# Point Compression Algorithms for Binary Curves

Julio López and Ricardo Dahab

{jlopez,rdahab}@ic.unicamp.br

Institute of Computing (IC)
UNICAMP

April, 14 2005

# Outline

- Introduction to ECC over $GF(2^m)$
- Background on: trace and halving a point
- Previous methods on point compression
  - IEEE-P1363
  - Seroussi's method
  - King's method
- New Methods
- Conclusions

## Point Compression Problem

Given a point $P = (x, y)$ on an elliptic curve $E$ defined over a finite field $\mathbb{F}_{2^m}$, find a method for representing the point $P$ with less than $2m$ bits.

$$\text{Compress}(P) = ? \quad \text{and} \quad \text{Decompress}(C_P) = ?$$

Main application: for the transmission of elliptic points

(Diffie-Hellman key agreement protocol)

# Elliptic Curve Cryptography

- An *elliptic curve* $E$ over the finite field $\mathbb{F}_{2^m}$ is defined by the equation:

$$E: \ y^2 + xy = x^3 + ax^2 + b \quad (1)$$

where $a, b \in \mathbb{F}_{2^m}$ and $b \neq 0$.

- The set of rational points on $E$:

$$E(\mathbb{F}_{2^m}) := \{(x, y) | \ y^2 + xy = x^3 + ax^2 + b\} \cup \mathcal{O},$$

where $\mathcal{O}$ is the *point at infinity*.

- There is a *chord-and-tangent rule* for adding two points in $E(\mathbb{F}_{2^m})$ to give a third point in $E(\mathbb{F}_{2^m})$.

$$R = P + Q$$

## Elliptic Curve Cryptography /2

- The set of points $E(\mathbb{F}_{2^m})$, together with the "+" operation, forms an abelian group with $\mathcal{O}$ as its identity. It is this group that is used for constructing elliptic curve cryptosystems.
- The **Point Multiplication** operation is: given a point $P$ on an elliptic curve $E$ and an integer $k$, the point multiplication $kP$ is

$$kP := P + P + \cdots + P \quad (k \text{ times}).$$

- The **Elliptic curve discrete logarithm problem** is: given a point $P \in E(\mathbb{F}_{2^m})$ of order $n$, and point $Q \in < P >$, find the integer $k \in [0, n-1]$ such that $Q = kP$.

# Elliptic Curve Cryptography

**Domain parameters** $D = (q, \mathrm{FR}, S, a, b, P, n, h)$ are comprised of:

- The field order $q = 2^m$.
- The FR (field representation) (normal basis or polynomial basis).
- A *seed* $S$ if the elliptic curve was randomly generated.
- Two coefficients $a, b \in \mathbb{F}_{2^m}, b \neq 0$ that define equation (1).
- Two field elements $x_P$ and $y_P$ in $\mathbb{F}_{2^m}$ that define a finite point $P = (x_P, y_P) \in E(\mathbb{F}_{2^m})$. The point $P$ has prime order and is called the *base point.*
- The *order* $n$ of $P$
- The *cofactor* $h = \#E(\mathbb{F}_{2^m})/n$.

## Group Law for Binary Curves over $\mathbb{F}_{2^m}$

1. *Identity.* $P + \mathcal{O} = \mathcal{O} + \mathcal{P} = \mathcal{P}$ for all $P \in E(\mathbb{F}_{2^m})$.
2. *Negatives.* If $P = (x, y) \in E(\mathbb{F}_{2^m})$, then $-P = (x, x + y)$.
   Note that $-P$ is a point in $E(\mathbb{F}_{2^m})$. Also, $-\mathcal{O} = \mathcal{O}$.
3. *Point addition.*
   Let $P = (x_1, y_1) \in E(\mathbb{F}_{2^m})$ and $Q = (x_2, y_2) \in E(\mathbb{F}_{2^m})$, where $P \neq \pm Q$. Then $P + Q = (x_3, y_3)$, where

   $$x_3 = \lambda^2 + \lambda + x_1 + x_2 + a \quad \text{and} \quad y_3 = \lambda(x_1 + x_3) + x_3 + y_1,$$

   and $\lambda = (y_1 + y_2)/(x_1 + x_2)$.

# Group Law for Binary Curves over $\mathbb{F}_{2^m}$ /2

4. *Point doubling.*
   Let $P = (x_1, y_1) \in E(\mathbb{F}_{2^m})$, where $x_1 \neq 0$.
   Then $2P = (x_2, y_2)$, where

$$\begin{aligned} x_2 &= \lambda^2 + \lambda + a = x_1^2 + \frac{b}{x_1^2} \\ y_2 &= x_1^2 + \lambda x_2 + x_2 \\ \lambda &= x_1 + \frac{y_1}{x_1} \end{aligned}$$

## The Trace Function

$$Tr : \mathbb{F}_{2^m} \Rightarrow \mathbb{F}_{2^m}$$

- $\mathrm{Tr}(c) := c + c^2 + c^{2^2} + \cdots + c^{2^{m-1}}$.
- $Tr(c) = Tr(c^2) = Tr(c)^2$; in particular $Tr(c) \in \{0, 1\}$.
- Trace is linear: $Tr(c + d) = Tr(c) + Tr(d)$.
- $Tr(x(kP)) = Tr(a)$ for generator $P \in E(\mathbb{F}_{2^m})$.

# Computing the Trace Function

- Normal basis: $\{\beta, \beta^2, ..., \beta^{2^m}\}$

$$c = \sum_{i=0}^{m-1} c_i \beta^{2^i} \Rightarrow Tr(c) = \sum_{i=0}^{m-1} c_i$$

- Polynomial basis: $\{1, x, x^2, ..., x^{m-1}\}$

$$c = \sum_{i=0} c_i x^i \Rightarrow Tr(c) = \sum_{i=0}^{m-1} c_i Tr(x^i)$$

## NIST Recommended Binary Curves

- In 1999, NIST releases a list of 10 binary curves over the finite fields $\mathbb{F}_{2^m}$, $m \in \{163, 233, 283, 409, 571\}$.
- For the random curves $a = 1$ ($\mathrm{Tr}(a) = 1$).
- For the Koblitz curves $a = 1$ for $m = 163$ and $a = 0$ for $m \neq 163$.
- For a polynomial basis representation, the trace can be computed as follows:
  - for $m = 163$, $\mathrm{Tr}(c) = c_0 + c_{157}$
  - for $m = 233$, $\mathrm{Tr}(c) = c_0 + c_{159}$
  - for $m = 283$, $\mathrm{Tr}(c) = c_0 + c_{277}$
  - for $m = 409$, $\mathrm{Tr}(c) = c_0$
  - for $m = 571$, $\mathrm{Tr}(c) = c_0 + c_{561} + c_{569}$.

## Solving a Quadratic Equation over $\mathbb{F}_{2^m}$

$$x^2 + x = c, \;\; c \in \mathbb{F}_{2^m}$$

- Let $m$ be an odd integer. The *half-trace function* $H : \mathbb{F}_{2^m} \Rightarrow \mathbb{F}_{2^m}$ is defined by

$$H(c) = \sum_{i=0}^{(m-1)/2} c^{2^{2i}}.$$

- $H(c)$ is a solution of the equation $x^2 + x = c + Tr(c)$.

- If $Tr(c) = 0$, $H(c)$ and $H(c) + 1$ are the only solutions of the equation $x^2 + x = c$.

# Point Compression Algorithm
# First Solution

$$P = (x, y) \in E(\mathbb{F}_{2^m}), \;\; x \neq 0$$

$$\text{Compress } (P) := x || Tr(x + \tfrac{y}{x})$$

$$|\text{Compress}(P)| = m + 1 \text{ bits}$$

$$y^2 + xy = x^3 + ax^2 + b \;\Rightarrow (x + \frac{y}{x})^2 + (x + \frac{y}{x}) = x^2 + a + \frac{b}{x^2}$$

$$\lambda^2 + \lambda = x^2 + a + \frac{b}{x^2}, \;\; Tr(\lambda) \in \{0, 1\}$$

# Point Compression Algorithm First Solution /2

INPUT: $C_P$.

OUTPUT: $P = (x, y)$.

Decompress($C_P$):

1. $C_p = x || t, \quad t = Tr(x + \frac{y}{x})$.
2. Solve $\lambda^2 + \lambda = x^2 + a + \frac{b}{x^2}$ for $\lambda$.
3. If $(\mathrm{Tr}(\lambda) \neq t)$ then $\lambda = \lambda + 1$.
4. Compute $y = x \cdot (\lambda + x)$.
5. Return $P = (x, y)$.

Note: $y = x \cdot (\lambda + Tr(\lambda) + t + x)$

# Point Compression
# IEEE P1363 Algorithm

$$P = (x, y) \in E(\mathbb{F}_{2^m})$$

$$\text{Compress }(P) := x || \tilde{y}, \quad \tilde{y} : \text{the rightmost bit of } \tfrac{y}{x}$$

$$|\text{Compress}(P)| = m + 1 \text{ bits}$$

$$y^2 + xy = x^3 + ax^2 + b \;\Rightarrow (\frac{y}{x})^2 + \frac{y}{x} = x + a + \frac{b}{x^2}$$

$$\mu^2 + \mu = x^2 + a + \frac{b}{x^2}, \;\tilde{\mu} \in \{0, 1\}$$

# Point Compression
# IEEE P1363 Algorithm /2

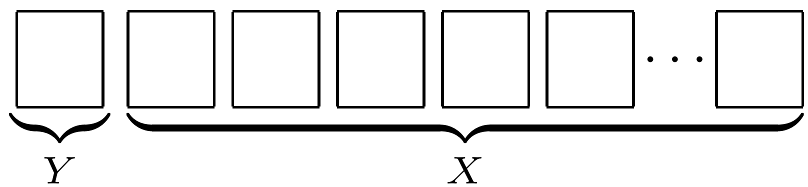
INPUT: $C_P$

OUTPUT: $P = (x, y)$.

Decompress($C_P$):

1. $C_p = x || \tilde{y}$.
2. If $x = 0$ then Return $P = (0, \sqrt{b})$.
3. Solve $\mu^2 + \mu = x + a + \frac{b}{x^2}$ for $\mu$.
4. Compute $y = x \cdot (x + \tilde{\mu} + \tilde{y})$.
5. Return $P = (x, y)$.

# IEEE P1363 Algorithm




$$Y = \begin{cases} \texttt{0x02} & \text{point compressed } x_P = 0 \\ \texttt{0x03} & \text{point compressed } x_P \neq 0 \\ \texttt{0x04} & \text{point not compressed} \end{cases}$$

# Point Compression
# Seroussi's Algorithm, 1998

$$P = (x, y) \in E(\mathbb{F}_{2^m})$$

- $Tr(x) = Tr(a)$
- Let $i$ be the smallest index such $Tr(x^i) = 1$.
  For NIST binary fields $i = 0$.
- $m-1$ bits are required to represent $x$.

$$x = (x_{m-1} \cdots x_i \cdots x_0)$$

$$\dot{x} := (x_{m-1} \cdots x_{i-1} x_{i+1} \cdots x_0)$$

$$\text{Compress } (P) := (x_{m-1} \cdots x_{i-1} \tilde{y} x_{i+1} \cdots x_0)$$

$$|\text{Compress}(P)| = m \text{ bits}$$

# Point Compression
## Seroussi's Algorithm /2

INPUT: $C_P,\ Tr(x) = Tr(a)$.

OUTPUT: $P = (x, y)$.

Decompress($C_P$):

1. From $C_P$ obtains $x_1 = (x_{m-1} \cdots x_{i-1}\ 1\ x_{i+1} \cdots x_0)$ and $y_1 = \tilde{y}$.
2. If $\mathrm{Tr}(x_1) \neq \mathrm{Tr}(a)$ then $x_1 = (x_{m-1} \cdots x_{i-1}\ 0\ x_{i+1} \cdots x_0)$.
3. If $x = 0$ then Return $P = (0, \sqrt{b})$.
4. Solve $\mu^2 + \mu = x + a + \frac{b}{x^2}$ for $\mu$.
5. Compute $y = x \cdot (x + \tilde{\mu} + y_1)$.
6. Return $P = (x, y)$.

## Point Compression
## New Algorithm I, 2004

INPUT: $P = (x, y)$, $Tr(x) = Tr(a)$, $m$ odd.

OUTPUT: $C_P \in \mathbb{F}_{2^m}$.

$$\text{Compress } (P) := C_P = x + Tr(x + \tfrac{y}{x})$$

$$|C_P| = m \text{ bits}$$

Note: $Tr(x + \frac{y}{x}) = Tr(C_P) + Tr(a)$.

## Point Compression
## New Algorithm I /2

INPUT: $C_P,\ Tr(x) = Tr(a),\ m$ odd.

OUTPUT: $P = (x, y)$.

Decompress($C_P$):

1. $x = C_P + Tr(C_P)$.
2. Solve $\lambda^2 + \lambda = x^2 + a + \frac{b}{x^2}$ for $\lambda$.
3. Compute $y = x \cdot (x + Tr(\lambda) + Tr(C_P) + Tr(a))$.
4. Return $P = (x, y)$.

## Point Halving for Binary Curves

- $P = (x, y) \;\Rightarrow\; 2P = (x_2, y_2)$ : let $\lambda = x + y/x$

$$\begin{aligned} x_2 &= \lambda^2 + \lambda + a = x^2 + b/x^2 \\ y_2 &= x^2 + \lambda x_2 + x_2 \end{aligned}$$

- Halving: given $Q = (x_2, y_2)$ find $P = (x, y)$ such $2P = Q$

$$\begin{aligned} x_2 &= \lambda^2 + \lambda + a \quad \text{solve for } \lambda \\ y_2 &= x^2 + \lambda x_2 + x_2 \quad \text{solve for } x \end{aligned}$$

E. Knudsen, 1999 and R. Schroeppel, 2000

## Computing Point Halving

INPUT: $P = (x, \lambda)$, $Tr(a) = 1$, $m$ odd.

OUTPUT: $2P = (x_2, \lambda_2)$.

$$\begin{array}{cc} x & \lambda \\ \uparrow & \uparrow \\ x_2 & \lambda_2 \end{array}$$

$$2P = (x_2, y_2) \Leftrightarrow (x_2, \lambda_2), \quad \lambda_2 = x_2 + y_2/x_2$$

# Computing Point Halving /2

$$Tr(a) = 1, \; m \text{ odd}$$

$$\begin{array}{ccc} x & & \lambda \\ \uparrow & \nearrow & \uparrow \\ x_2 & & \lambda_2 \end{array}$$

1. Solve $\lambda^2 + \lambda = x_2 + a$ for $\lambda$.
2. Compute $T = x_2 \cdot (\lambda + \lambda_2 + x_2 + 1)$.
3. If $\mathrm{Tr}(T) = 1$ then $x = \sqrt{T}$
   else $\lambda = \lambda + x_2, \; x = \sqrt{T + x_2}$.
4. Return $(x, \lambda)$.

# Computing Point Halving /3

$$Tr(a) = 1, \; m \text{ odd}$$

| | |
|---|---|
| *output*: $x$ | $\lambda$ |
| $z = \frac{x^2}{x_2}$ $\uparrow$ | $\uparrow$ |
| *input*: $x_2$ | $\lambda_2$ |
| $\downarrow$ | |
| $x_4$ | |

1. Compute $x_4 = \lambda_2^2 + \lambda_2 + a$.
2. Solve $z^2 + z = x_4 + x_2^2 = (b/x_2^2)$ for $z$.
3. Compute $T = z \cdot x_2$.
4. If $\text{Tr}(T) = 1$ then $x = \sqrt{T}, \; \lambda = z + \lambda_2 + x_2 + 1$ else $x = \sqrt{T + x_2}, \lambda = z + \lambda_2 + x_2$.
5. Return $(x, \lambda)$.

## Point Compression
## New Algorithm II, 2004

INPUT: $P = (x, y)$, $Tr(x) = Tr(a) = 1$, $m$ odd.

OUTPUT: $C_P \in \mathbb{F}_{2^m}$.

$$\text{Compress } (P) := C_P = x + \frac{y}{x}$$

$$|C_P| = m \text{ bits}$$

Note: $Tr(x + \frac{y}{x}) = Tr(C_P)$.

# Point Compression
# New Algorithm II /2

INPUT: $C_P,\ Tr(x) = Tr(a) = 1,\ m$ odd.

OUTPUT: $P = (x, y)$.

Decompress($C_P$):

1. $x_2 = C_P^2 + C_P + a$.
2. Solve $z^2 + z = \frac{b}{x_2^2}$ for $z$.
3. Compute $T = z \cdot x_2$
4. If $Tr(T) = 1$ then $x = \sqrt{T}$ else $x = \sqrt{T + x_2}$.
5. Compute $y = x \cdot (x + C_P \cdot x)$.
6. Return $P = (x, y)$.

## Point Compression
## King's Algorithm, 2004

INPUT: $P = (x, y)$, $Tr(x) = Tr(a)$, $m$ odd.

OUTPUT: $C_P \in \mathbb{F}_{2^m}$.

$$\text{Compress}(P) := C_P = \begin{cases} x & \text{if } Tr(\frac{y}{x}) = 0 \\ \frac{\sqrt{b}}{x} & \text{if } Tr(\frac{y}{x}) = 1 \end{cases}$$

$$|C_P| = m \text{ bits}$$

Note: $Tr(\frac{y}{x}) = Tr(C_P) + 1$.

## Point Compression
## King's Algorithm /2

INPUT: $C_P$, $Tr(x) = Tr(a)$, $m$ odd.

OUTPUT: $P = (x, y)$.

Decompress($C_P$):

1. If $Tr(C_P) = 0$ then $x = C_p$ else $x = \frac{\sqrt{b}}{C_P}$
2. Solve $\mu^2 + \mu = x + a + \frac{b}{x^2}$ for $\mu$.
3. If $Tr(\mu) = Tr(C_P) + 1$ then $y = x \cdot \mu$ else $y = x \cdot \mu + x$.
4. Return $P = (x, y)$.

## Point Compression
## King's Algorithm /3

INPUT: $P = (x, y)$, $Tr(x) = Tr(a) = 0$, $m$ odd.

OUTPUT: $C_P \in \mathbb{F}_{2^m}$.

$$\text{Compress}(P) := C_P = \begin{cases} \dot{x} & \text{if } Tr(\frac{y}{x}) = 0 \\ \dot{\frac{\sqrt{b}}{x}} & \text{if } Tr(\frac{y}{x}) = 1 \end{cases}$$

$$|C_P| = m - 1 \text{ bits}$$

Note: $Tr(x) = Tr(\frac{\sqrt{b}}{x}) = 0$.

# Point Compression
# King's Algorithm /4

INPUT: $P = (x, y)$, $Tr(x) = Tr(a) = 0$, $m \in \{233, 289, 409, 571\}$.

OUTPUT: $C_P \in \mathbb{F}_{2^m}$.

For Koblitz curves recommended by NIST

$$\text{Compress}(P) := C_P = \begin{cases} \dot{x} & \text{if } Tr(\frac{y}{x}) = 0 \\ \frac{\dot{1}}{x} & \text{if } Tr(\frac{y}{x}) = 1 \end{cases}$$

$$|C_P| = m - 1 \text{ bits}$$

# Point Compression Algorithms Summary

$$\text{Compress}(P) = C_P$$

| Method | $C_P$ | Bits |
|---|---|---|
| IEEE P1363 | $x\|\|\tilde{y}$ | $m+1$ |
| Seroussi | $\dot{x}\|\|\tilde{y}$ | $m$ |
| King | $\begin{cases} x & \text{if } Tr(\frac{y}{x}) = 0 \\ \frac{\sqrt{b}}{x} & \text{if } Tr(\frac{y}{x}) = 1 \end{cases}$ | $m$ or $m-1$ |
| New I | $x + Tr(x + \frac{y}{x})$ | $m$ |
| New II* | $x + \frac{y}{x}$ | $m$ |

*$Tr(a) = 1$.

## References


- IEEE P1363 *Appendix A.*
  `htpp://www.grouper.org/groups/1363`
- G. Seroussi, " Compact Representation of Elliptic Curve Points over $\mathbb{F}_{2^n}$", *HP Labs Technical Reports*,
  `htpp://www.hp1.hp.com/techreports/98/HPL-98-94R1.html`.
- B. King, " A point Compression Method for Elliptic Curve Defined over $GF(2^n)$", PKC 2004, LNCS 2947, pp. 333-345, 2004.
- K. Fong, D. Hankerson, J. López and A. Menezes," Field Inversion and Point Halving revisited", *IEEE Transaction on Computers*, Vol 53, no. 8, pp. 1047-1059, 2004.