal{D}(\mathtt{x}'_A)\big).$$

Lemma 2. *Alice and Bob compute the same key.*

Proof. Alice computes

$$\begin{aligned} k_A &= \mathcal{C}\big([a]\mathcal{D}(\mathtt{x}'_B)\big) \\ &= \mathcal{C}\big([a]\mathcal{D}\big(\mathcal{C}\big([[b]P]\big)\big)\big) \\ &= \mathcal{C}\big([a][[b]P]\big) \\ &= \mathcal{C}\big([[ab]P]\big) \end{aligned}$$

using the property $\mathcal{D}(\mathcal{C}([P])) = [P]$. It is easy to check that Bob computes the same value. □

Obviously the same ideas can be used for any other cryptosystem which is fundamentally based on Diffie-Hellman key exchange (for example, Elgamal encryption).

4. Short Representatives of Equivalence Classes

In the following subsections we explain the compression and decompression algorithms. Recall that a *normal basis* for $\mathbb{F}_{2^n}$ over $\mathbb{F}_2$ is a vector space basis of the form $\{\beta, \beta^2, \beta^{2^2}, \ldots, \beta^{2^{n-1}}\}$. One represents elements of $\mathbb{F}_{2^n}$ with respect to the normal basis as an n-bit string. The action of the 2-power Frobenius map ψ is simply a rotation to the of the binary string, which is fast to compute.

4.1. **Seroussi's Point Compression for Curves $E/\mathbb{F}_{2^n}$.** We describe Seroussi's method [6] for saving one bit in the representation of points in $E(\mathbb{F}_{2^n})$. Recall that for $\alpha \in \mathbb{F}_{2^n}$ the trace map is defined to be $\mathrm{Tr}(\alpha) = \sum_{i=0}^{n-1} \alpha^{2^i}$. It is well known that $\mathrm{Tr}(\alpha) \in \mathbb{F}_2$, $\mathrm{Tr}(\alpha+\beta) = \mathrm{Tr}(\alpha) + \mathrm{Tr}(\beta)$ and $\mathrm{Tr}(\alpha^2) = \mathrm{Tr}(\alpha)$ for all $\alpha, \beta \in \mathbb{F}_{2^n}$. Seroussi's main result is the following.

Lemma 3. *Let $E/\mathbb{F}_{2^n}$ be defined by the Weierstrass equation*

$$E : y^2 + xy = x^3 + ax^2 + b$$

and $P = (x_P, y_P) \in E(\mathbb{F}_{2^n})$ be of odd prime order ℓ. Then one has

$$\mathrm{Tr}(x_P) = \mathrm{Tr}(a).$$

The following result is standard.

Lemma 4. *When using a normal basis representation of $\mathbb{F}_{2^n}/\mathbb{F}_2$, for an $x \in \mathbb{F}_{2^n}$ represented by the vector $(x_{n-1}, x_{n-2}, \ldots, x_0)$ one has*

$$\mathrm{Tr}(x) = \sum_{i=0}^{n-1} x_i.$$

It follows that given a point $P = (x_P, y_P) \in E(\mathbb{F}_{2^n})$ of odd order one can write x_P in binary with respect to a normal basis for $\mathbb{F}_{2^n}$ and remove the least significant bit before transmission. The receiver obtains a bitstring of length $n-1$ and can append the correct bit so that the sum of the bits is equal to $\mathrm{Tr}(a)$.

4.2. **Compressing Abscissæ of Points on Koblitz Curves.** As mentioned, to compress a point $P = (x_P, y_P)$ we first throw away y_P and then represent x_P, with respect to a normal basis, as an n-bit string. We consider all the rotations of this bitstring.

Definition 3. *Let $n \geq 3$ and let $\mathtt{x} = \mathtt{x}_{n-1}\mathtt{x}_{n-2}\cdots\mathtt{x}_1\mathtt{x}_0$ represent a binary string with $\mathtt{x}_i \in \{0,1\}$. We say $\mathtt{x}$ contains a* right padded run *of length t if and only if*

$$\mathtt{x} = \mathtt{x}_{n-1}\cdots\mathtt{x}_{t+2}\mathtt{011}\cdots\mathtt{110}.$$

In other words, bit 0 and the $(t+1)$–th bit are `0` and the intermediate t bits are `1`.

Lemma 5. *Up to rotation only three strings of length ≥ 3 do not have a right padded run. Namely*

$$\mathtt{11}\cdots\mathtt{11} = (\mathtt{1})^n, \qquad \mathtt{00}\cdots\mathtt{00} = (\mathtt{0})^n \qquad \text{and} \qquad (\mathtt{1})^{n-1} \parallel \mathtt{0}.$$

The first two of these do not correspond to points on a Koblitz curve over $\mathbb{F}_2$ of odd order.

Proof. The first claim is obvious. The final claim follows since the all zero and all one strings correspond to $x = 0, 1 \in \mathbb{F}_2$. Such values satisfy the equation $y^2 + xy = x^3 + ax^2 + 1$ for $y \in \mathbb{F}_2$ or $y \in \mathbb{F}_{2^2}$, and either way correspond to points of even order. □

The compression algorithm can now be described. Given $P = (x_P, y_P)$ of odd order on a Koblitz curve consider all n rotations of the binary string x_P and determine which has the longest right padded run. In the case there are two or more runs of the same length we choose the binary string with lowest lexicographical ordering (i.e., smallest value when the bitstring is interpreted as an integer). Call the binary string x and let t be the length of the run of ones. One has $\mathtt{x} = \mathtt{x}_{n-1}\ldots\mathtt{x}_{t+2}011\ldots10$. In Section 5 we discuss the expected size of t.

We now rotate $\mathtt{x}$ once to the left, to get

$$\mathtt{x}_{n-2} \dots \mathtt{x}_{t+2} 011 \dots 10 \mathtt{x}_{n-1}$$

Applying the Seroussi trick one can delete $\mathtt{x}_{n-1}$.

We first consider the variable length bit string communication model. It is only necessary to send the $n-t-3$ bits $x_{n-2}...x_{t+2}$. When the reciever is given these $n-t-3$ bits, knowing n, she can compute t and thus append the length $t+2$ pattern $01\dots10$. Finally, the receiver obtains the remaining bit using the Seroussi method.

More realistically, we are in a fixed bit or byte communication model. The receiver expects to get a fixed number m of bits (or a fixed number of bytes) and must determine the $n-m$ missing bits. One subtlety here is that the receiver cannot determine t by the data sent to them. Hence the receiver must assume that the missing bits include only a substring of the run $\mathtt{011\cdots110}$. If $n-t-3 \geq m$ then the sender fails to obtain the desired level of compression and the algorithm must terminate (the sender can perhaps repeat the cryptographic protocol with different random choices). Hence the receiver assumes $m \leq n-t-2$. The sender sends $x_{n-2}...x_{t+2}$ and an initial segment of the run $011\dots$ to make up the m bits. The receiver then adds $n-m-2$ ones, followed by a zero, followed the the bit coming from the Seroussi trick.

One easily checks that the compression and decompression functions satisfy Definition 2. In particular, $\mathcal{D}(\mathcal{C}([P])) = [P]$ for any equivalence class $[P]$.

5. Expected Bandwidth Saving

We now analyse what values t can be expected in practice, and thus how effective our method is. Our compression function acts by finding the longest run of ones in the binary expansion of the x-coordinate. Hence, the question is to determine the probability of certain lengths of runs of ones. We assume in our analysis that the x-coordinate of an elliptic curve point is essentially a random n-bit string (the Seroussi trick shows that this is not true, but we do not believe this has much bearing on the issue of lengths of runs of ones).

We first consider a related question: Given a fixed integer r and a fixed binary string x_0 of length r, what is the probability that a uniformly chosen binary string of length n has some rotation such that the r least significant bits are equal to x_0? Clearly, the probability that a randomly chosen n-bit string has r least significant bits equal to x_0 is $1/2^r$. Under the (plainly false) hypothesis that the n rotations of an n bit string are independent random binary strings then, when $n \approx 2^r$, we expect at least one rotation to have the desired property. Similarly, if n is significantly smaller than 2^r then there is no reason to expect any rotations to have the property. Hence, it is clear that $r \approx \log_2(n)$ is the best we can hope for, and that $r = \lfloor \log_2(n) \rfloor$ is unlikely to be achievable.

One feature of our method (in the variable length bitstring case at least) is that we don't just look for a run of ones, but use the fact that a run of ones has zeroes on each end. This allows us to get closer to the desired saving of $\log_2(n)$ bits.

We now present the results of some simulations. We estimate the probability that a random n–bit string can be compressed by removing $t+3$ bits using our method (this probability is most relevant for the variable bitstring communication model). For the given values of n our programme generated random n-bit strings and determined the length t of the longest run of ones. This shows whether compression of a given size $r = t+3$ can be achieved.

The results are given in Table 2. The black coloured cells indicate the value $r = t+3 = \lceil \log_2 n \rceil$ and the grey cells are $r = \lfloor \log_2 n \rfloor$ when these values are different. The case $n = 2^{11}-1 = 2047$ has been included purely for theoretical interest. One sees that, with high probability, 7 bits can be saved when $n = 163$, 8 bits saved when $233 \leq n \leq 283$ and 9 bits saved when $n \geq 409$. Hence, we have a good solution to the original problem.

6. Conclusions

We have given a compression method and have given experimental results that show it works well when the communication model allows variable length bit strings.

We now consider the more standard setting of fixed length bit or byte communications. We use the results of Table 2 to show how successfully the the byte overhead r_{byte} can be reduced to the expected minimal

t	$r = t + 3$	bit–size n						
		163	233	239	283	409	571	2047
3	6	1.00	1.00	1.00	1.00	1.00	1.00	1.00
4	7	0.99	1.00	1.00	1.00	1.00	1.00	1.00
5	8	0.94	0.98	0.98	0.99	1.00	1.00	1.00
6	9	0.74	0.86	0.86	0.90	0.96	0.99	1.00
7	10	0.49	0.61	0.61	0.68	0.80	0.89	1.00
8	11	0.28	0.37	0.38	0.44	0.55	0.67	0.98
9	12	0.15	0.20	0.21	0.25	0.33	0.43	0.64

TABLE 2. Estimated probability that a randomly chosen binary string of length n has, up to rotation, a run of t ones.

n	r_{byte}	Number bits to remove	Success Probability
163	1	3	1.00
233	2	9	0.86
239	1	7	1.00
283	1	3	1.00
409	2	9	0.96
571	1	3	1.00

TABLE 3. Experimental estimates for the probability of successful compression in the fixed byte length model.

value when using Koblitz curves (some values for r_{byte} were given in Table 1). The results depend on n modulo 8, which is why the behaviour is not uniform as n grows. We give our results in Table 3.

It follows that one can achieve the expected number of bytes of communication with overwhelming probability for most values of n, and even for the less good n the method works with reasonably good reliability.

Our results can be extended to other subfield curves (i.e., curves over small fields other than $\mathbb{F}_2$) and hyperelliptic curves. However, the original issue is less serious in these cases (as the equivalence classes are smaller) so we do not give the details.

ACKNOWLDEGEMENTS

The first author is supported by EPSRC grant EP/D069904/1.

REFERENCES

1. Certicom Research, Certicom Corporation, *SEC 2: Recommended Elliptic Curve Domain Parameters*, SECG: Standards For Efficient Cryptography Group, 2000.
2. R. P. Gallant, R. Lambert, and S. A. Vanstone, *Improving the parallelized Pollard lambda search on binary anomalous curves*, Mathematics of Computation, **69** (2000) 1699-1705.
3. P. Gaudry, F. Hess and N. Smart, "Constructive and destructive facets of Weil descent on elliptic curves", *Journal of Cryptology*, 15 (2002) 19-46.
4. N. Koblitz, *CM–curves with good cryptographic properties*, in J. Feigenbaum (ed.), CRYPTO '91, Springer LNCS 576, (1992) 279-287.
5. J. Pollard, *Monte Carlo methods for index computation mod p*, Mathematics of Computation, **32** (1978) 918-924.
6. G. Seroussi, *Compact Representation of Elliptic Curve Points over* $\mathbb{F}_{2^n}$, Tech. report, HP Labs Tech. Report HPL-98-94R1, September 1998.
7. J. A. Solinas, *Efficient Arithmetic on Koblitz Curves*, Designs, Codes and Cryptography, **19**, nos. 2-3 (2000) 195-249.
8. P. C. van Oorschot and M. J. Wiener, *Parallel collision search with cryptanalytic applications*, *J. Crypt.*, **12** (1999) 1-28.
9. M. J. Wiener and R. J. Zuccherato, *Faster Attacks on Elliptic Curve Cryptosystems*, in S. E. Tavares and H. Meijer (eds.), SAC 1998, Springer LNCS 1556 (1999) 190-200.

INFORMATION SECURITY GROUP, MATHEMATICS DEPARTMENT, ROYAL HOLLOWAY UNIVERSITY OF LONDON, EGHAM, SURREY TW20 0EX, UK.