Small high-security public-key encryption and signatures

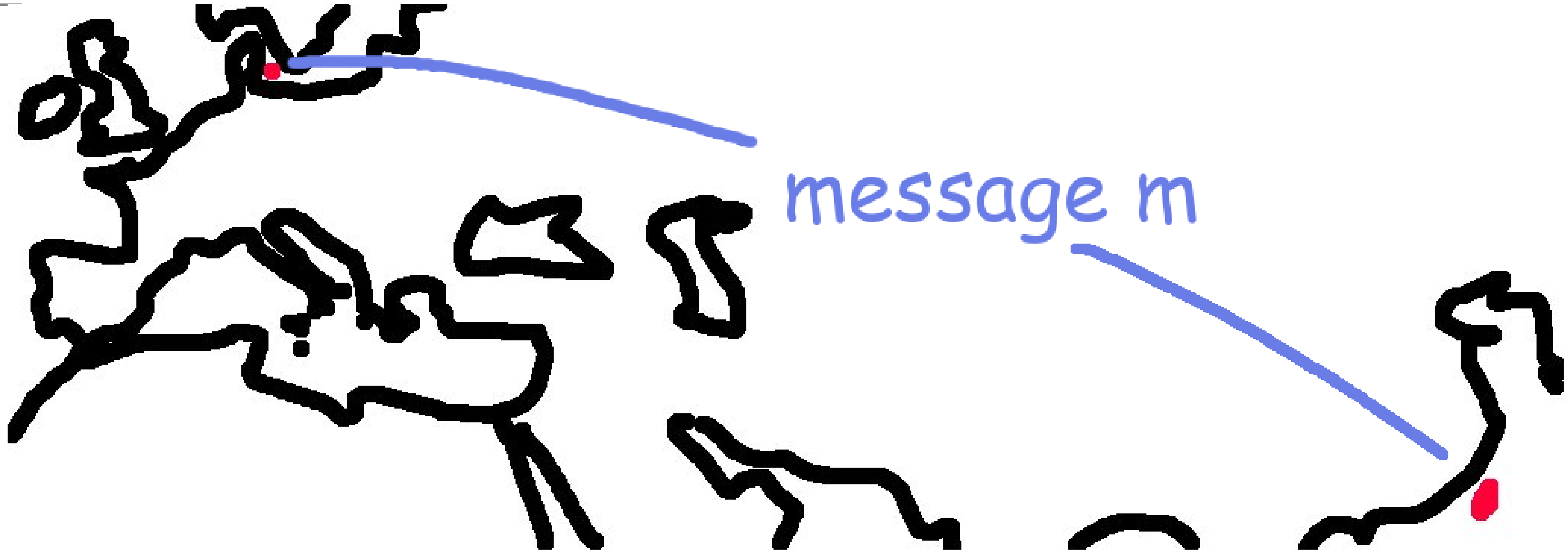
Tanja Lange

Technische Universiteit Eindhoven

tanja@hyperelliptic.org

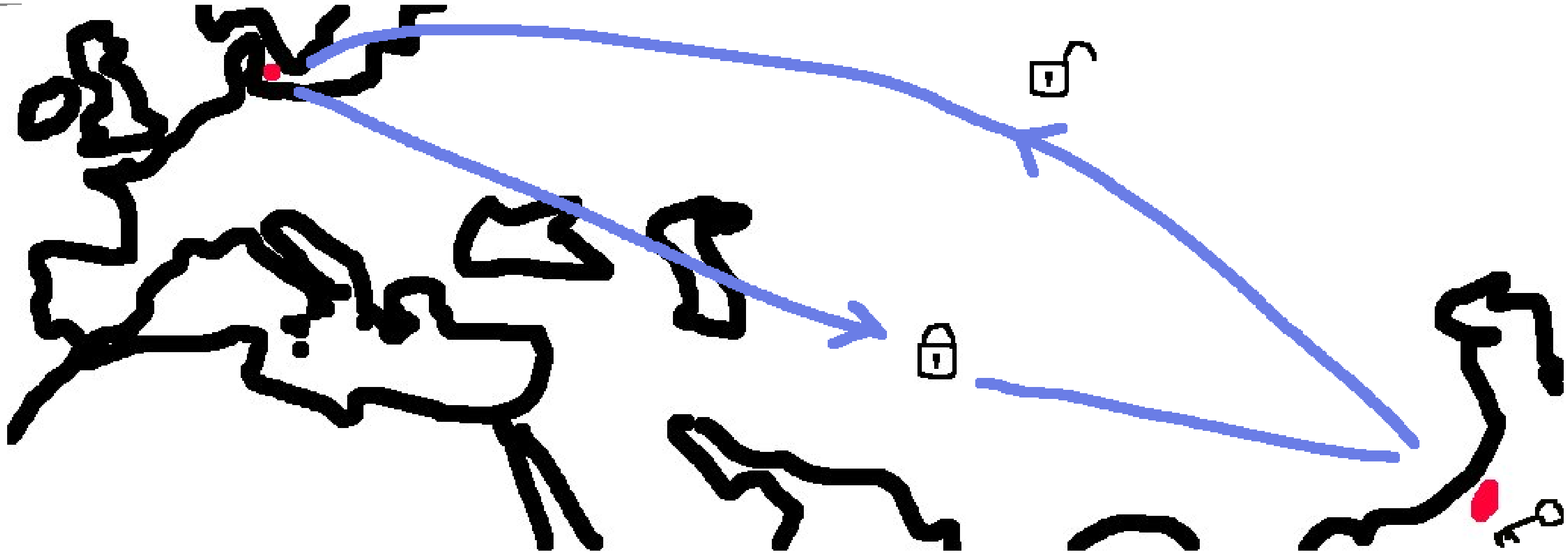
26.02.2010

The set up



- 2006 I get invited to a nice workshop in Taipei. How can I send Bo-Yin Yang some personal data?
- He could fly over to Denmark and get it in person or he could have a trusted courier fly over and deliver me a secret key (and then use secret-key cryptography from Dan's talk).
- Both have a high latency and cost.

Just mail an open padlock



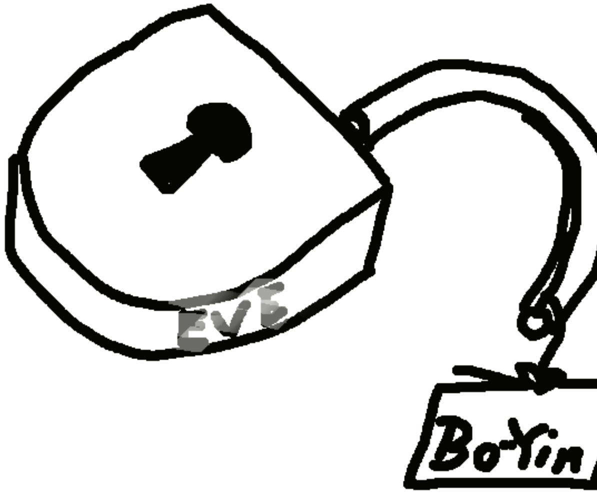
- Bo-Yin keeps key in Taiwan, mails open padlock.
- I can put message in a solid box, lock it with the lock and send that by regular mail.
- Bo-Yin can open box upon receipt.
- Don't need key to close padlock.
- I cannot read my own message once locked up.

Man in the middle attack

- This solves some problems but not all.
- Distribution: can have post offices with locks (so that I can initiate message and don't have to wait for padlock).
- Still some trust in mailman:

Man in the middle attack

- This solves some problems but not all.
- Distribution: can have post offices with locks (so that I can initiate message and don't have to wait for padlock).
- Still some trust in mailman:
  - Mailman cannot open Bo-Yin's lock.
  - But mailman can give me a substitute lock to which he has the key and claim that it's Bo-Yin's.
  - Picks up locked box, opens it with his key, reads my message, locks message in box with Bo-Yin's key.
- Need certificate that lock really belongs to Bo-Yin; trust in post office, web of trust.



How to simulate locks?

- Want to have mathematical replacement for physical lock and key.
- Hide my message so that only Bo-Yin can undo wrapping.
- Example: Bo-Yin publishes the following instructions:
  - Convert your message into a number m (binary representation, standard).
  - Pick a random number r and compute $R = 2^r$.
  - Compute $S = 1024^r \cdot m$ and send R and S to me.
- How can he get back m? Given r that's easy. It's also easy if he knows $b = \log_2 1024$; actually $b = 10$.
- Can compute $m = S/(R^b)$ because $R^b = 1024^r$.

How to simulate locks?

- Want to have mathematical replacement for physical lock and key.
- Hide my message so that only Bo-Yin can undo wrapping.
- Example: Bo-Yin publishes the following instructions:
  - Convert your message into a number m (binary representation, standard).
  - Pick a random number r and compute $R = 2^r$.
  - Compute $S = 1024^r \cdot m$ and send R and S to me.
- How can he get back m? Given r that's easy. It's also easy if he knows $b = \log_2 1024$; actually $b = 10$.
- Can compute $m = S/(R^b)$ because $R^b = 1024^r$.
- Problem: $\log_2$ easily computable; attacker sees $b = 10$.

How to simulate locks?

- Want to have mathematical replacement for physical lock and key.
- Hide my message so that only Bo-Yin can undo wrapping.
- Example: Bo-Yin publishes the following instructions:
  - Convert your message into a number m (binary representation, standard).
  - Pick a random number r and compute $R = 2^r$.
  - Compute $S = 1024^r \cdot m$ and send R and S to me.
- How can he get back m? Given r that's easy. It's also easy if he knows $b = \log_2 1024$; actually $b = 10$.
- Can compute $m = S/(R^b)$ because $R^b = 1024^r$.
- Solution: use modular reduction modulo a prime p.

ElGamal encryption

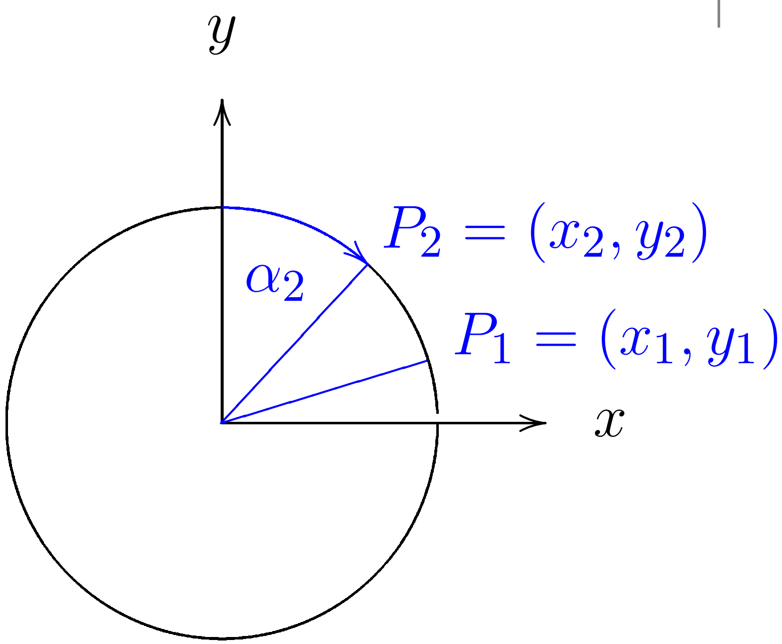
- Public parameters: big prime p (at least 2048 bits), integer $g < p$, order ℓ of g modulo p (should be a prime).
- Every user computes a public key $B = g^b$ using a random integer b. The public key is put in a public database; b is the secret key which stays with the user.
- If I want to send a message to this user I
  - look up his public key B,
  - convert my message into sequence of numbers m_i each smaller than p,
  - pick a random r_i for each m_i and put $R_i = g^{r_i} \bmod p$,
  - compute $S_i = B^{r_i} \cdot m_i \bmod p$ and send (R_i, S_i) to user. This ciphertext has length about $2 \log_2 p$.
- User computes $m_i = S_i/(R_i^b)$ (because $R_i^b = g^{r_i b} = B^{r_i}$).

Higher efficiency

- Hybrid encryption: For long messages this is very inefficient – modular computations take longer than symmetric cryptography, message length doubles. Use ElGamal to send AES key, then use that to encrypt message; this leads to much smaller overhead.
- Use elliptic curves: operate with much smaller numbers while achieving same level of security.
- Search the crypto heaven: Needs good mathematical understanding to estimate security; can break down description into simple instructions.
- Understand the deep-submicron hell of physics to protect your implementation against implementation dependent attacks, e.g. side-channel attacks.
- Heaven to hell interplay: some systems are easier to implement securely than others.

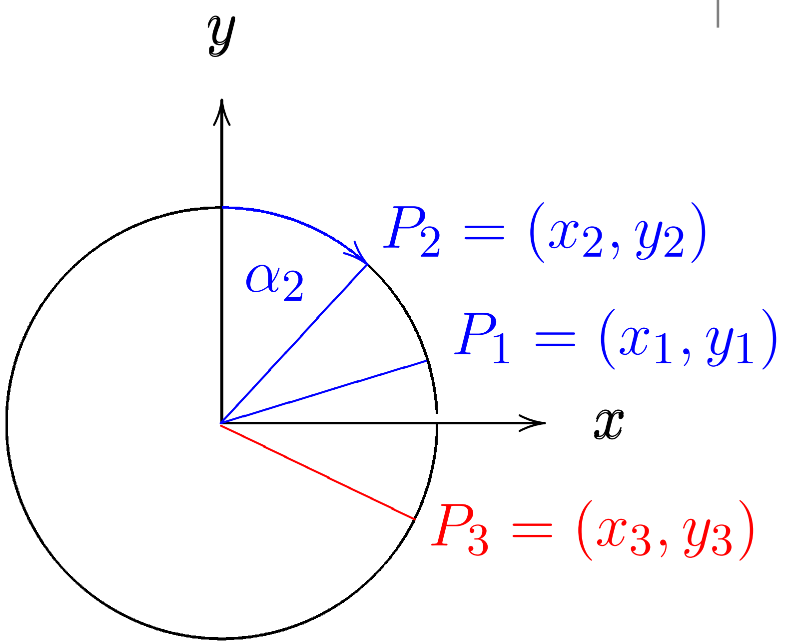
Do you know how to add on a circle?

Let K be a field with $2 \neq 0$.
Circle: $\{(x, y) \in K \times K | x^2 + y^2 = 1\}$



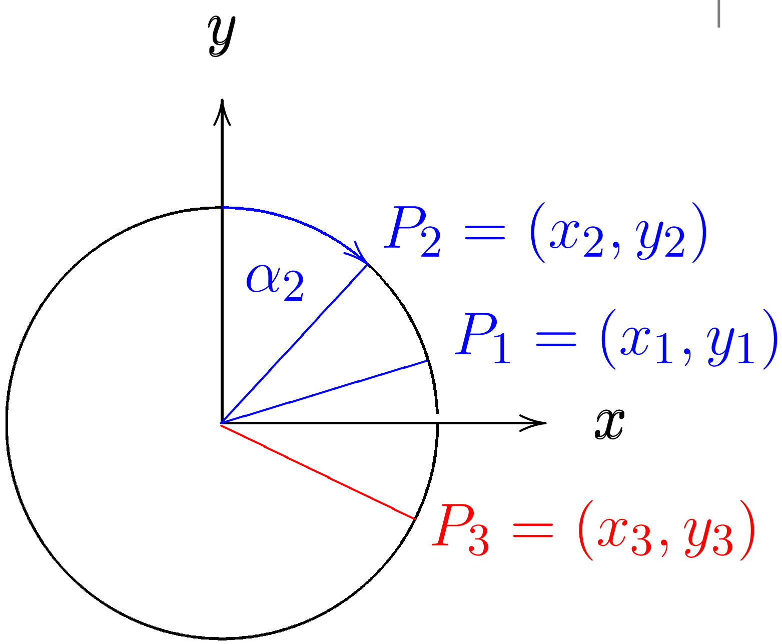
Do you know how to add on a circle?

Let K be a field with $2 \neq 0$.
Circle: $\{(x, y) \in K \times K | x^2 + y^2 = 1\}$
$x_i = \sin(\alpha_i),\ y_i = \cos(\alpha_i)$



Do you know how to add on a circle?

Let K be a field with $2 \neq 0$.
Circle: $\{(x,y) \in K \times K | x^2 + y^2 = 1\}$
$x_i = \sin(\alpha_i),\ y_i = \cos(\alpha_i)$



$$
\begin{aligned}
x_3 &= \sin(\alpha_1 + \alpha_2) \\
&= \sin(\alpha_1)\cos(\alpha_2) + \cos(\alpha_1)\sin(\alpha_2)
\end{aligned}
$$

$$
\begin{aligned}
y_3 &= \cos(\alpha_1 + \alpha_2) \\
&= \cos(\alpha_1)\cos(\alpha_2) - \sin(\alpha_1)\sin(\alpha_2)
\end{aligned}
$$

Addition of angles defines commutative group law
$(x_1, y_1) + (x_2, y_2) = (x_3, y_3)$, where

$$x_3 = x_1 y_2 + y_1 x_2 \text{ and } y_3 = y_1 y_2 - x_1 x_2.$$

Cryptography on the circle I

- $(0, 1)$ is at $\alpha = 0$. Then $(0, 1) + Q = Q + (0, 1) = Q$.
- $R = (0, -1)$ is at angle $180°$. Then $[2]R =$

Cryptography on the circle I

- $(0, 1)$ is at $\alpha = 0$. Then $(0, 1) + Q = Q + (0, 1) = Q$.
- $R = (0, -1)$ is at angle $180°$. Then $[2]R =(0, 1)$.
- What is the order of $(1, 0)$?

Cryptography on the circle I

- $(0, 1)$ is at $\alpha = 0$. Then $(0, 1) + Q = Q + (0, 1) = Q$.
- $R = (0, -1)$ is at angle $180°$. Then $[2]R =(0, 1)$.
- What is the order of $(1, 0)$? $[2](1, 0) = (0, -1)$.

Cryptography on the circle I

- $(0,1)$ is at $\alpha = 0$. Then $(0,1) + Q = Q + (0,1) = Q$.
- $R = (0,-1)$ is at angle $180°$. Then $[2]R =(0,1)$.
- What is the order of $(1,0)$? $[2](1,0) = (0,-1)$.
- Negative of (x,y) is $(-x,y)$.
- These observations are clear from the angles on the circle, e.g. $(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}})$ is at $45°$ and has order 8.
- How about $S = (\frac{3}{5}, \frac{4}{5})$?

Cryptography on the circle I

- $(0,1)$ is at $\alpha = 0$. Then $(0,1) + Q = Q + (0,1) = Q$.
- $R = (0,-1)$ is at angle $180°$. Then $[2]R =(0,1)$.
- What is the order of $(1,0)$? $[2](1,0) = (0,-1)$.
- Negative of (x,y) is $(-x,y)$.
- These observations are clear from the angles on the circle, e.g. $(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}})$ is at $45°$ and has order 8.
- How about $S = (\frac{3}{5}, \frac{4}{5})$?
  Compute $[2]S$: $x_3 = \frac{3}{5}\frac{4}{5} + \frac{4}{5}\frac{3}{5} = \frac{24}{25}, \;\; y_3 = \frac{4}{5}\frac{4}{5} - \frac{3}{5}\frac{3}{5} = \frac{7}{25}$

$[3]S = [2]S + S$: $x_3 = \frac{24}{25}\frac{4}{5} + \frac{7}{25}\frac{3}{5} = \frac{103}{125}, \;\; y_3 = \frac{7}{25}\frac{4}{5} - \frac{24}{25}\frac{3}{5} = \frac{-54}{125}$.

$[4]P = (\frac{536}{625}, \frac{-527}{625}) = (\frac{536}{5^4}, \frac{-527}{5^4})$.

ElGamal on the circle

- Public parameter: $P = (\frac{3}{5}, \frac{4}{5})$.
- Every user computes a public key $B = [b]P$ using a random integer b. The public key is put in a public database; b is the secret key which stays with the user.
- If I want to send a message to this user I
  - look up his public key B,
  - convert my message into a point M on the circle,
  - pick a random r and put $R = [r]P$,
  - compute $S = [r]B + M$ and send (R, S) to user.
- User computes $m = S - [b]R$ (because $[b]R = [rb]P = [r]B$).

Computation $[b]P$ is called scalar multiplication.

Cryptography on the circle II

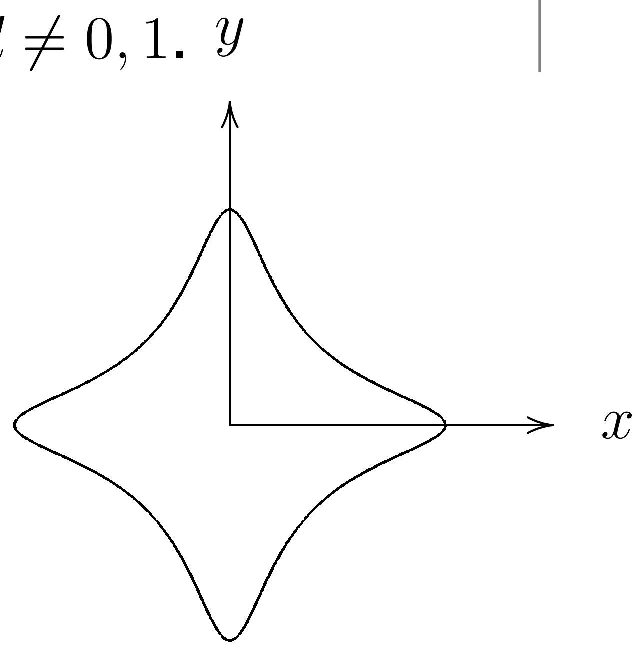
- First observation: can see which multiple was used in power of 5.
- Same problem as before – and same solution: use integers modulo a number.
- Implementation got a bit more complicated – need to do 4 modular multiplications for each step (can optimize this down to 3 multiplications).
- Security? Haven't gained much. For this set up we need to have p with at least 1024 bits (half the length as before).
- Large numbers come from index calculus attacks that work in finite fields.
- These are similar to attacks on RSA in complexity; so also similar key sizes.

Now add on an elliptic curve

Let K be a field with $2 \neq 0$. Let $d \in K$ with $d \neq 0, 1$.
Edwards curve (nice form of elliptic curve):

$$\{(x,y) \in K \times K | x^2 + y^2 = 1 + dx^2y^2\}$$

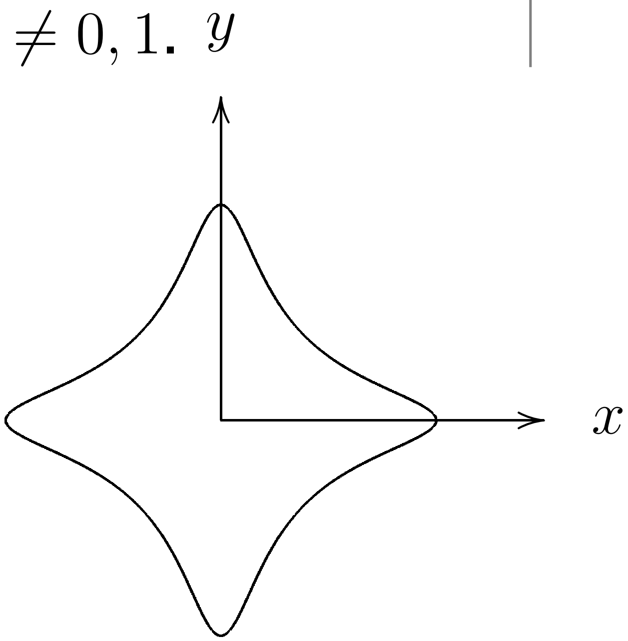
Harold M. Edwards,
(Bulletin of the AMS, **44**, 393–422, 2007)



Now add on an elliptic curve

Let K be a field with $2 \neq 0$. Let $d \in K$ with $d \neq 0, 1$.
Edwards curve (nice form of elliptic curve):

$$\{(x, y) \in K \times K | x^2 + y^2 = 1 + dx^2y^2\}$$



Harold M. Edwards,
(Bulletin of the AMS, **44**, 393–422, 2007)

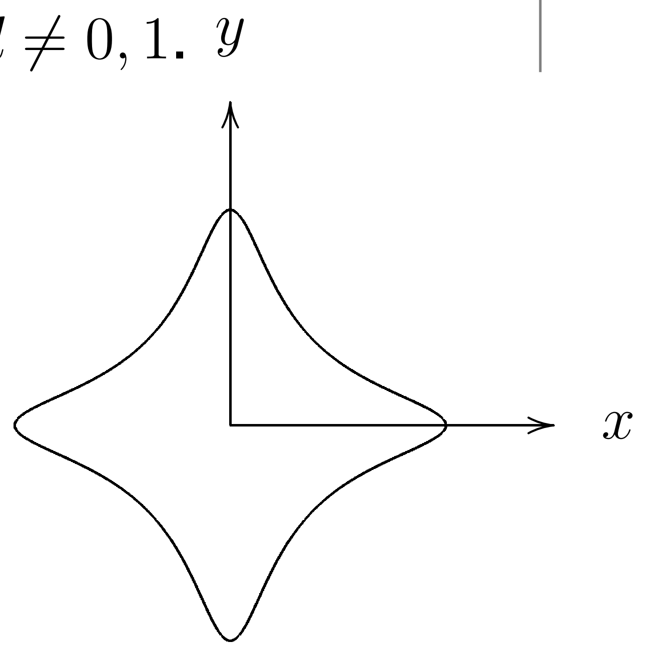
Associative operation on points
$(x_1, y_1) + (x_2, y_2) = (x_3, y_3)$
defined by Edwards addition law

$$x_3 = \frac{x_1y_2 + y_1x_2}{1 + dx_1x_2y_1y_2} \text{ and } y_3 = \frac{y_1y_2 - x_1x_2}{1 - dx_1x_2y_1y_2}.$$

Now add on an elliptic curve

Let K be a field with $2 \neq 0$. Let $d \in K$ with $d \neq 0, 1$.
Edwards curve (nice form of elliptic curve):

$$\{(x, y) \in K \times K | x^2 + y^2 = 1 + dx^2y^2\}$$



Harold M. Edwards,
(Bulletin of the AMS, **44**, 393–422, 2007)

Associative operation on points
$(x_1, y_1) + (x_2, y_2) = (x_3, y_3)$
defined by Edwards addition law

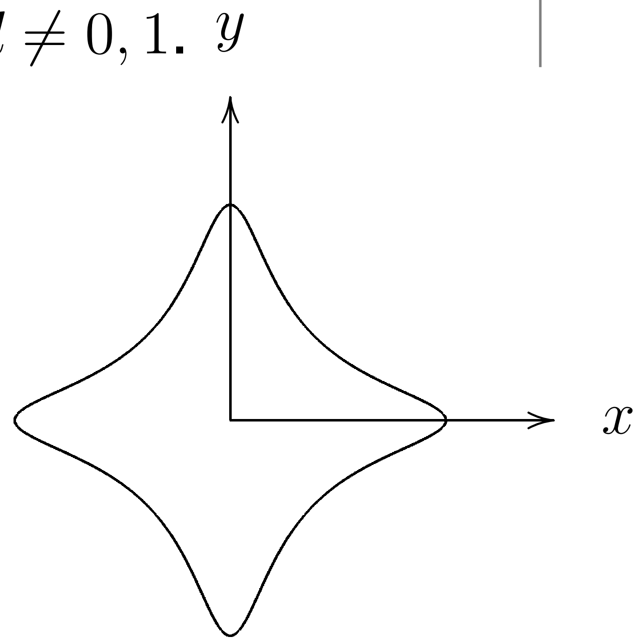
$$x_3 = \frac{x_1y_2 + y_1x_2}{1 + dx_1x_2y_1y_2} \text{ and } y_3 = \frac{y_1y_2 - x_1x_2}{1 - dx_1x_2y_1y_2}.$$

- Neutral element is

Now add on an elliptic curve

Let K be a field with $2 \neq 0$. Let $d \in K$ with $d \neq 0, 1$.
Edwards curve (nice form of elliptic curve):

$$\{(x, y) \in K \times K | x^2 + y^2 = 1 + dx^2y^2\}$$



Harold M. Edwards,
(Bulletin of the AMS, **44**, 393–422, 2007)

Associative operation on points
$(x_1, y_1) + (x_2, y_2) = (x_3, y_3)$
defined by Edwards addition law

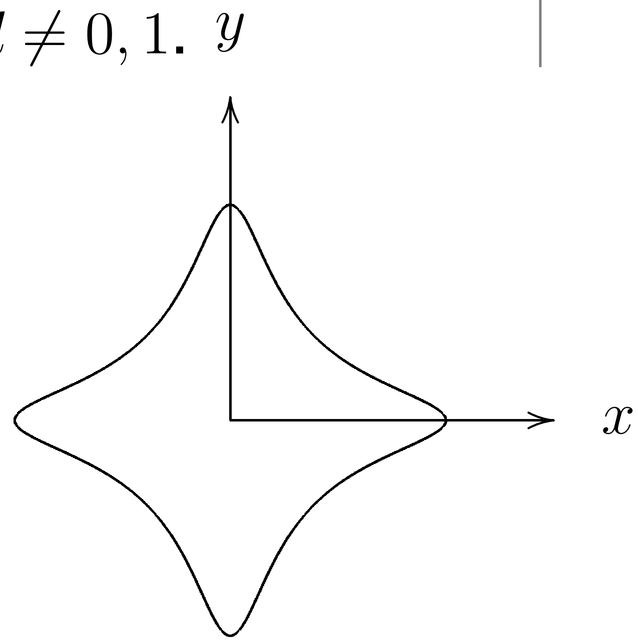
$$x_3 = \frac{x_1y_2 + y_1x_2}{1 + dx_1x_2y_1y_2} \text{ and } y_3 = \frac{y_1y_2 - x_1x_2}{1 - dx_1x_2y_1y_2}.$$

- Neutral element is $(0, 1)$ (like on circle).

Now add on an elliptic curve

Let K be a field with $2 \neq 0$. Let $d \in K$ with $d \neq 0, 1$.
Edwards curve (nice form of elliptic curve):

$$\{(x, y) \in K \times K | x^2 + y^2 = 1 + dx^2y^2\}$$



Harold M. Edwards,
(Bulletin of the AMS, **44**, 393–422, 2007)

Associative operation on points
$(x_1, y_1) + (x_2, y_2) = (x_3, y_3)$
defined by Edwards addition law

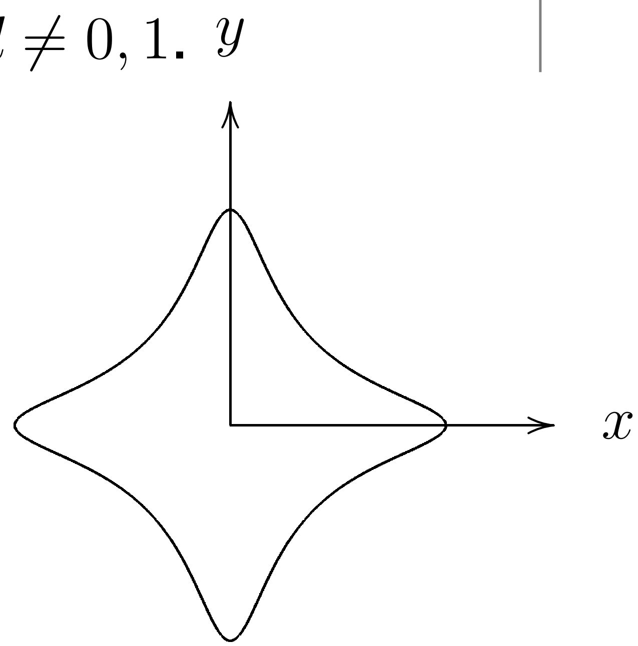
$$x_3 = \frac{x_1y_2 + y_1x_2}{1 + dx_1x_2y_1y_2} \text{ and } y_3 = \frac{y_1y_2 - x_1x_2}{1 - dx_1x_2y_1y_2}.$$

- Neutral element is $(0, 1)$ (like on circle).
- $-(x_1, y_1) =$

Now add on an elliptic curve

Let K be a field with $2 \neq 0$. Let $d \in K$ with $d \neq 0, 1$.
Edwards curve (nice form of elliptic curve):

$$\{(x, y) \in K \times K | x^2 + y^2 = 1 + dx^2y^2\}$$



Harold M. Edwards,
(Bulletin of the AMS, **44**, 393–422, 2007)

Associative operation on points
$(x_1, y_1) + (x_2, y_2) = (x_3, y_3)$
defined by Edwards addition law

$$x_3 = \frac{x_1y_2 + y_1x_2}{1 + dx_1x_2y_1y_2} \text{ and } y_3 = \frac{y_1y_2 - x_1x_2}{1 - dx_1x_2y_1y_2}.$$

- Neutral element is $(0, 1)$ (like on circle).
- $-(x_1, y_1) = (-x_1, y_1)$ (like on circle).

Explicit formulas: addition

- $(x_1, y_1) + (x_2, y_2) = \left(\frac{x_1y_2 + y_1x_2}{1 + dx_1x_2y_1y_2}, \frac{y_1y_2 - x_1x_2}{1 - dx_1x_2y_1y_2}\right)$.
- Avoid inversions: Use $(X_1 : Y_1 : Z_1)$ with $Z_1 \neq 0$ to represent $(x_1, y_1) = (X_1/Z_1, Y_1/Z_1)$, i. e., $(X_1 : Y_1 : Z_1) = (\lambda X_1 : \lambda Y_1 : \lambda Z_1)$ for $\lambda \neq 0$.
- Addition formulas in projective coordinates:

$$\begin{aligned}
A &= Z_1 \cdot Z_2;\ B = A^2;\ C = X_1 \cdot X_2;\ D = Y_1 \cdot Y_2;\\
E &= d \cdot C \cdot D;\ F = B - E;\ G = B + E;\\
X_3 &= A \cdot F \cdot ((X_1 + Y_1) \cdot (X_2 + Y_2) - C - D);\\
Y_3 &= A \cdot G \cdot (D - C);\\
Z_3 &= F \cdot G.
\end{aligned}$$

- Needs 10M + 1S + 1D + 7A, i.e. 10 multiplications, 1 squaring, 1 multiplication involving d, and 7 additions.

Faster dedicated doubling

- If $(x_2, y_2) = (x_1, y_1)$ we speak of doubling.
- Doubling formulas in projective coordinates:

$$\begin{aligned} B &= (X_1 + Y_1)^2;\ C = X_1^2;\ D = Y_1^2; \\ E &= C + D;\ H = Z_1^2;\ J = E - 2H; \\ X_3 &= (B - E) \cdot J;\ Y_3 = E \cdot (C - D);\ Z_3 = E \cdot J. \end{aligned}$$

- Needs 3M + 4S + 6A.
- Can compose any computation $[n]P$ from additions and doublings (use sliding window methods).
- Much higher security per bit – for security equivalent to the previous systems, q needs to have only 224 bits.
- Much slower increase in size of p. Using 3072 bits in the first system corresponds to 256 bits here.
- ECC (and HECC with $g = 2$) best in space consumption.

Fast addition law

- Very fast point addition 10M + 1S + 1D. Even faster with Inverted Edwards coordinates (9M+1S+1D) and Extended Edwards coordinates (8M+1S+1D).
- Dedicated doubling formulas need only 3M + 4S.
- Fastest scalar multiplication in the literature.
- For comparison: IEEE standard P1363 provides “the fastest arithmetic on elliptic curves” by using Jacobian coordinates on Weierstrass curves.
  - Point addition 12M + 4S.
  - Doubling formulas need only 4M + 4S.
- For more curve shapes, better algorithms (even for Weierstrass curves) and many more operations (mixed addition, re-addition, tripling, scaling,. . .) see `www.hyperelliptic.org/EFD`.

Beyond speed

- Neutral element of addition law is affine point.
- Addition law is symmetric in both inputs.
- $P + Q = \left(\dfrac{x_1 y_2 + y_1 x_2}{1 + d x_1 x_2 y_1 y_2}, \dfrac{y_1 y_2 - x_1 x_2}{1 - d x_1 x_2 y_1 y_2} \right).$

Beyond speed

- Neutral element of addition law is affine point.
- Addition law is symmetric in both inputs.
- $P+Q=\left(\dfrac{x_1y_2+y_1x_2}{1+dx_1x_2y_1y_2},\dfrac{y_1y_2-x_1x_2}{1-dx_1x_2y_1y_2}\right).$
- $[2]P=\left(\dfrac{x_1y_1+y_1x_1}{1+dx_1x_1y_1y_1},\dfrac{y_1y_1-x_1x_1}{1-dx_1x_1y_1y_1}\right).$
  Can compute this using the same formulas as general addition; different for Jacobian coordinates.

Beyond speed

- Neutral element of addition law is affine point.
- Addition law is symmetric in both inputs.
- $P + Q = \left(\frac{x_1 y_2 + y_1 x_2}{1 + d x_1 x_2 y_1 y_2}, \frac{y_1 y_2 - x_1 x_2}{1 - d x_1 x_2 y_1 y_2} \right).$
- $[2]P = \left(\frac{x_1 y_1 + y_1 x_1}{1 + d x_1 x_1 y_1 y_1}, \frac{y_1 y_1 - x_1 x_1}{1 - d x_1 x_1 y_1 y_1} \right).$
  Can compute this using the same formulas as general addition; different for Jacobian coordinates.
- If d is not a square in K the denominators $1 + d x_1 x_2 y_1 y_2$ and $1 - d x_1 x_2 y_1 y_2$ are never 0; addition law is complete.

Beyond speed

- Neutral element of addition law is affine point.
- Addition law is symmetric in both inputs.
- $P + Q = \left(\dfrac{x_1y_2 + y_1x_2}{1 + dx_1x_2y_1y_2}, \dfrac{y_1y_2 - x_1x_2}{1 - dx_1x_2y_1y_2} \right).$
- $[2]P = \left(\dfrac{x_1y_1 + y_1x_1}{1 + dx_1x_1y_1y_1}, \dfrac{y_1y_1 - x_1x_1}{1 - dx_1x_1y_1y_1} \right).$
  Can compute this using the same formulas as general addition; different for Jacobian coordinates.
- If d is not a square in K the denominators $1 + dx_1x_2y_1y_2$ and $1 - dx_1x_2y_1y_2$ are never 0; addition law is complete.
- Having addition law work for doubling removes some checks from the code; addition law also works for adding $P + (-P)$ or the neutral element.

Convenience and security

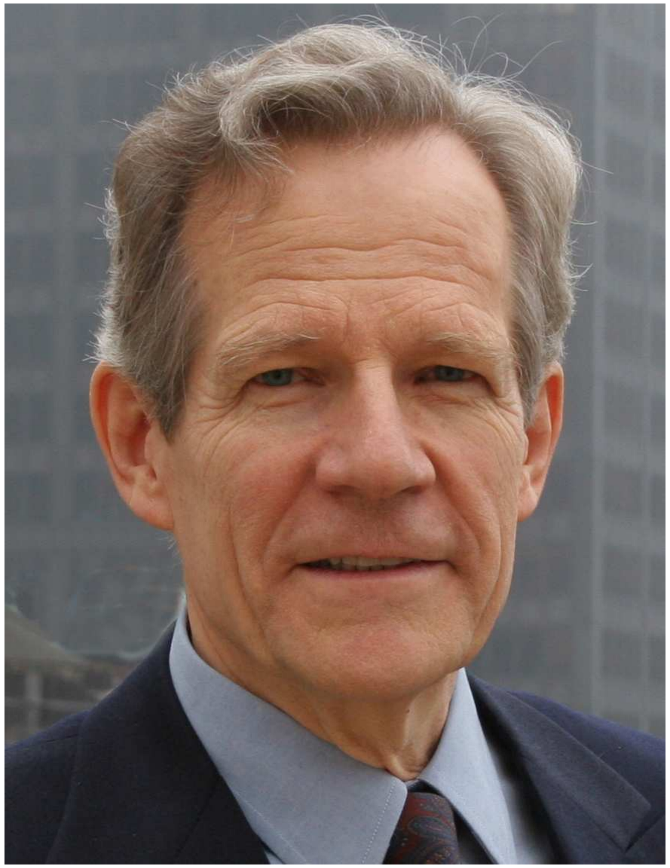
- Can reduce area by implementing only addition formula and using that for doublings. This is slower but ...
- ... has benefit that certain side-channel attacks are not feasible:
  - execution time of $[b]P$ does not depend on the chosen scalar (but does leak Hamming weight);
  - power traces of addition and doubling do not differ if addition formulas are used uniformly.
  - Still need to handle differential and template attacks.
- Attention: There are other formulas for Edwards curves that need distinguishing additions and doublings:

$$(x_3, y_3) = ((x_1y_1 + x_2y_2)/(x_1x_2 + y_1y_2), (x_1y_1 - x_2y_2)/(x_1y_2 - y_1x_2))$$

is addition on Edwards curve ... and fails for doublings.

Other representations

- Systems in which doublings can be computed via addition formulas called strongly unified.
- Besides Edwards curves also Hessian, twisted Hessian, and Jacobi quartics have this feature.
- Completeness, i.e. no exceptions to the addition law, more special requirement; depends on curve. Benefit: better protection against fault attacks.



- Older favorite system: Montgomery curves using Montgomery ladder for unified side-channel traces.

Requirements

- Computations modulo a prime q with ≥ 224 bits: Need additions, multiplications, squarings (if speed matters); maybe divisions.
- Arithmetic modulo p can be replaced by computations in a binary field $\mathbb{F}_{2^k}$ for some prime k.
- Base point needs 2 or 3 coordinates, each of the same size as q (or 2^k).
- Computations need space for 5 extra elements of this size; preferably have these in registers.
- Space for the scalar b – can be part of the code.
- Make sure to consider side-channel attacks as part of the design; depending on choice this might consume extra space. Physical shielding important but not enough.

Thank you for your attention!