Subject: OFFICIAL COMMENT: ECOH
From: Niels Ferguson <niels@microsoft.com>
Date: Fri, 10 Apr 2009 13:47:35 -0700
To: "hash-function@nist.gov" <hash-function@nist.gov>
CC: "hash-forum@nist.gov" <hash-forum@nist.gov>

Michael Halcrow and myself found a 2nd pre-image attack on ECOH. Our attack requires 2^{143} time for ECOH-224 and ECOH-256, 2^{206} time for ECOH-384, and 2^{287} time for ECOH-512. The attack sets the checksum block to a fixed value and uses a collision search on the elliptic curve points.

I have attached the paper. It should also appear on eprint.iacr.org under 2009/198.

Cheers!

Niels

ecohattack-final.pdf	**Content-Description:**	ecohattack-final.pdf
	Content-Type:	application/pdf
	Content-Encoding:	base64

A Second Pre-image Attack Against Elliptic Curve Only Hash (ECOH)

Michael A. Halcrow
mhalcrow@microsoft.com

Niels Ferguson
niels@microsoft.com

April 6, 2009

Abstract

We present a second pre-image attack on ECOH. Our attack requires 2^{143} time for ECOH-224 and ECOH-256, 2^{206} time for ECOH-384, and 2^{287} time for ECOH-512. The attack sets the checksum block to a fixed value and uses a collision search on the elliptic curve points.

1 An outline of ECOH

We first give a description of the essential elements of ECOH. We restrict ourselves to messages that are an integral number of blocks, which is what we use in our attack. For full details, please refer to the ECOH specifications [1].

ECOH divides the message into blocks, maps each block to an elliptic curve point, and adds these points together with two more points. One additional point contains the padding and depends only on the message length. The second additional point depends on the message length and the exclusive-or of all message blocks.

More formally, given n message blocks $M_0, \ldots, M_{n-1}$ we have

$$\begin{aligned} P_i &:= P(M_i, i) \qquad \text{for } i = 0, \ldots, n-1 \\ X_1 &:= P'(n) \\ X_2 &:= P''(\bigoplus_{i=0}^{n-1} M_i, n) \\ Q &:= \sum_{i=0}^{n-1} P_i + X_1 + X_2 \\ R &:= f(Q) \end{aligned}$$

Here, P is a function that maps a message block and an integer to an elliptic curve point. P' computes the padding point which depends only on the length of M. P'' computes the checksum point X_2 which depends on the exclusive-or of all the message blocks, and on the length of M. Finally, the $n+2$ elliptic curve points are added, and the result Q is passed through an output transformation function f to get the hash result R.

2 General form of the attack

For a second pre-image attack we are given a message M and try to find an M' that hashes to the same message. We will find an M' that results in the same value for Q that occurred in the hash computation of M. This lets us ignore the output transformation.

For this attack, we fix the message length to six blocks: $M' = (M_0,\ M_1,\ M_2,\ M_3,\ M_4,\ M_5)$. We choose K different random values for (M_0, M_1) and define M_2 by $M_2 := M_0 \oplus M_1$. We compute the K corresponding elliptic curve points $P(M_0, 0) + P(M_1, 1) + P(M_2, 2)$ and store them in a list. We then choose K different random values for (M_3, M_4), define $M_5 := M_3 \oplus M_4$, compute $Q - X_1 - X_2 - P(M_3, 3) - P(M_4, 4) - P(M_5, 5)$, and store them in a second list. Note that the target Q is known. X_1 only depends on the length of the message which we have fixed. X_2 depends on the length and the xor of all message blocks, but we choose the message blocks such that the xor is always zero. Thus, X_2 is fixed for all our tries.

If K is larger than the square root of the number of points on the elliptic curve then we expect one collision between the two lists. This gives us a message (M_0, M_1, M_2,

M_3, M_4, M_5) with

$$Q - X_1 - X_2 - \sum_{i=3}^{5} P(M_i, i) = \sum_{i=0}^{2} P(M_i, i)$$

and thus

$$Q = \sum_{i=0}^{5} P(M_i, i) + X_1 + X_2$$

which shows that this message leads to the target value Q and thus is a second pre-image.

The workload of this attack is $2K$ partial hash computations. As this is a direct collision search, well-known techniques can be used to convert this algorithm into one that uses only a limited amount of memory.

3 Actual parameters

ECOH-224 and ECOH-256 use the elliptic curve B-283 with approximately 2^{283} points on the curve. We choose $K = 2^{142}$ and get an attack with complexity 2^{143}.

ECOH-384 uses the elliptic curve B-409 with approximately 2^{409} points on the curve. Choosing $K = 2^{205}$ gives an attack with complexity 2^{206}.

ECOH-512 uses the elliptic curve B-571 with approximately 2^{571} points on the curve. Choosing $K = 2^{286}$ gives an attack with complexity 2^{287}.

4 Discussion

The ECOH authors discuss a possible second pre-image birthday attack in Section 6.2.2 of the ECOH paper. They claim that Wagner's Generalized Birthday Attack does not work because of the checksum block. Our method of choosing the message pieces effectively fixes the checksum block to zero, thereby circumventing this countermeasure.

5 Acknowledgements

We would like to thank Daniel Brown, the principal submitter of ECOH, for his helpful comments and suggestions.

References

[1] Daniel R. L. Brown, Adrian Antipa, Matt Campagna, Rene Struik, "ECOH: the Elliptic Curve Only Hash" `http://ehash.iaik.tugraz.at/uploads/a/a5/Ecoh.pdf`, Submission to NIST, 2008

Subject: Re: OFFICIAL COMMENT: ECOH
From: Daniel Brown <dbrown@certicom.com>
Date: Sat, 11 Apr 2009 16:57:15 -0400
To: Multiple recipients of list <hash-forum@nist.gov>

Thank you Niels and Michael for looking at ECOH.

Although I'm not sure why there is need to have n rather than n/2 bit security for n bit hashes, the ECOH team is working on a way that may achieve this. It will likely involve doubling the bit size of the elliptic curve, and as such may be viewed as an instance of the generalization of ECOH in our submission. Details to follow.

Best regards,

Dan

From: hash-forum@nist.gov
To: Multiple recipients of list
Sent: Fri Apr 10 17:01:48 2009
Subject: OFFICIAL COMMENT: ECOH

Michael Halcrow and myself found a 2nd pre-image attack on ECOH. Our attack requires 2^{143} time for ECOH-224 and ECOH-256, 2^{206} time for ECOH-384, and 2^{287} time for ECOH-512. The attack sets the checksum block to a fixed value and uses a collision search on the elliptic curve points.

I have attached the paper. It should also appear on eprint.iacr.org under 2009/198.

Cheers!

Niels

From: hash-forum@nist.gov on behalf of Daniel Brown [dbrown@certicom.com]
Sent: Tuesday, June 02, 2009 1:31 PM
To: Multiple recipients of list
Subject: OFFICIAL COMMENT: ECOH
Attachments: ECOH2.pdf

Hi All,

This paper tunes the parameters of generalized ECOH in an effort to resist second preimage attacks like Ferguson and Halcrow's.

Best regards,

Dan

ECOH^2

Daniel R. L. Brown

June 1, 2009

Abstract

Parameters in the SHA-3 candidate ECOH are tuned to yield ECOH^2, which has potentially better security and efficiency than ECOH with the original parameters. The elliptic curve bit length is effectively doubled, so that the Ferguson-Halcrow second preimage attack on ECOH becomes worse than a generic hash attack on ECOH^2. Despite the doubling of the curve size, efficiency is not too adversely affected, and is indeed potentially improved, for two main reasons: more message bits are used per point, and elliptic curve twists are used to lessen the number of attempted points per message block.

1 Introduction

ECOH, the elliptic curve only hash, is a submission to NIST's SHA-3 competition. The competition guidelines request that an n-bit hash should provide about n bits of second preimage resistance for short messages.

Ferguson and Halcrow [1] found a second preimage against ECOH that is significantly faster than the goal above from the SHA-3 competition guidelines. For example, they find 143-bit attack second preimage attack against ECOH-256.

The NIST competition guidelines request for submitters to suggest tunable parameters that would allow NIST to adjust the security or performance of the submitted hash functions. At the first SHA-3 workshop, NIST stated that submitters should prepare any adjustments to these recommended parameters, as soon as possible, in anticipation of the fact the SHA-3 candidates accepted to the second round of the competition would be allowed to make such adjustments.

The ECOH submission documentation contains a description of a generalization of ECOH. This generalization may be regarded as providing the tunable parameters requested in the SHA-3 competition guidelines. This document presents an adjustment of the tunable parameters. The primary goal of the adjustment is to improve the security against second preimage attacks. Secondary goals are to improve efficiency and pseudorandomness.

2 Two 2nd Preimage Attacks Against ECOH

This section summarizes some second preimage attacks against ECOH.

2.1 Ferguson-Halcrow Second Preimage Attack

Ferguson and Halcrow [1] found a second preimage attack on ECOH. When an m-bit elliptic curve is used, the Ferguson-Halcrow attack takes about $2^{m/2}$ partial ECOH evaluations. Their attack can be implemented with a negligible amount of memory.

In the ECOH specification and implementations, the parameter m (curve bit size) is chosen to be slightly larger than n (hash bit size). The effect was that the $2^{m/2}$ steps in their attack was considerably smaller than the 2^n steps that the SHA-3 guidelines requested for the minimum number of steps in a second preimage attack.

2.2 A Second Second Preimage Attack

In hindsight, it is noted that in §6.1.2 of the ECOH documentation, a method to find second preimages in about 2^{blen} steps is almost described. Starting from the paragraph containing equation (16), what is described is essentially a second preimage attack, since the message M' is fixed and arbitrary and the possibility of finding a message M colliding with M' is considered. By equation (20), it is noted that one need only find a low degree solution of a given quartic (with the quartic determined by M'). However, in the next paragraph, it is noted that this can be generalized to a two-variable polynomial, which by the theory of Semaev summation polynomials will be octic in each variable, whose low degree solution leads to a colliding message M. Here, by low degree, we mean a polynomial of degree $blen$. The ECOH documentation notes that there is likely to be such a low degree solution. (The reason is by heuristics and counting arguments.)

The ECOH documentation fails to note that by exhaustively searching all low degree values of one of the variables, solving the octic for the second variables, and checking at most eight possible solutions for the second variable for having low degree, that one could find the desired low degree solution and thus colliding message M. The cost of this attack is about 2^{blen} steps, with each step involving the solving of an octic polynomial. This is likely to be faster than the Ferguson-Halcrow attack, and further it is only slightly above the birthday bound of $2^{n/2}$ hash computations, because $blen = n/2$ in some cases of ECOH.

3 Specification

In this section, a version of generalized ECOH is presented. This version of generalized ECOH is called ECOH^2. It may be also be called ECOH2 (especially where superscripts are not available).

3.1 Overview and Motivation

In ECOH^2, the elliptic curves used are defined over finite fields that are, for practical purposes, quadratic extensions of the corresponding finite fields used in ECOH. Occasionally, we also employ quartic extensions. So, for example, the function ECOH-224 uses an elliptic curve defined over the field $\mathbb{F}_{2^{283}}$, and thus the function ECOH^2-224 uses elliptic curves defined over the fields $\mathbb{F}_{2^{566}}$ and $\mathbb{F}_{2^{1132}}$.

Note that a reason for using even degree extension fields, rather than similarly sized prime degree fields, is that by choosing quadratic extension fields of the field used in elliptic curve public key cryptography, some of the field arithmetic implementation code can be shared between the

hash function and the public key operations. This is because efficient implementation of quadratic field arithmetic is possible using tower representation involving the lower field arithmetic.

Note that quartic extension are used to have ECOH^2 adhere as closely as possible to the original ECOH pseudocode, and also to work with elliptic curve twists. Using the twisted curve reduces the number of attempted x-coordinates that must be tried, and therefore the number of field inversions needed.

3.2 Bit Length Parameters

For the purposes of comparison between ECOH and ECOH^2, the notation in the pseudocode of ECOH^2 has been changed: m has been changed to d. Where ECOH used $d = m$, ECOH^2 uses $d = 4m$. So, the parameter d will be quadrupled in ECOH^2, compared to ECOH. The parameters *blen* is also increased. The parameters *ilen* and *clen* remain the same. All these parameters are given in Table 1.

Hash	n	d	*blen*	*ilen*	*clen*
ECOH^2-224	224	1132	384	64	64
ECOH^2-256	256	1132	384	64	64
ECOH^2-384	384	1636	640	64	64
ECOH^2-512	512	2284	768	128	128

Table 1: Parameters for ECOH^2

Note that Table 1, unlike the corresponding table for ECOH, does not specify the elliptic curve and base point. These are specified later in this document.

3.3 Pseudocode

The unified pseudocode in Table 2 for ECOH^2 is the almost same as the code for ECOH: there are only two changes. First is the notation change, discussed above, of switching m to d. The second is that *rightmost* bit in Step 6 has been changed to *constant* bit. This technical change is actually consistent with ECOH, because the rightmost bit was the constant bit in ECOH. In ECOH^2, however, the location of the constant bit has been moved in order to facilitate Step 5.

Note that reusing the pseudocode from ECOH is intended to avoid introducing new attacks and to avoid redoing any cryptanalytic effort that has been done, at least to extent that these are related to the pseudocode.

3.4 Tower Representations

Recall that the finite field used in ECOH were $\mathbb{F}_{2^m}$ where m was prime, more specifically $m \in \{283, 409, 571\}$. Furthermore, finite field elements were represented as bit strings using a polynomial basis representation. This means that the finite field elements were expressed as polynomials:

$$a = a_{m-1}t^{m-1} + \cdots + a_1 t + a_0 \tag{1}$$

where $a_i \in \{0, 1\}$ and $t \in \mathbb{F}_{2^m}$ has an irreducible polynomial, written $f(t)$, of degree m. The polynomials $f(t)$ were specified in FIPS 186-2. Every $a \in \mathbb{F}_{2^m}$ can be expressed uniquely in the

1. Let $N = M\|1\|0^j$, with j chosen to be the smallest nonnegative integer such that the length of N is divisible by *blen*.
2. Parse N into blocks $N_0, \ldots, N_{k-1}$ each of length *blen* bits.
3. Let $O_i = N_i\|I_i$, where I_i is the *ilen*-bit representation of the integer i, for i from 0 to $k-1$.
4. Let $O_k = (\bigoplus_{i=0}^{k-1} N_i)\|I_{mlen}$ where I_{mlen} is the *ilen*-bit representation of the bit length *mlen* of the message M.
5. Let $X_i = (0^{d-(blen+ilen+clen)}\|O_i\|0^{clen}) \oplus C_i$ where C_i is the bit string of length d representing the smallest nonnegative integer c such that X_i represents the x-coordinate x_i of an element of $\langle G \rangle$.
6. Let $P_i = (x_i, y_i)$ be such that the constant bit of y_i/x_i equals the leftmost bit of N_i.
7. Let $Q = \sum_{i=0}^{k} P_i$.
8. Output the n-bit representation of $\lfloor x(Q + \lfloor x(Q)/2 \rfloor G)/2 \rfloor \bmod 2^n$.

Table 2: Unified Pseudocode for ECOH^2

form given by (1). Also, every $(a_{m-1}, \ldots, a_0) \in \{0,1\}^m$ gives rise to a unique element element $a \in \mathbb{F}_{2^m}$ in the form (1). The bit string $(a_{m-1}, \ldots, a_0)$ represents the finite field element a.

When moving to the quadratic extension $\mathbb{F}_{2^{2m}}$ we use what is sometimes called a *tower* representation of the finite field elements. We introduce an element $u \in \mathbb{F}_4$ which does not belong to $\mathbb{F}_{2^m}$ and satisfies equation $u^2 + u + 1 = 0$. Elements of $\mathbb{F}_{2^{2m}}$ are now written as:

$$a = (a_{2m-1}t^{m-1} + \cdots + a_{m+1}t + a_m) + (a_{m-1}t^{m-1} + \cdots + a_0)u. \tag{2}$$

and represented by a bit string as $(a_{2m-1}, \ldots, a_0)$. This may be called a mixed order (t, u)-tower representation, since the polynomials in t are represented in descending-degree order, while those in u are represented in ascending-degree order. The bit a_m is referred to as the constant bit, since it represents the constant coefficient of the polynomial.

Again, every finite field element has a unique representation in the form (2), and every bit string corresponds to some finite field element in this manner (2). As a short form for (2), we sometimes write $a = a_1 u + a_0$, with $a_i \in \mathbb{F}_{2^m}$ and the meaning of a_0 being clear from context.

Note that $\mathbb{F}_{2^m}$ exists as a subfield of $\mathbb{F}_{2^{2m}}$ and in the tower representation above, the bit string representation of an element in $\mathbb{F}_{2^m}$ is postpended by m zero bits to obtain its representation as an element of $\mathbb{F}_{2^{2m}}$.

We also employ a quartic extension. In practice, one would mostly implement ECOH^2 using two elliptic curves over the quadratic extension field, with the curves being known as twists of each other. However, in order to comply with the pseudocode of ECOH in Table 2, the curve and its twist over the quadratic extension can be regarded as subgroups of a larger elliptic curve group defined over the quartic extension field.

To represent quartic extension field elements, we introduce another element $v \in \mathbb{F}_{16}$ satisfying

$v^2 + v + u = 0$. We represent elements of $\mathbb{F}_{2^{4m}}$ using a tower representation, specifically a (t, u, v)-tower representation. So, elements of $\mathbb{F}_{2^{4m}}$ are represented first as polynomials,

$$a = (a_{10} + a_{11}u)v + (a_{00} + a_{01}u) \tag{3}$$

where $a_{ij} \in \mathbb{F}_{2^m}$. To represent this as a bit string, concatenate the bit string representation of the $\mathbb{F}_{2^m}$ elements a_{10}, a_{11}, a_{00}, and a_{01}, in that order. The constant bit is the rightmost bit of a_{00}.

Note that $\mathbb{F}_{2^{2m}}$ exists as a subfield of $\mathbb{F}_{2^{4m}}$ and in the tower representation above, the bit string representation of an element in $\mathbb{F}_{2^{2m}}$ is prepended by $2m$ zero bits to obtain its representation as element of $\mathbb{F}_{2^{4m}}$.

3.5 Curve Definitions

ECOH² uses the curves with equation of the form

$$y^2 + xy = x^3 + b \tag{4}$$

where $b \in \mathbb{F}_{2^{2m}}$ is chosen as the finite field element of $\mathbb{F}_{2^{2m}}$ whose bit string representation is lexicographically least among those curves such that

1. the curve (4) has $4r$ $\mathbb{F}_{2^{2m}}$-rational points for some prime r, and
2. the twist of the curve (4) has $2s$ $\mathbb{F}_{2^{2m}}$-rational points for some prime s.

In other words, the curve has cofactor four, and its twist has cofactor two. Note that the curve

$$y^2 + xy = x^3 + ux^2 + b \tag{5}$$

is a twist of the curve (4). Note that these two criteria are equivalent to the following single criterion:

1. The curve (4) has $8rs$ $\mathbb{F}_{2^{4m}}$-rational points for some primes r and s.

Note that this means that b will have the form

$$b = 1 + uw \tag{6}$$

where, recall $u \in \mathbb{F}_4$ with $u^2 + u + 1 = 0$, and $w \in \mathbb{F}_{2^m}$, with furthermore w being the element in this field with least lexicographic representation such that b meets the criteria above.

For a base point G, we choose an $\mathbb{F}_{2^{4m}}$-rational element of order rs, in the notation above. In other words, we choose a base point of maximal odd order. Among such points we choose the lexicographically least, according to bit representations for the quartic extension field $\mathbb{F}_{2^{4m}}$.

4 Efficiency

This section compares a potential implementation of ECOH² to that of ECOH, and concludes that ECOH² may be faster than ECOH.

4.1 Efficient Order Checking

As in ECOH, the definition of ECOH^2 requires the points derived from the message blocks to have odd order. In the submitted ECOH implementation this was achieved virtually for free, by correctly chosen the parity of the incremented counter.

Because ECOH^2 works over a quartic extension, the situation is slightly different. In general, given a point $P = (x, y)$, with $x \in \mathbb{F}_{2^{2m}}$, it will have odd order if and only if it has a $\mathbb{F}_{2^{4m}}$-rational quarter, that is, a point $Q = (u, v)$ such that $4Q = P$ and $u, v \in \mathbb{F}_{2^{4m}}$. Furthermore, if $y \notin \mathbb{F}_{2^{2m}}$, then it suffices for P to have a half $R = (r, s)$, such that $2R = P$ and $(r, s) \in \mathbb{F}_{2^{4m}}$. Also, if P has odd order, then whether or not $y \in \mathbb{F}_{2^{2m}}$, it will have a half $R = (r, s)$ such that $r \in \mathbb{F}_{2^{2m}}$.

Assuming that $P = (x, y)$ is on the curve, then P having a half $R = (r, s)$ such that $r \in \mathbb{F}_{2^{2m}}$ is equivalent to the following condition: there exists $z \in \mathbb{F}_{2^{2m}}$ such that $z^2 + z = x$. In this case $r = y + xz$.

Writing $x = x_1 u + x_0$ and $z = z_1 u + z_0$, then $x = z^2 + z$ in $\mathbb{F}_{2^{2m}}$ is equivalent to the pair of equations in $\mathbb{F}_{2^m}$:

$$x_1 = z_1^2 + z_1 \tag{7}$$
$$x_0 = z_0^2 + z_0 + z_1^2 \tag{8}$$

Letting T be the trace function from $\mathbb{F}_{2^m}$ to $\mathbb{F}_2$, the system of equations has a solution if and only the following pair of equations hold:

$$T(x_1) = 0 \tag{9}$$
$$T(x_0 + z_1^2) = 0 \tag{10}$$

The second equation, however, is redundant because when $T(x_1) = 0$, there will always be two valid solutions for z_1 differing by 1, and exactly one of these two solutions will cause (10) to be true. Therefore, for determining that there exists a half, it suffices to check that $T(x_1) = 0$.

In the cases where one has to check for a quarter, that is, when $y \in \mathbb{F}_{2^{2m}}$, we need to check if the half $R = (r, s)$ has a half. As we saw above, writing $r = r_1 u + r_0$ this equivalent to $T(r_1) = 0$. But $r = y + xz$. So, after solving for y, and also computing $z = H(x)$, where H is the half-trace function on $\mathbb{F}_{2^m}$, which, recall, is usually defined as:

$$H(z) = z + z^4 + \cdots + z^{4^{(m-1)/2}}, \tag{11}$$

which has the property that $H(z)^2 + H(z) = z + T(z)$, for $z \in \mathbb{F}_{2^m}$. Note that $t = H(z)$ and $t = H(z) + 1$ give all solutions to the polynomial equation $t^2 + t = z$.

4.2 Efficient y Recovery

Recall that ECOH and ECOH^2 map message bits into the x-coordinate of an elliptic curve point. These points must then be summed, which requires knowing their y-coordinates. We first review how the submitted implementation of ECOH efficiently calculated these y-coordinates, then review how one might do so for ECOH^2.

Given a value for x, one computes $y = x(\beta + H(x + a + b/x^2))$ where $\beta \in \{0, 1\}$ is chosen as the sum of the rightmost bit of $H(x + a + b/x^2)$ and the leftmost bit of the padded message block N_i from which x is derived.

In ECOH^2, one would recover y slightly differently. As usual, we can divide the elliptic curve equation by x^2 to formulate the equation:

$$(y/x)^2 + (y/x) = x + b/x^2 \tag{12}$$

Here $x + b/x^2 \in \mathbb{F}_{2^{2m}}$ and we shall attempt to find solve for $y/x \in \mathbb{F}_{2^{4m}}$. Let $z = y/x$ and let $w = x + b/x^2$. Then we want to solve $z^2 + z = w$. Note that there will be exactly two solutions, differing by one. Let $z = (z_0 + z_1 u) + (z_2 + z_3 u)v$ and $w = w_0 + w_1 u$, with $z_i, w_i \in \mathbb{F}_{2^m}$, and recall that $u \in F_4$ and $v \in \mathbb{F}_{16}$ with $u^2 + u = 1$ and $v^2 + v = u$. Also, for convenience, define a function F such that $F(s) = s^2 + s$ for any s. The equation to solve is now $F(z) = w$. Note that F is an additive function. Now we can compute:

$$\begin{aligned} F(z) &= F(z_0) + F(z_1 u) + F(z_2 v) + F(z_3 uv) \\ &= F(z_0) + (z_1^2 + F(z_1)u) + (z_2^2 u + F(z_2)v) + (z_3^2(1+v) + F(z_3)uv) \end{aligned} \tag{13}$$

Taking the $\mathbb{F}_{2^m}$ coefficients in the basis $(1, u, v, uv)$ for $\mathbb{F}_{2^{4m}}$, we see by above that equation $z^2 + z = w$, becomes four equations in $\mathbb{F}_{2^m}$:

$$w_0 = F(z_0) + z_1^2 + z_3^2 \tag{14}$$
$$w_1 = F(z_1) + z_2^2 \tag{15}$$
$$0 = F(z_2) + z_3^2 \tag{16}$$
$$0 = F(z_3) \tag{17}$$

The fourth equation implies $z_3 \in \{0, 1\}$. If $z_3 = 1$, then the third equation becomes $F(z_2) = 1$, which has no solution in $\mathbb{F}_{2^m}$ (the solution is $z_2 = u$ in $\mathbb{F}_{2^{4m}}$, of course). Therefore $z_3 = 0$, and $z_2 \in \{0, 1\}$. Recall that in general, $F(z) = w$ has a solution in $\mathbb{F}_{2^m}$ if and only if $T(z) = 0$, and if so the solution z satisfies $z \in \{H(w), H(w) + 1\}$. For the second equation to have a solution, we need to have that $T(w_1 + z_2^2) = 0$. This forces $z_2 = T(w_1)$. In that case, $z_1 = H(w_1 + T(w_1)) + \beta_1$ where $\beta \in \{0, 1\}$. The choice of β_1 will be determined by looking at the first equation, which requires $T(w_0 + z_1^2) = 0$ in order to have a solution. Since the trace function T is additive and invariant under the squaring operation we can just set $\beta_1 = T(w_0 + H(w_1 + T(w_1)))$. After setting β_1 this way to ensure a solution, we can solve for z_0 again using the half-trace function $z_0 = H(w_0 + z_1^2) + \beta_0$, where $\beta_0 \in \{0, 1\}$. The choice of β_0 is determined such that the constant bit of z equals the leftmost bit of the corresponding padded message block.

4.3 Fitting the Twist Into the Quartic

For every $x \in \mathbb{F}_{2^{2m}}$, there exists $y \in \mathbb{F}_{2^{4m}}$ such that (x, y) satisfies the curve equation. For approximately half of all x values, it will be true that $y \in \mathbb{F}_{2^{2m}}$. In this case, the point (x, y) is an $\mathbb{F}_{2^{2m}}$-rational points. Group operations with such points may be done using arithmetic only in $\mathbb{F}_{2^{2m}}$, that is, the quadratic extension field, without having to use the quartic extension field arithmetic.

For the remaining values of x, it will be true that $y \notin \mathbb{F}_{2^{2m}}$. In this case, the point (x, y) can be mapped into another $\mathbb{F}_{2^{2m}}$-rational point $(x, \hat{y})$ that lies on the twist of the curve, where

$$\hat{y} = y + xv \tag{18}$$

Furthermore, by the general theorem that rational maps between elliptic curves, in this case the curve and its twist, produce group homomorphisms, then we can conduct group operations with the $\mathbb{F}_{2^{2m}}$-irrational points (x, y) by using the corresponding $\mathbb{F}_{2^{2m}}$-rational points $(x, \hat{y})$. Therefore, when summing these rational points together, quartic extension field arithmetic can be avoided.

The only time when quartic extension field operations are actually needed in ECOH^2 would therefore appear to be during the final two steps of Table 2.

In the second last step, one can obtain two $\mathbb{F}_{2^{2m}}$-rational points, one on the curve and on the twist. The point on the twist can be mapped back onto the original curve to become $\mathbb{F}_{2^{2m}}$-irrational, but still $\mathbb{F}_{2^{4m}}$-rational. These two points can then be added. (To be precise, in some cases there will only be one point, since one of the curves may have no points landing on it. In this case, treat the point for that curve as the neutral element of the group, that is, the point at infinity.)

In the last step, to avoid extensive use of the quartic extension, one can do something akin to the Chinese remainder theorem. One can decompose the base point as follows. Let

$$G = G_0 + G_1 \tag{19}$$

where G_0 is an $\mathbb{F}_{2^{2m}}$-rational point, and the G_1 is the image of a $\mathbb{F}_{2^{2m}}$-rational point from the twist of the curve. The points G_0 and G_1 can be precomputed. Therefore the single scalar multiplication over the quartic extension can be replaced by two scalar multiplications over the quadratic extension.

4.4 Approximate Algorithm Analysis

Preliminary.

In this section, we make some theoretical comparisons between the potential performance of ECOH and ECOH^2. This comparison makes a number of simplifications and is not based on actual implementations. It should be regarded as preliminary.

Firstly, we note that in the submitted implementations of ECOH, that field inversions and field multiplications (not including squarings) take most of the time. Some other distinct operations needed, such as computing field square, trace and half-trace functions, took much less time because the submitted implementations used the additive nature of these functions and table lookups. We assume that something similar could be done for ECOH^2, and therefore we focus on the total cost of field inversion and field multiplications for the purposes of the comparison.

Let I_1 and M_1 be the cost of field inversion and field multiplication in the field $\mathbb{F}_{2^m}$. Similarly, let I_2 and M_2 be the cost of field inversion and field multiplication in the field $\mathbb{F}_{2^{2m}}$. Then we can get the following estimates:

$$M_2 \approx 3M_1 \tag{20}$$

$$I_2 \approx 3M_1 + I_1 \tag{21}$$

For the first estimate, we use Karatsuba multiplication:

$$(a_1u + a_0)(b_1u + b_0) = ((a_1 + a_0)(b_1 + b_0) + a_0b_0)u + (a_1b_1 + a_0b_0) \tag{22}$$

So, there are three field multiplications in $\mathbb{F}_{2^m}$, namely, a_0b_0 and a_1b_1 and $(a_1 + a_0)(b_1 + b_0)$. This multiplication also involves four $\mathbb{F}_{2^m}$ field additions, but in our simplified analysis we shall ignore these (slightly exaggerating the benefit to ECOH^2). Of course, a more efficient algorithm for

multiplication, such as, perhaps, some kind of combing method, would only give a lower estimate for M_2 and thereby help ECOH² further.

For the second estimate, we use the following formula:

$$(a_1u + a_0)^{-1} = (a_1^2 + a_0a_1 + a_0^2)^{-1}(a_1u + a_0 + a_1) \tag{23}$$

So, there is one field inversion in $\mathbb{F}_{2^m}$, namely, $b = (a_1^2 + a_0a_1 + a_0^2)^{-1}$ and three field multiplications in $\mathbb{F}_{2^m}$, namely, a_0a_1 and ba_1 and $b(a_0 + a_1)$. There are also, two squarings in $\mathbb{F}_{2^m}$ and three additions in $\mathbb{F}_{2^m}$, but in our simplified comparison, we shall ignore the cost of these operations (slightly exaggerating the benefit to ECOH²). Of course, a more efficient for field inversion, such as, perhaps, some variant of the extended Euclidean algorithm, would only give a lower estimate for I_2, and thereby help ECOH² further.

We focus the rest of our algorithm analysis on long messages.

In the submitted ECOH implementation, for each point contributing to the total point Q, there was needed an average of three field inversions and five field multiplications. In terms of message length in bits, the cost per bit of message was approximately:

$$\frac{3I_1 + 5M_1}{blen_1} \tag{24}$$

where now $blen_1$ indicates the value of *blen* specific to ECOH. We will use $blen_2$ for the corresponding value of *blen* in ECOH². Note that this estimate ignores the padding bits, the final scalar multiplication and the formation of the checksum block, so is not accurate for short messages.

Other implementation strategies for ECOH are also possible. For example, simultaneous inversion replaces each I_1 by $3M_1$, creating a numerator of $14M_1$ above. Also, multiplication by fixed values, such as the curve coefficient can be accelerated by means of precomputation and additivity. Some form of projective or Edwards coordinates may improve efficiency. Bit slicing (suggested by Bernstein for batch elliptic curve operations via personal communication) may improve the efficiency of ECOH. For simplicity of our analysis, though, we compare the submitted implementation for ECOH to how an implementation with a similar strategy would perform for ECOH².

If we look a little closer into ECOH at the source of the inversion and multiplications, we see that, one average each point requires one failed point decompressions, one successful point decompression, and one point addition. Each failed point decompression requires one inversion and one multiplication. Each successful point decompression requires one inversion and two multiplications. Each point addition requires one inversion and two multiplications.

Because ECOH² uses the twisted curve, we expect less failed point decompressions. More precisely, only half the points will have a failed point decompression. Therefore, we expect that per point, on average, we have 1 point addition, 1 successful point decompression, and 0.5 failed point decompressions. Therefore, the average cost per bit for ECOH² on long messages is:

$$\frac{2.5I_2 + 4.5M_2}{blen_2} \tag{25}$$

With the approximation $blen_2 \approx 3blen_1$, and the approximations (20) and (21), we get a cost per bit for ECOH² of

$$\frac{\frac{5}{6}I_1 + 7M_1}{blen_1} \tag{26}$$

This would suggest that ECOH^2 should be faster than ECOH, since the extra $2M_1$ is certainly faster than the saved $\frac{7}{6}I_1$. Indeed, if we approximate $I_1 \approx 3M_1$, then the speedup factor is approximately 1.5. If we approximate $I_1 \approx 2.5M_1$, then the speedup factor is approximately 1.4.

For very short messages, it is the final point multiplication that dominates. As noted earlier, though technically this point multiplication is done over the quartic extension, it can be implemented over the quadratic extension. We expect this step of ECOH^2 to about 12 times slower than the corresponding step of ECOH, because one does 2 point multiplications, each with a point multiplier 2 times as long, and using quadratic field arithmetic, which is about 3 times slower. Therefore, when ECOH^2 is applied to many short messages, there is greater incentive to use extensive precomputation to speed up this step.

5 Security Analysis

This section is preliminary and to be completed.

The doubling of the effective elliptic curve size would suggest that generic group attacks, especially those finding an internal collision or second preimage at the intermediate value Q, such as the Ferguson-Halcrow attack, would take at least approximately $\sqrt{2^{2m}} = 2^m$ group operations. Since $m \geqslant n$, this suggests that cryptanalysis using generic group operations is less efficient than a generic n-bit hash attacks.

Also, the fact that significantly more bits of an elliptic curve point are truncated to produce the final hash in ECOH^2 suggests that distinguishing attacks based on guessing many values of the truncated bits and testing for the suitability as an elliptic curve point would be become much harder for ECOH^2.

Elliptic curves over extension fields have in some cases been shown to be vulnerable to a class of attacks known as Weil descent attacks. Currently, such attacks are not deemed more effective for quadratic extensions than generic group attacks. If they do become effective, they would need to become quite effective to be used to find a collision in ECOH^2.

References

[1] M. A. Halcrow and N. Ferguson. A second pre-image attack against elliptic curve only hash (ECOH). ePrint 2009/168, International Association for Cryptologic Research, 2009. http://eprint.iacr.org/2009/168.