Elliptic-curve cryptography

D. J. Bernstein
University of Illinois at Chicago

January 2010 news:

An academic team announces successful RSA-768 factorization.

Used ≈ 2 years of computation on ≈ 1000 CPU cores.

"Factoring a 1024-bit RSA modulus would be about a thousand times harder."

January 2010 news:

An academic team announces successful RSA-768 factorization.

Used ≈ 2 years of computation on ≈ 1000 CPU cores.

"Factoring a 1024-bit RSA modulus would be about a thousand times harder."

Many users of 1024-bit RSA: https://www.abnamro.nl, the root DNSSEC trial, etc.

2009 Kolkman et al.: "It is estimated that most zones can safely use 1024-bit keys for at least the next ten years."

1000 cores in perspective:

My laptop has 2 cores.

1000 cores in perspective:

My laptop has 2 cores.

A GTX 295 graphics card
has 60 cores ("MPs").

1000 cores in perspective:

My laptop has 2 cores.

A GTX 295 graphics card has 60 cores ("MPs").

EPFL's 200-Playstation cluster has 1200 cores.

1000 cores in perspective:

My laptop has 2 cores.

A GTX 295 graphics card
has 60 cores ("MPs").

EPFL's 200-Playstation
cluster has 1200 cores.

I have an account on the
TACC Ranger supercomputer,
which has 62976 cores.

1000 cores in perspective:

My laptop has 2 cores.

A GTX 295 graphics card has 60 cores ("MPs").

EPFL's 200-Playstation cluster has 1200 cores.

I have an account on the TACC Ranger supercomputer, which has 62976 cores.

The Conficker/Downadup criminal-controlled botnet has $\approx 10\,000\,000$ cores.

2003 Shamir et al.:
An attacker building ASICs
for $10 million can break
RSA-1024 in a year.

2003 RSA company:
Move to 2048 bits “over the
remainder of this decade.”

2003 Shamir et al.:
An attacker building ASICs
for $10 million can break
RSA-1024 in a year.

2003 RSA company:
Move to 2048 bits “over the
remainder of this decade.”

2007 NIST: Same.

2003 Shamir et al.:
An attacker building ASICs
for $10 million can break
RSA-1024 in a year.

2003 RSA company:
Move to 2048 bits “over the
remainder of this decade.”

2007 NIST: Same.

Another big reason to worry:
Attackers with more money
can use *batch* algorithms
that save time in breaking
many keys together.

A 1024-bit RSA key is built from two secret 512-bit primes.

There are $\approx 2^{503}$
possible 512-bit primes.
Can't imagine trying them all.
But the attacks are much faster:
only $\approx 2^{80}$ calculations.

A 1024-bit RSA key is built from two secret 512-bit primes.

There are $\approx 2^{503}$
possible 512-bit primes.
Can't imagine trying them all.
But the attacks are much faster:
only $\approx 2^{80}$ calculations.

2048-bit key: 1024-bit primes;
$\approx 2^{1014}$ possible primes.
Still below modern standards!
Attacks: $\approx 2^{112}$ calculations.

A 1024-bit RSA key is built from two secret 512-bit primes.

There are $\approx 2^{503}$
possible 512-bit primes.
Can't imagine trying them all.
But the attacks are much faster:
only $\approx 2^{80}$ calculations.

2048-bit key: 1024-bit primes;
$\approx 2^{1014}$ possible primes.
Still below modern standards!
Attacks: $\approx 2^{112}$ calculations.

3072-bit key: 1536-bit primes;
$\approx 2^{1526}$ possible primes.
Attacks: $\approx 2^{128}$ calculations.

These attacks use a simple idea:
"combining congruences."
Long history, including
many major improvements:
1975, CFRAC;
1977, linear sieve;
1982, quadratic sieve;
1990, number-field sieve.
Also many smaller improvements.

Costs of these algorithms for
breaking RSA-1024, RSA-2048:
$\approx 2^{120}$, 2^{170}, CFRAC;
$\approx 2^{110}$, 2^{160}, LS;
$\approx 2^{100}$, 2^{150}, QS;
$\approx 2^{80}$, 2^{112}, NFS.

1977: RSA is introduced.

1985: Miller proposes switching from RSA to elliptic curves. Explains several obstacles to congruence-combination attacks on elliptic curves.

Subsequent ECC history: Negligible security losses.

Subsequent RSA history: Continued security losses from improved algorithms for combining congruences. Major loss in 1990 (NFS); many smaller losses since then.

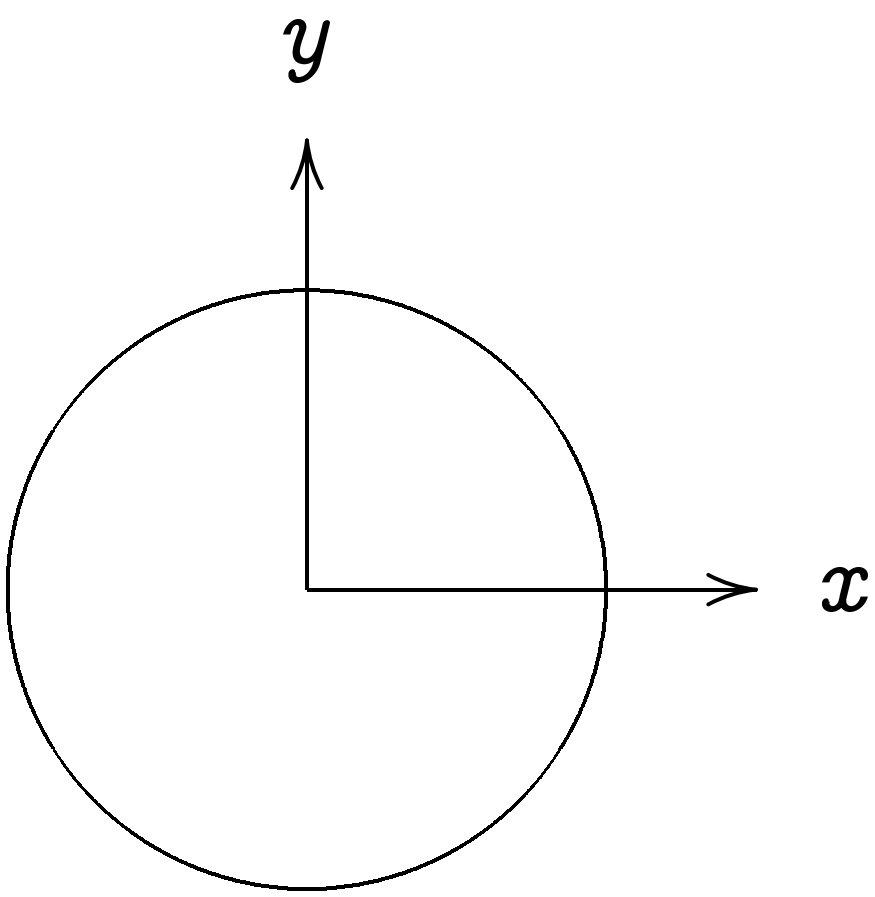
256-bit ECC keys match
security of 3072-bit RSA keys.

When properly implemented,
256-bit ECC is much faster
than 3072-bit RSA for
almost all real-world applications.

ANSI, IEEE, NIST issued
ECC standards ten years ago.
US government "Suite B"
now prohibits RSA, requires ECC.

For much more information see
the Handbook of Elliptic and
Hyperelliptic Curve Cryptography:
www.hyperelliptic.org/HEHCC

The clock



This is the curve $x^2 + y^2 = 1$.

Warning:

This is *not* an elliptic curve.

"Elliptic curve" $\neq$ "ellipse."

Examples of points on this curve:

Examples of points on this curve:
$(0, 1) =$ “12:00”.

Examples of points on this curve:

$(0, 1) =$ “12:00”.

$(0, -1) =$ “6:00”.

Examples of points on this curve:

$(0, 1) =$ “12:00”.

$(0, -1) =$ “6:00”.

$(1, 0) =$ “3:00”.

Examples of points on this curve:

$(0, 1) =$ “12:00”.

$(0, -1) =$ “6:00”.

$(1, 0) =$ “3:00”.

$(-1, 0) =$ “9:00”.

Examples of points on this curve:

$(0, 1) =$ “12:00”.

$(0, -1) =$ “6:00”.

$(1, 0) =$ “3:00”.

$(-1, 0) =$ “9:00”.

$(\sqrt{3/4}, 1/2) =$

Examples of points on this curve:

$(0, 1) =$ “12:00”.

$(0, -1) =$ “6:00”.

$(1, 0) =$ “3:00”.

$(-1, 0) =$ “9:00”.

$(\sqrt{3/4}, 1/2) =$ “2:00”.

Examples of points on this curve:

$(0, 1) =$ “12:00”.

$(0, -1) =$ “6:00”.

$(1, 0) =$ “3:00”.

$(-1, 0) =$ “9:00”.

$(\sqrt{3/4}, 1/2) =$ “2:00”.

$(1/2, -\sqrt{3/4}) =$

Examples of points on this curve:

$(0, 1) =$ “12:00”.

$(0, -1) =$ “6:00”.

$(1, 0) =$ “3:00”.

$(-1, 0) =$ “9:00”.

$(\sqrt{3/4}, 1/2) =$ “2:00”.

$(1/2, -\sqrt{3/4}) =$ “5:00”.

$(-1/2, -\sqrt{3/4}) =$

Examples of points on this curve:

$(0, 1) =$ “12:00”.

$(0, -1) =$ “6:00”.

$(1, 0) =$ “3:00”.

$(-1, 0) =$ “9:00”.

$(\sqrt{3/4}, 1/2) =$ “2:00”.

$(1/2, -\sqrt{3/4}) =$ “5:00”.

$(-1/2, -\sqrt{3/4}) =$ “7:00”.

Examples of points on this curve:

$(0, 1) =$ “12:00”.

$(0, -1) =$ “6:00”.

$(1, 0) =$ “3:00”.

$(-1, 0) =$ “9:00”.

$(\sqrt{3/4}, 1/2) =$ “2:00”.

$(1/2, -\sqrt{3/4}) =$ “5:00”.

$(-1/2, -\sqrt{3/4}) =$ “7:00”.

$(\sqrt{1/2}, \sqrt{1/2}) =$ “1:30”.

$(3/5, 4/5)$. $(-3/5, 4/5)$.

Examples of points on this curve:

$(0, 1) =$ "12:00".

$(0, -1) =$ "6:00".

$(1, 0) =$ "3:00".

$(-1, 0) =$ "9:00".

$(\sqrt{3/4}, 1/2) =$ "2:00".

$(1/2, -\sqrt{3/4}) =$ "5:00".

$(-1/2, -\sqrt{3/4}) =$ "7:00".

$(\sqrt{1/2}, \sqrt{1/2}) =$ "1:30".

$(3/5, 4/5)$. $(-3/5, 4/5)$.

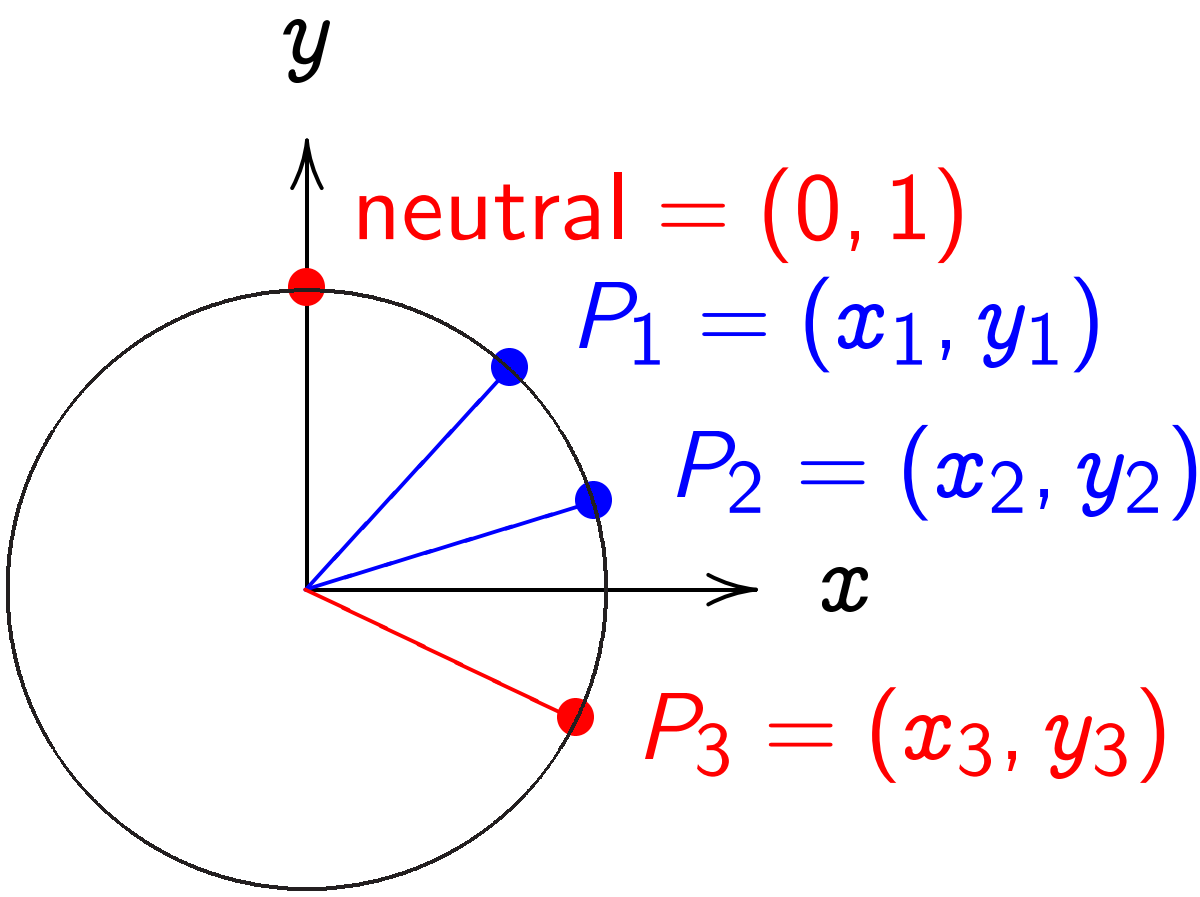
$(3/5, -4/5)$. $(-3/5, -4/5)$.

$(4/5, 3/5)$. $(-4/5, 3/5)$.

$(4/5, -3/5)$. $(-4/5, -3/5)$.

Many more.

Clock addition



Standard addition formula for the clock $x^2 + y^2 = 1$: sum of (x_1, y_1) and (x_2, y_2) is $(x_1y_2 + y_1x_2, y_1y_2 - x_1x_2)$.

Examples of clock addition:

“2:00” + “5:00”

$= (\sqrt{3/4}, 1/2) + (1/2, -\sqrt{3/4})$

Examples of clock addition:

$$\text{“2:00”} + \text{“5:00”}$$

$$= (\sqrt{3/4}, 1/2) + (1/2, -\sqrt{3/4})$$

$$= (-1/2, -\sqrt{3/4}) =$$

Examples of clock addition:

$$\text{“2:00”} + \text{“5:00”}$$
$$= (\sqrt{3/4}, 1/2) + (1/2, -\sqrt{3/4})$$
$$= (-1/2, -\sqrt{3/4}) = \text{“7:00”}.$$

Examples of clock addition:

$$\text{“2:00”} + \text{“5:00”}$$
$$= (\sqrt{3/4}, 1/2) + (1/2, -\sqrt{3/4})$$
$$= (-1/2, -\sqrt{3/4}) = \text{“7:00”}.$$

$$\text{“5:00”} + \text{“9:00”}$$
$$= (1/2, -\sqrt{3/4}) + (-1, 0)$$

Examples of clock addition:

$$\text{“2:00”} + \text{“5:00”}$$
$$= (\sqrt{3/4}, 1/2) + (1/2, -\sqrt{3/4})$$
$$= (-1/2, -\sqrt{3/4}) = \text{“7:00”}.$$

$$\text{“5:00”} + \text{“9:00”}$$
$$= (1/2, -\sqrt{3/4}) + (-1, 0)$$
$$= (\sqrt{3/4}, 1/2) = \text{“2:00”}.$$

Examples of clock addition:

$$\text{“2:00”} + \text{“5:00”}$$
$$= (\sqrt{3/4}, 1/2) + (1/2, -\sqrt{3/4})$$
$$= (-1/2, -\sqrt{3/4}) = \text{“7:00”}.$$

$$\text{“5:00”} + \text{“9:00”}$$
$$= (1/2, -\sqrt{3/4}) + (-1, 0)$$
$$= (\sqrt{3/4}, 1/2) = \text{“2:00”}.$$

$$2\left(\frac{3}{5}, \frac{4}{5}\right) =$$

Examples of clock addition:

$$\text{“2:00”} + \text{“5:00”}$$
$$= (\sqrt{3/4}, 1/2) + (1/2, -\sqrt{3/4})$$
$$= (-1/2, -\sqrt{3/4}) = \text{“7:00”}.$$

$$\text{“5:00”} + \text{“9:00”}$$
$$= (1/2, -\sqrt{3/4}) + (-1, 0)$$
$$= (\sqrt{3/4}, 1/2) = \text{“2:00”}.$$

$$2\left(\frac{3}{5}, \frac{4}{5}\right) = \left(\frac{24}{25}, \frac{7}{25}\right).$$

Examples of clock addition:

$$\text{“2:00”} + \text{“5:00”}$$
$$= (\sqrt{3/4}, 1/2) + (1/2, -\sqrt{3/4})$$
$$= (-1/2, -\sqrt{3/4}) = \text{“7:00”}.$$

$$\text{“5:00”} + \text{“9:00”}$$
$$= (1/2, -\sqrt{3/4}) + (-1, 0)$$
$$= (\sqrt{3/4}, 1/2) = \text{“2:00”}.$$

$$2\left(\frac{3}{5}, \frac{4}{5}\right) = \left(\frac{24}{25}, \frac{7}{25}\right).$$

$$3\left(\frac{3}{5}, \frac{4}{5}\right) = \left(\frac{117}{125}, \frac{-44}{125}\right).$$

Examples of clock addition:

$$\text{“2:00”} + \text{“5:00”}$$
$$= (\sqrt{3/4}, 1/2) + (1/2, -\sqrt{3/4})$$
$$= (-1/2, -\sqrt{3/4}) = \text{“7:00”}.$$

$$\text{“5:00”} + \text{“9:00”}$$
$$= (1/2, -\sqrt{3/4}) + (-1, 0)$$
$$= (\sqrt{3/4}, 1/2) = \text{“2:00”}.$$

$$2\left(\frac{3}{5}, \frac{4}{5}\right) = \left(\frac{24}{25}, \frac{7}{25}\right).$$

$$3\left(\frac{3}{5}, \frac{4}{5}\right) = \left(\frac{117}{125}, \frac{-44}{125}\right).$$

$$4\left(\frac{3}{5}, \frac{4}{5}\right) = \left(\frac{336}{625}, \frac{-527}{625}\right).$$

Examples of clock addition:

$$\text{“2:00”} + \text{“5:00”}$$
$$= (\sqrt{3/4}, 1/2) + (1/2, -\sqrt{3/4})$$
$$= (-1/2, -\sqrt{3/4}) = \text{“7:00”}.$$

$$\text{“5:00”} + \text{“9:00”}$$
$$= (1/2, -\sqrt{3/4}) + (-1, 0)$$
$$= (\sqrt{3/4}, 1/2) = \text{“2:00”}.$$

$$2\left(\frac{3}{5}, \frac{4}{5}\right) = \left(\frac{24}{25}, \frac{7}{25}\right).$$

$$3\left(\frac{3}{5}, \frac{4}{5}\right) = \left(\frac{117}{125}, \frac{-44}{125}\right).$$

$$4\left(\frac{3}{5}, \frac{4}{5}\right) = \left(\frac{336}{625}, \frac{-527}{625}\right).$$

$$(x_1, y_1) + (0, 1) =$$

Examples of clock addition:

$$\text{“2:00”} + \text{“5:00”}$$
$$= (\sqrt{3/4}, 1/2) + (1/2, -\sqrt{3/4})$$
$$= (-1/2, -\sqrt{3/4}) = \text{“7:00”}.$$

$$\text{“5:00”} + \text{“9:00”}$$
$$= (1/2, -\sqrt{3/4}) + (-1, 0)$$
$$= (\sqrt{3/4}, 1/2) = \text{“2:00”}.$$

$$2\left(\frac{3}{5}, \frac{4}{5}\right) = \left(\frac{24}{25}, \frac{7}{25}\right).$$

$$3\left(\frac{3}{5}, \frac{4}{5}\right) = \left(\frac{117}{125}, \frac{-44}{125}\right).$$

$$4\left(\frac{3}{5}, \frac{4}{5}\right) = \left(\frac{336}{625}, \frac{-527}{625}\right).$$

$$(x_1, y_1) + (0, 1) = (x_1, y_1).$$

$$(x_1, y_1) + (-x_1, y_1) =$$

Examples of clock addition:

“2:00” + “5:00”

$= (\sqrt{3/4}, 1/2) + (1/2, -\sqrt{3/4})$

$= (-1/2, -\sqrt{3/4}) =$ “7:00”.

“5:00” + “9:00”

$= (1/2, -\sqrt{3/4}) + (-1, 0)$

$= (\sqrt{3/4}, 1/2) =$ “2:00”.

$$2\left(\frac{3}{5}, \frac{4}{5}\right) = \left(\frac{24}{25}, \frac{7}{25}\right).$$

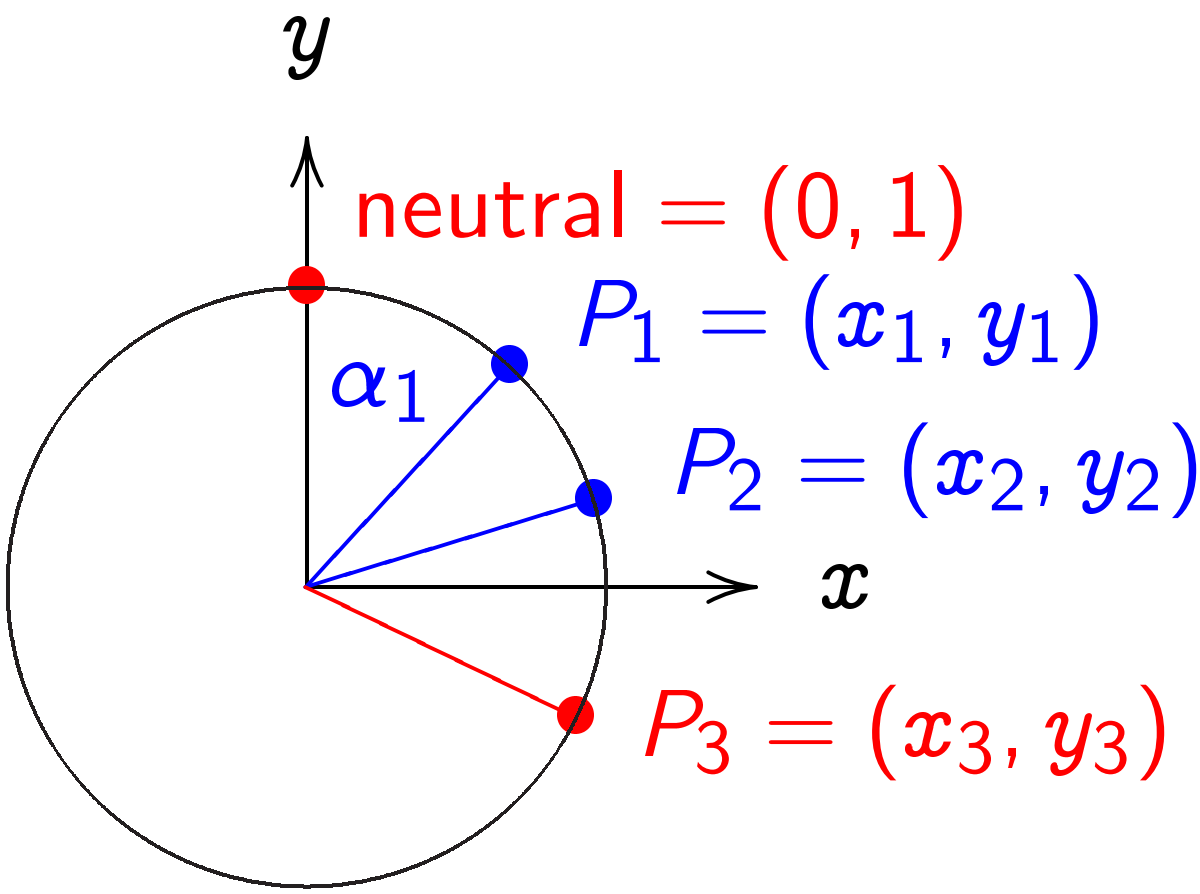
$$3\left(\frac{3}{5}, \frac{4}{5}\right) = \left(\frac{117}{125}, \frac{-44}{125}\right).$$

$$4\left(\frac{3}{5}, \frac{4}{5}\right) = \left(\frac{336}{625}, \frac{-527}{625}\right).$$

$$(x_1, y_1) + (0, 1) = (x_1, y_1).$$

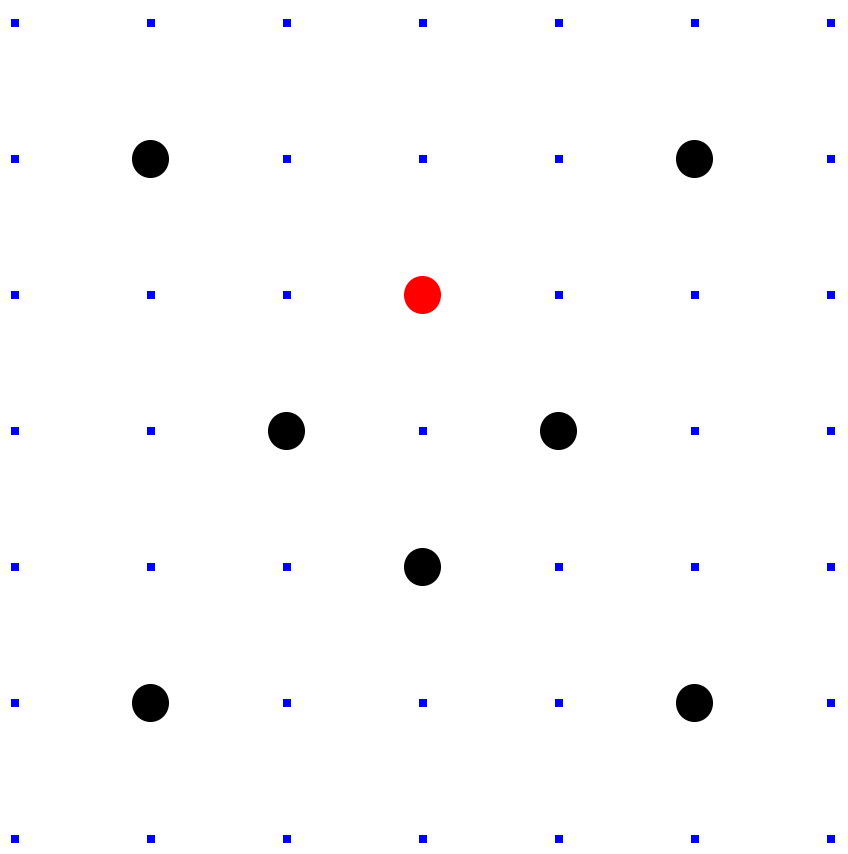
$$(x_1, y_1) + (-x_1, y_1) = (0, 1).$$

One way to remember
the clock addition law:



$x^2 + y^2 = 1$, parametrized by
$x = \sin\alpha, \quad y = \cos\alpha.$ Recall
$(\sin(\alpha_1 + \alpha_2), \cos(\alpha_1 + \alpha_2)) =$
$(\sin\alpha_1 \cos\alpha_2 + \cos\alpha_1 \sin\alpha_2,$
$\cos\alpha_1 \cos\alpha_2 - \sin\alpha_1 \sin\alpha_2).$

Clocks over finite fields



$\mathrm{Clock}(\mathbf{F}_7) =$
$\{(x, y) \in \mathbf{F}_7 \times \mathbf{F}_7 : x^2 + y^2 = 1\}.$
Here $\mathbf{F}_7 = \{0, 1, 2, 3, 4, 5, 6\}$
$= \{0, 1, 2, 3, -3, -2, -1\}$
with $+, -, \times$ modulo 7.

Larger example: Clock($\mathbf{F}_{1000003}$).

Examples of clock addition:

$2(1000, 2) = (4000, 7)$.

$4(1000, 2) = (56000, 97)$.

$8(1000, 2) = (863970, 18817)$.

$16(1000, 2) = (549438, 156853)$.

$17(1000, 2) = (951405, 877356)$.

With 30 additions I computed

$n(1000, 2) = (947472, 736284)$

for some 6-digit n.

Can you figure out n?

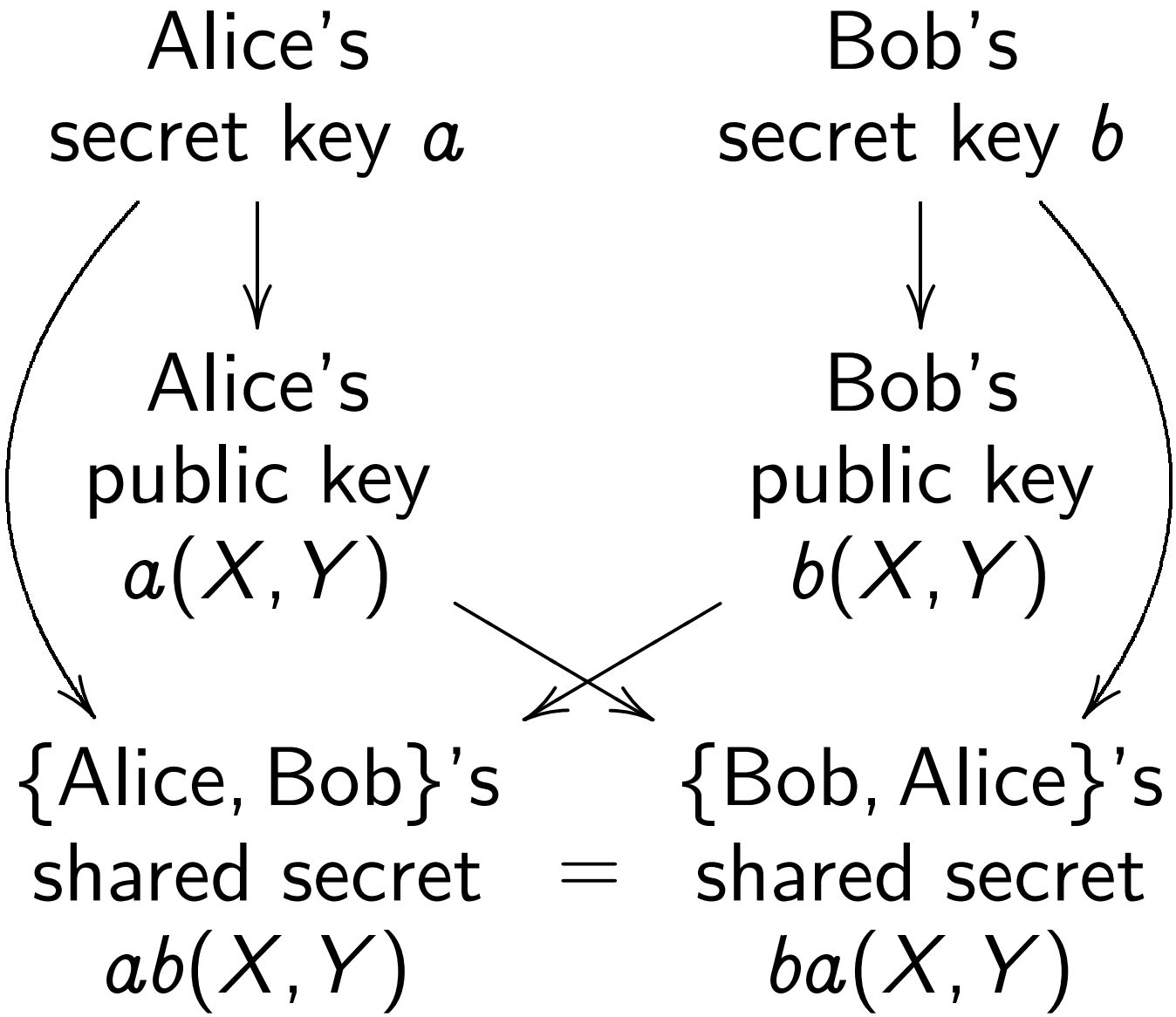
Clock cryptography

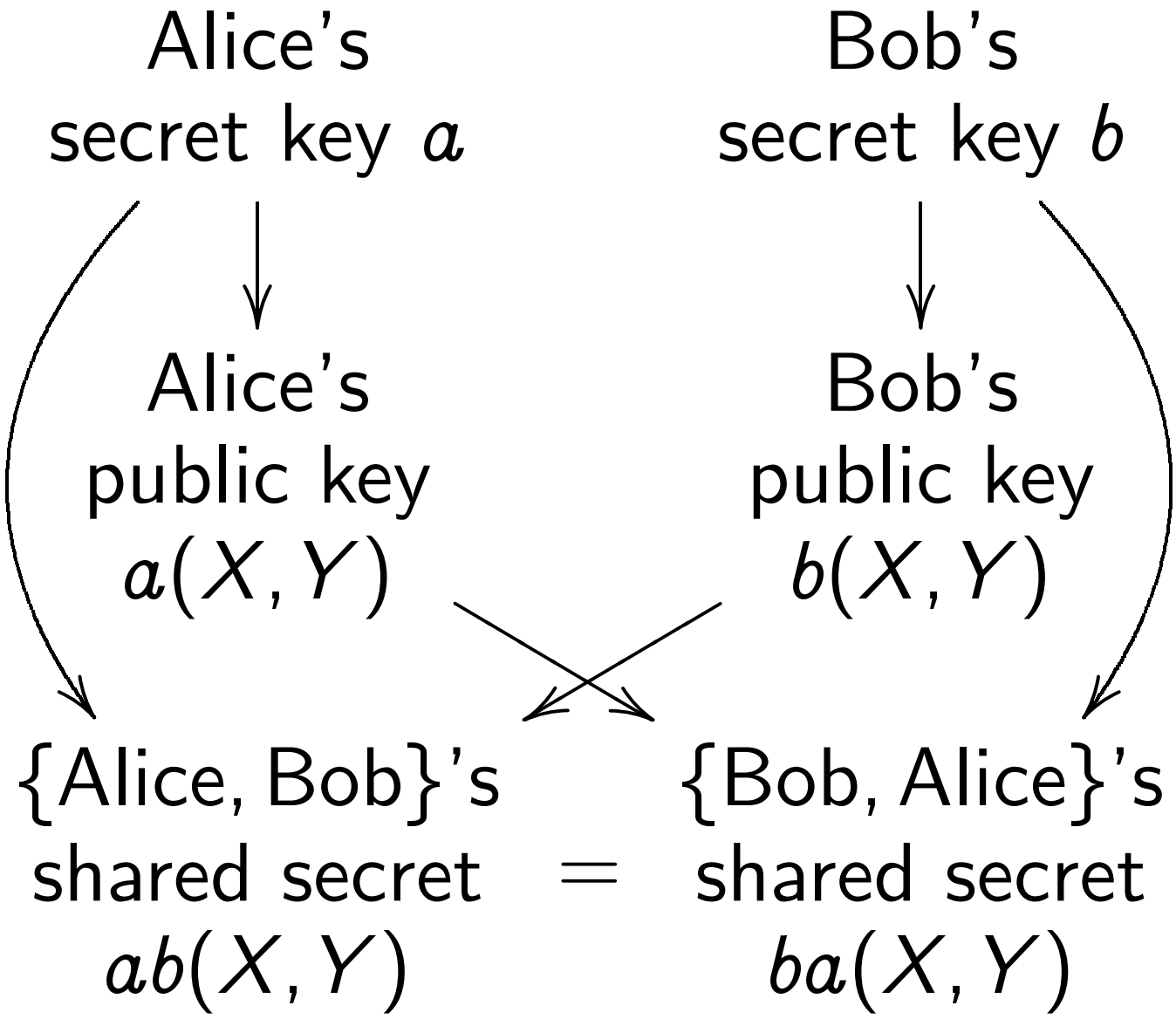
Standardize a large prime p
and some $(X, Y) \in \mathrm{Clock}(\mathbf{F}_p)$.
Follow standard security criteria.

Alice chooses big secret a.
Computes her public key $a(X, Y)$.

Bob chooses big secret b.
Computes his public key $b(X, Y)$.

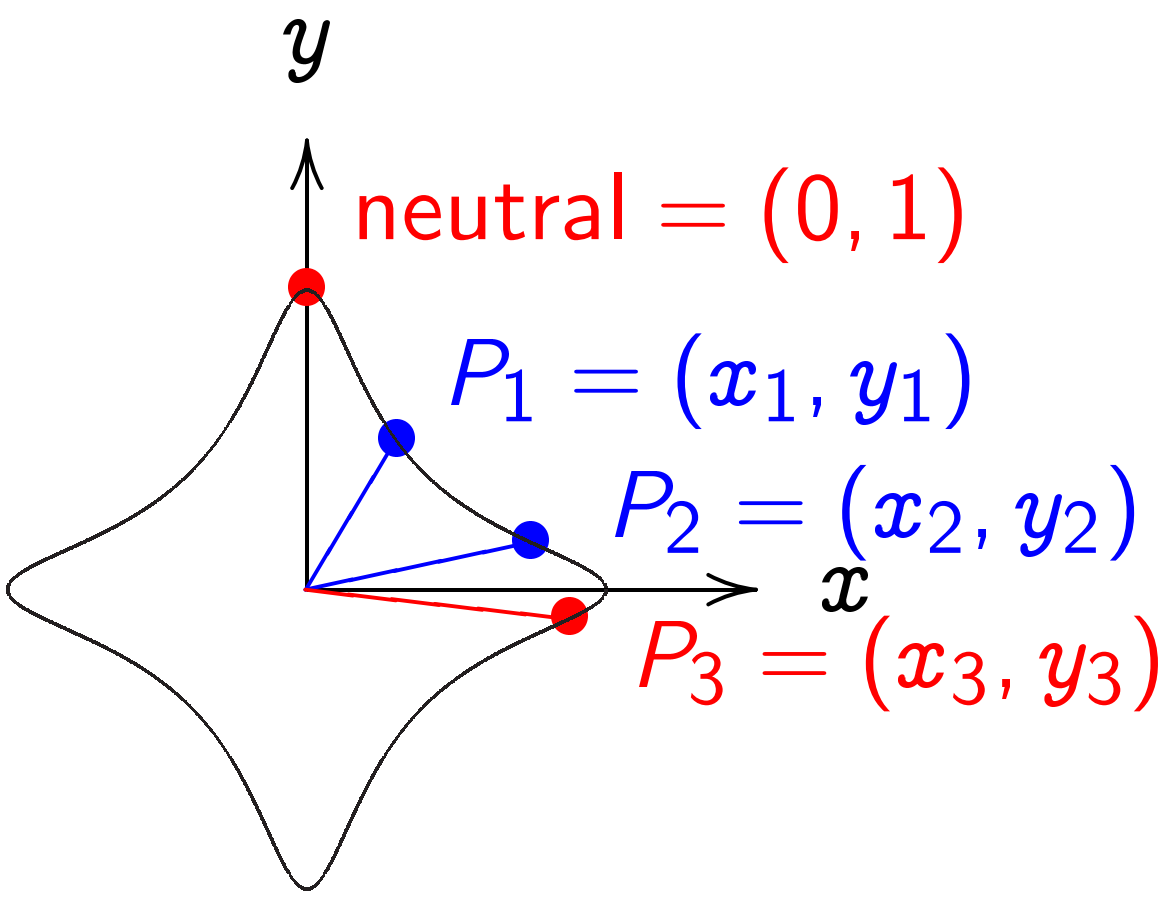
Alice computes $a(b(X, Y))$.
Bob computes $b(a(X, Y))$.
They use this shared secret
to encrypt with AES-GCM etc.


Alice's
secret key a
Bob's
secret key b
Alice's
public key
$a(X, Y)$
Bob's
public key
$b(X, Y)$
{Alice, Bob}'s
shared secret
$ab(X, Y)$
=
{Bob, Alice}'s
shared secret
$ba(X, Y)$



Warning: Clocks aren't elliptic!
Can attack clock cryptography
by combining congruences.
To match RSA-3072 security
need $p \approx 2^{1536}$.

Addition on an elliptic curve



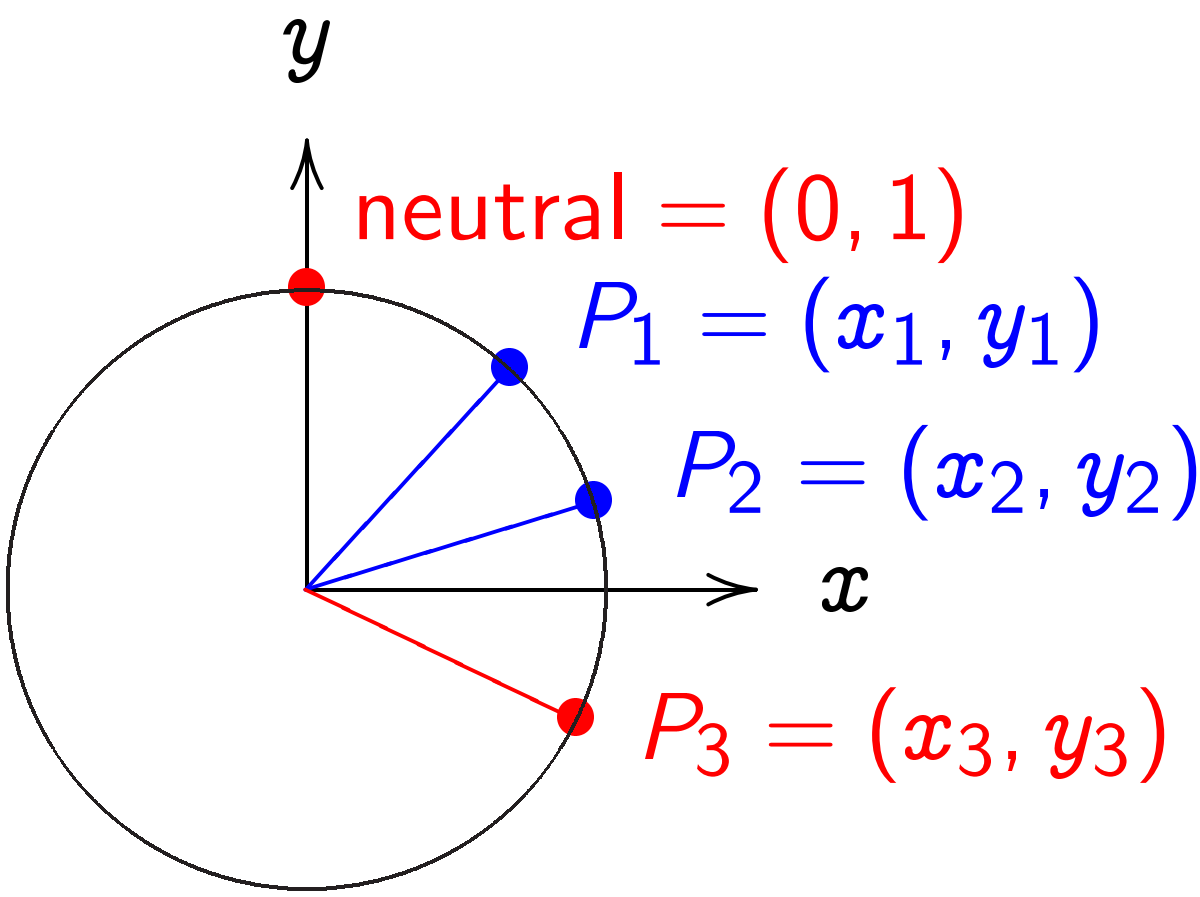
$x^2 + y^2 = 1 - 30x^2y^2$.

Sum of (x_1, y_1) and (x_2, y_2) is

$((x_1y_2+y_1x_2)/(1-30x_1x_2y_1y_2)$,

$(y_1y_2-x_1x_2)/(1+30x_1x_2y_1y_2))$.

The clock again, for comparison:



$x^2 + y^2 = 1.$

Sum of (x_1, y_1) and (x_2, y_2) is

$(x_1 y_2 + y_1 x_2,$

$y_1 y_2 - x_1 x_2).$

"Hey, there were divisions
in the elliptic addition law!
What if the denominators are 0?"

Answer: They aren't!

Can replace -30 by
anything that isn't a square.
The denominators will never be 0.

"Hey, there were divisions
in the elliptic addition law!
What if the denominators are 0?"

Answer: They aren't!

Can replace -30 by
anything that isn't a square.
The denominators will never be 0.

A typical high-security
elliptic curve, "Curve25519":
replace $\mathbf{R}$ by $\mathbf{F}_{2^{255}-19}$;
replace -30 by $1 - \frac{1}{121666}$.

Using ECC sensibly

Typical starting point:
Client knows secret key a
and server's public key $b(X, Y)$.
Client computes (and caches)
shared secret $ab(X, Y)$.

Client has packet for server.
Generates unique nonce.
Uses shared secret to encrypt
and authenticate packet.

Total packet overhead:
24 bytes for nonce,
16 bytes for authenticator,
32 bytes for client's public key.

Server receives packet,
sees client's public key $a(X, Y)$.
Server computes (and caches)
shared secret $ab(X, Y)$.

Server uses shared secret
to verify authenticator
and decrypt packet.

Client and server encrypt,
authenticate, verify, and decrypt
all subsequent packets
in the same way,
using the same shared secret.

Easy-to-use packet protection:
`crypto_box` from
`nacl.cace-project.eu`.

High-security curve (Curve25519).
High-security implementation
(e.g., no secret array indices).
Extensive code validation.

Very high speed:
Server can compute shared secrets
for 1000000 new clients
in 40 seconds of computation
on a Core 2 Quad.