Asymmetric Cryptography: A Deep Dive

Eli Holderness — @eli.holderness.dev — they/them/theirs

Eli (pronounced /ˈiːlaɪ̯/) is a is a research software advocate, recovering mathematician, and audience participator.

They like people, the web, and learning weird facts about computers.

@eli.holderness.dev

Agenda

Agenda

1. Brief history

Agenda

1. Brief history
2. How RSA works

Agenda

1. Brief history
2. How RSA works
3. How ECC works

Agenda

1. Brief history
2. How RSA works
3. How ECC works
4. QC & Shor's Algorithms

Agenda

1. Brief history
2. How RSA works
3. How ECC works
4. QC & Shor's Algorithms
5. What next?

1

A brief history of cryptography

NDC is great!

A	B	C	D	E	F	G	H	I	J	K
G	H	I	J	K	L	M	N	O	P	Q

TJI oy mxkgz!

NDC is great!

	1	2	3	4	5	6	7	8	9	10	11
+6	7	8	9	10	11	12	13	14	15	16	17

TJI oy mxkgz!

NDCISGREAT 14 4 3 9 19 7 18 5 1 20

NDCISGREAT

+

PRIVATEKEY

14 4 3 9 19 7 18 5 1 20

NDCISGREAT	14 4 3 9 19 7 18 5 1 20
+	+
PRIVATEKEY	16 18 9 22 1 20 5 11 5 25

NDCISGREAT	14 4 3 9 19 7 18 5 1 20
+	+
PRIVATEKEY	16 18 9 22 1 20 5 11 5 25
	=
	30 22 12 31 20 27 23 16 6 45

@eli.holderness.dev

NDCISGREAT	14 4 3 9 19 7 18 5 1 20
+	+
PRIVATEKEY	16 18 9 22 1 20 5 11 5 25
	=
	4 22 12 5 20 1 23 16 6 19

NDCISGREAT	14 4 3 9 19 7 18 5 1 20
+	+
PRIVATEKEY	16 18 9 22 1 20 5 11 5 25
=	=
DVLETAWPFS	4 22 12 5 20 1 23 16 6 19

symmetric cryptography
requires both parties to know
a specific secret

asymmetric cryptography relies on mathematical solutions that are very expensive to compute

2

RSA & group theory

RSA cryptosystem

RSA cryptosystem

published ‘officially’ in 1977 by Rivest, Shamir and Adleman

RSA cryptosystem

published 'officially' in 1977 by Rivest, Shamir and Adleman

also developed independently in 1973 by Clifford Cocks at GCHQ

RSA cryptosystem

published ‘officially’ in 1977 by Rivest, Shamir and Adleman

also developed independently in 1973 by Clifford Cocks at GCHQ

security based on the difficulty of factoring large numbers N = pq where p, q prime

worked example with $N = 323 = 17 * 19$

worked example with $N = 323 = 17 * 19$

We need to know $\lambda(N)$, the smallest number where $a^{\lambda(N)} \equiv 1 \bmod N$ for every a coprime to N

worked example with N = 323 = 17 * 19

$\lambda(N) = 144$

We need to know $\lambda(N)$, the smallest number where $a^{\lambda(N)} \equiv 1 \bmod N$ for every a coprime to N

$$\lambda(N) = \mathrm{lcm}(\lambda(p), \lambda(q)) = \mathrm{lcm}(p-1, q-1) = \mathrm{lcm}(16, 18) = 144$$

worked example with $N = 323 = 17 * 19$

$\lambda(N) = 144$

We need to know $\lambda(N)$, the smallest number where $a^{\lambda(N)} \equiv 1 \bmod N$ for every a coprime to N

$\lambda(N) = \text{lcm}(\lambda(p), \lambda(q)) = \text{lcm}(p\text{-}1, q\text{-}1) = \text{lcm}(16, 18) = 144$

Choose e between 2 and N coprime to N; let's pick 5

worked example with N = 323 = 17 * 19

$\lambda(\mathrm{N}) = 144$ $e = 5; d = 29$

We need to know $\lambda(\mathrm{N})$, the smallest number where $a^{\lambda(\mathrm{N})} \equiv 1 \bmod \mathrm{N}$ for every a coprime to N

$\lambda(\mathrm{N}) = \mathrm{lcm}(\lambda(p), \lambda(q)) = \mathrm{lcm}(p-1, q-1) = \mathrm{lcm}(16, 18) = 144$

Choose e between 2 and N coprime to N; let's pick 5

Find d such that $d * e \equiv 1 \bmod \lambda(\mathrm{N})$; this is 29

@eli.holderness.dev

worked example with N = 323 = 17 * 19

$\lambda(N) = 144$ $\quad$ $e = 5; d = 29$

Our public key is $(N, e) = (323, 5)$ and our private key is $d = 29$

worked example with N = 323 = 17 * 19

$\lambda(N) = 144$ $e = 5; d = 29$

Our public key is $(N, e) = (323, 5)$ and our private key is $d = 29$

Someone wants to send us the message NDC = 14, 4, 3

worked example with $N = 323 = 17 * 19$

$\lambda(N) = 144$ $e = 5; d = 29$

Our public key is $(N, e) = (323, 5)$ and our private key is $d = 29$

Someone wants to send us the message NDC = 14, 4, 3

To encrypt a number, they raise it to the power of $e = 5$:

$14^5, 4^5, 3^5 = 537824, 1024, 243$

worked example with N = 323 = 17 * 19

$\lambda(N) = 144$ $\quad$ $e = 5; d = 29$ $\quad$ $m = (29, 55, 243)$

Our public key is $(N, e) = (323, 5)$ and our private key is $d = 29$

Someone wants to send us the message NDC = 14, 4, 3

To encrypt a number, they raise it to the power of $e = 5$:

$14^5, 4^5, 3^5 = 537824, 1024, 243$

Then take the modulus of N:

$14^5, 4^5, 3^5 \equiv 29, 55, 243 \pmod{N}$

@eli.holderness.dev

worked example with N = 323 = 17 * 19

$\lambda(N) = 144$ $e = 5; d = 29$ $m = (29, 55, 243)$

We received the message (29, 55, 243)

@eli.holderness.dev

worked example with $N = 323 = 17 * 19$

$\lambda(N) = 144$ $e = 5; d = 29$ $m = (29, 55, 243)$

We received the message (29, 55, 243)

Decode by raising each number to the power of $d = 29$, then taking the modulus of N

worked example with N = 323 = 17 * 19

$\lambda(N) = 144$ $e = 5; d = 29$ $m = (29, 55, 243)$

We received the message (29, 55, 243)

Decode by raising each number to the power of $d = 29$, then taking the modulus of N

$29^{29}, 55^{29}, 243^{29} \equiv 14, 4, 3 \bmod N$

@eli.holderness.dev

worked example with $N = 323 = 17 * 19$

$\lambda(N) = 144$ $e = 5;\ d = 29$ $m = (29, 55, 243)$

worked example with N = 323 = 17 * 19

$\lambda(\mathrm{N}) = 144$ $e = 5$; $d = 29$ $m = (29, 55, 243)$

This works because $a^{\lambda(\mathrm{N})} \equiv 1 \bmod \mathrm{N}$ for every a coprime to N

worked example with $N = 323 = 17 * 19$

$\lambda(N) = 144$ $e = 5; d = 29$ $m = (29, 55, 243)$

This works because $a^{\lambda(N)} \equiv 1 \bmod N$ for every a coprime to N

so $a^{\lambda(N)+1} \equiv a \bmod N$ for every a coprime to N

worked example with N = 323 = 17 * 19

$\lambda(N) = 144$ $e = 5; d = 29$ $m = (29, 55, 243)$

This works because $a^{\lambda(N)} \equiv 1 \bmod N$ for every a coprime to N

so $a^{\lambda(N)+1} \equiv a \bmod N$ for every a coprime to N

$$a^{\lambda(N)+1} = a^{145} = a^{5 \times 29} = (a^5)^{29}$$

@eli.holderness.dev

worked example with N = 323 = 17 * 19

$\lambda(\mathrm{N}) = 144$ $\quad e = 5; d = 29$ $\quad m = (29, 55, 243)$

This works because $a^{\lambda(\mathrm{N})} \equiv 1 \bmod \mathrm{N}$ for every a coprime to N

so $a^{\lambda(\mathrm{N})+1} \equiv a \bmod \mathrm{N}$ for every a coprime to N

$$a^{\lambda(\mathrm{N})+1} = a^{145} = a^{5 \times 29} = (a^5)^{29}$$

So $(a^5)^{29} \equiv a \bmod \mathrm{N}$ and we can recover the original message from the encrypted intermediate

@eli.holderness.dev

limitations & considerations

limitations & considerations

requires large prime numbers, which are expensive to find

limitations & considerations

requires large prime numbers, which are expensive to find

if e is small enough that $M = m^e < N$, an attacker can simply do $\sqrt[e]{M}$ to recover m

limitations & considerations

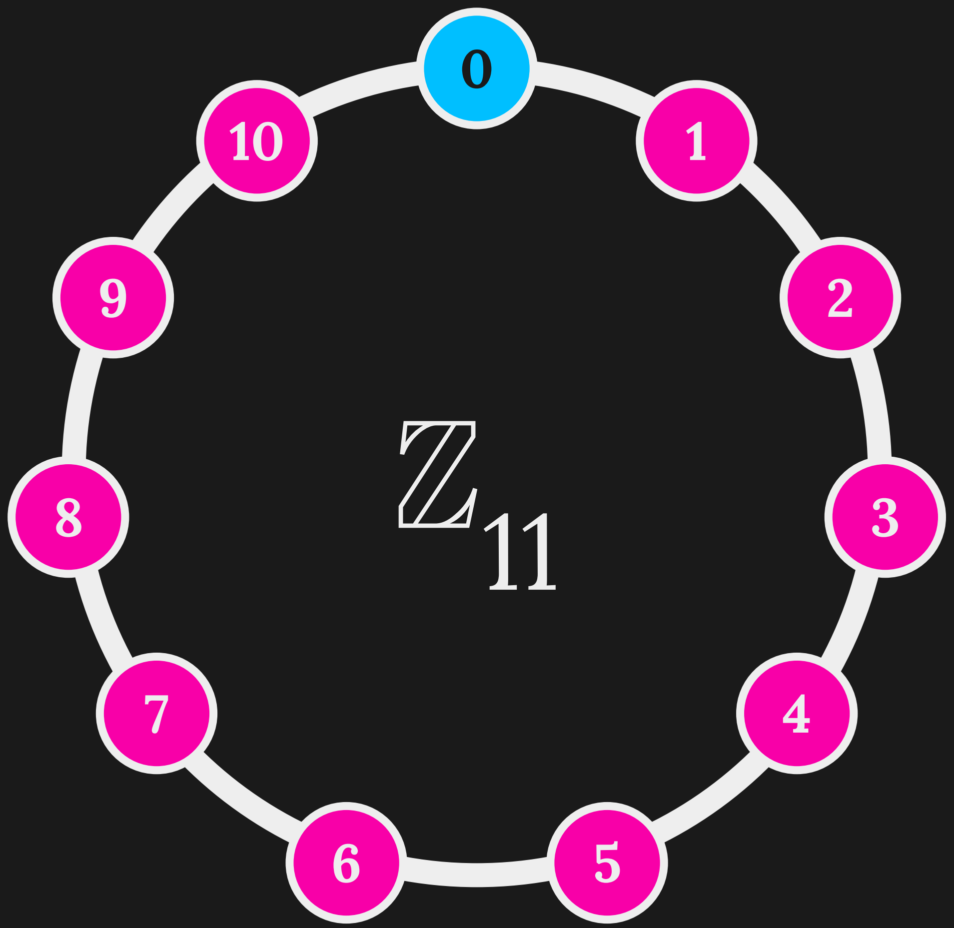
requires large prime numbers, which are expensive to find

if e is small enough that $M = m^e < N$, an attacker can simply do $\sqrt[e]{M}$ to recover m

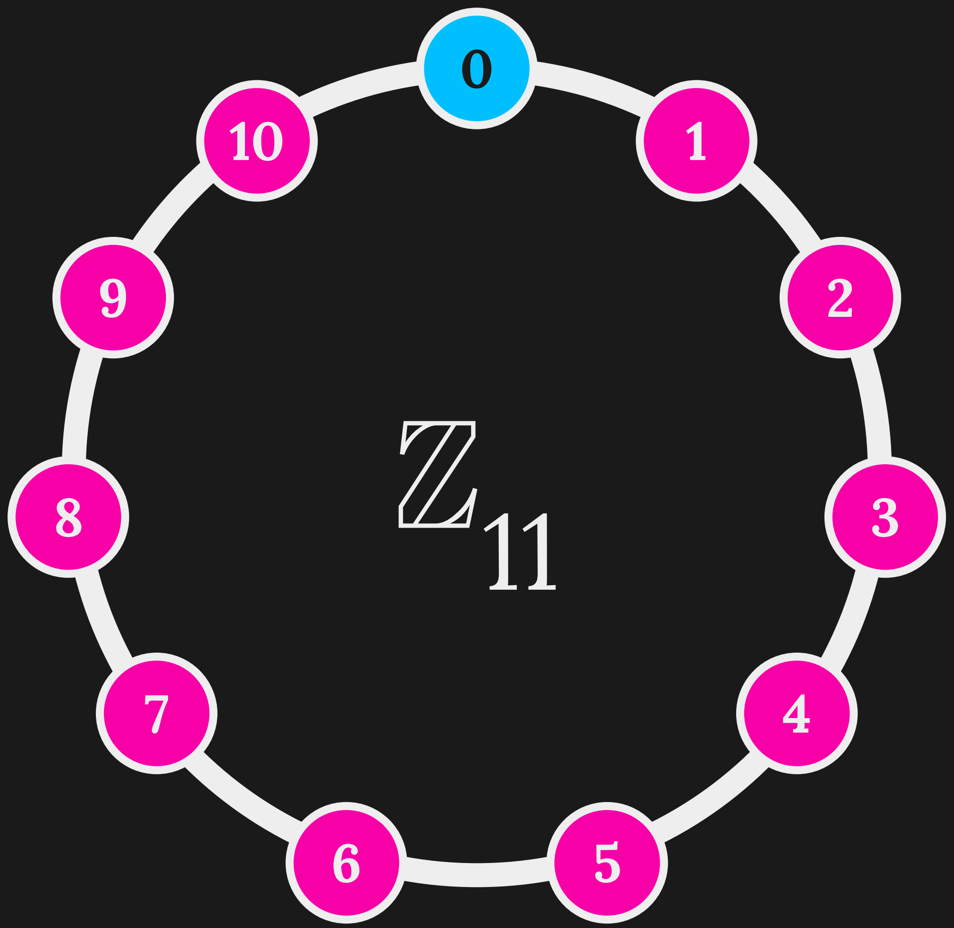
without padding, messages can be vulnerable to chosen plaintext attacks

TURKEY TROTS TO WATER GG
FROM CINCPAC ACTION COM
THIRD FLEET INFO COMINCH
CTF SEVENTY-SEVEN X WHERE
IS RPT WHERE IS TASK FORCE
THIRTY FOUR RR **THE WORLD**
WONDERS

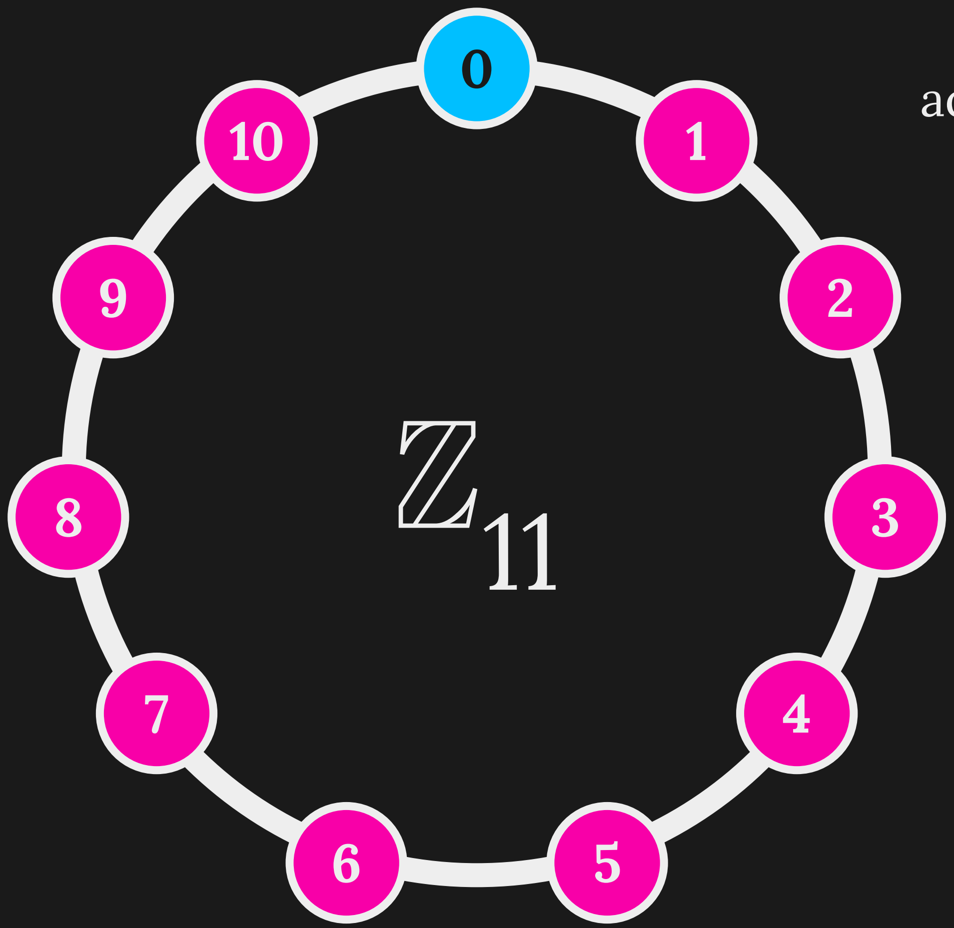
@eli.holderness.dev



@eli.holderness.dev

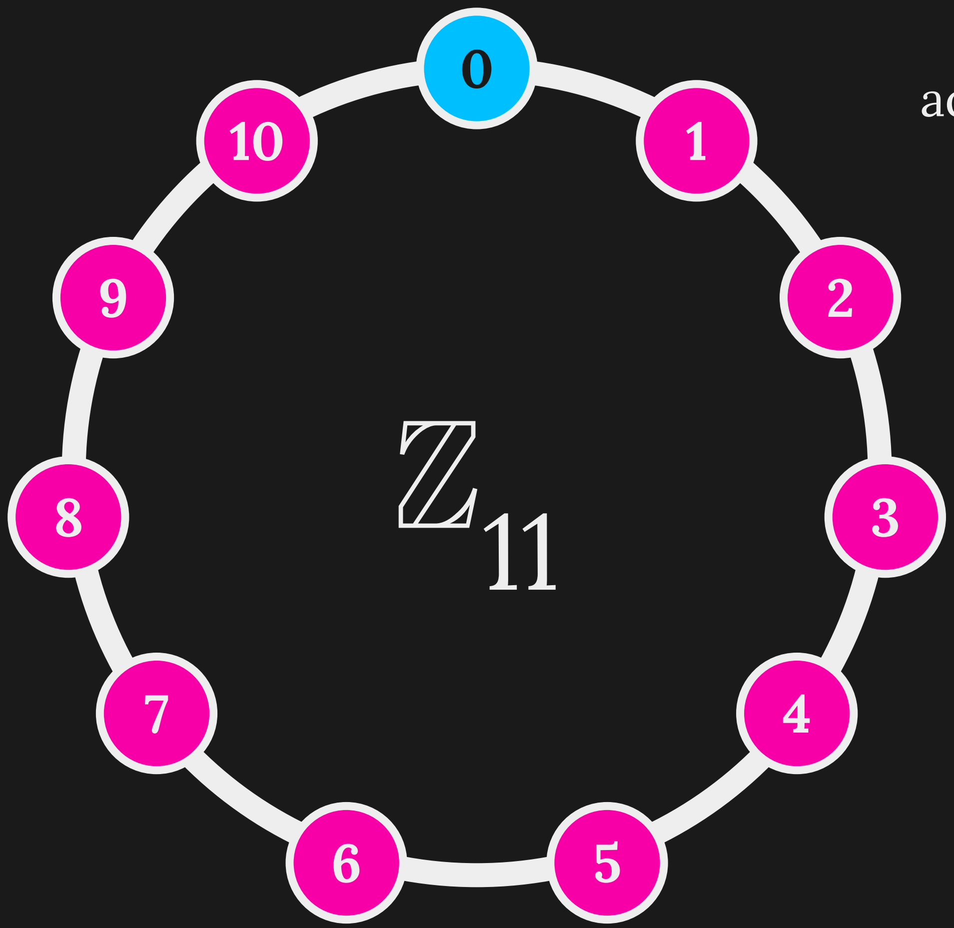


@eli.holderness.dev



identity element

adding 0 doesn't change an element

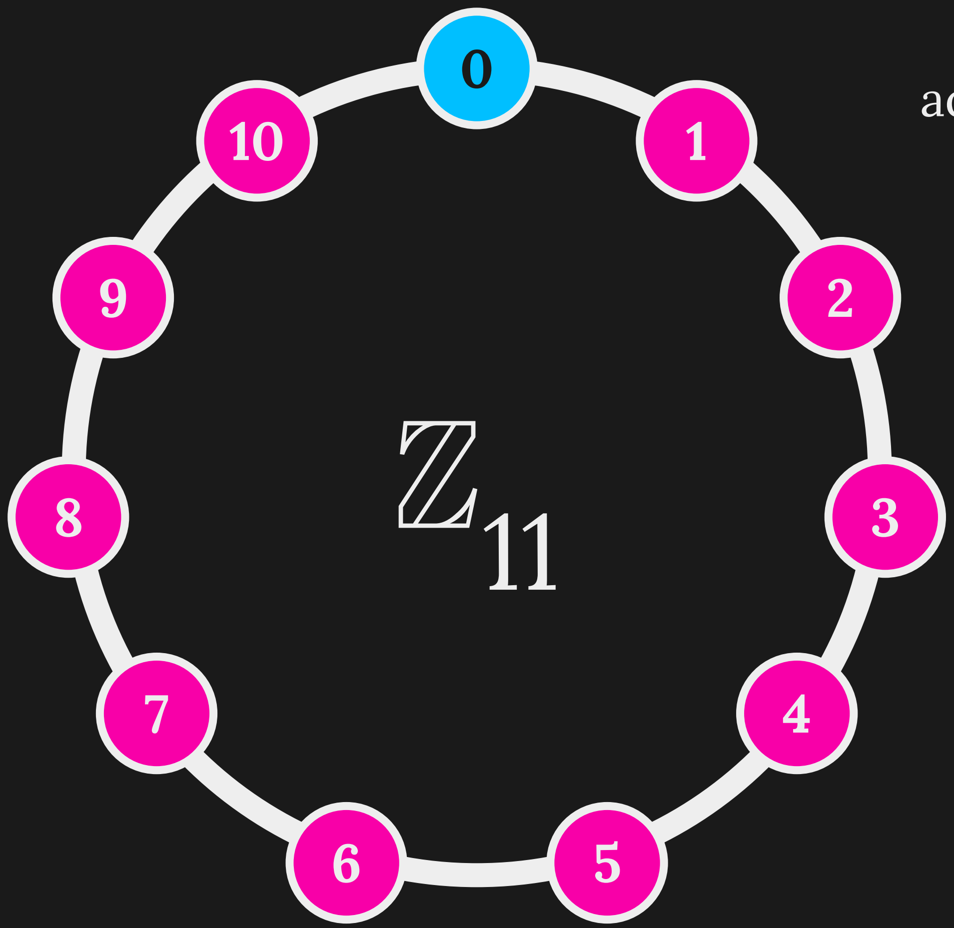


identity element

adding 0 doesn't change an element

inverses

for every a in the group, there's a b that makes $a + b = 0$ true



identity element

adding 0 doesn't change an element

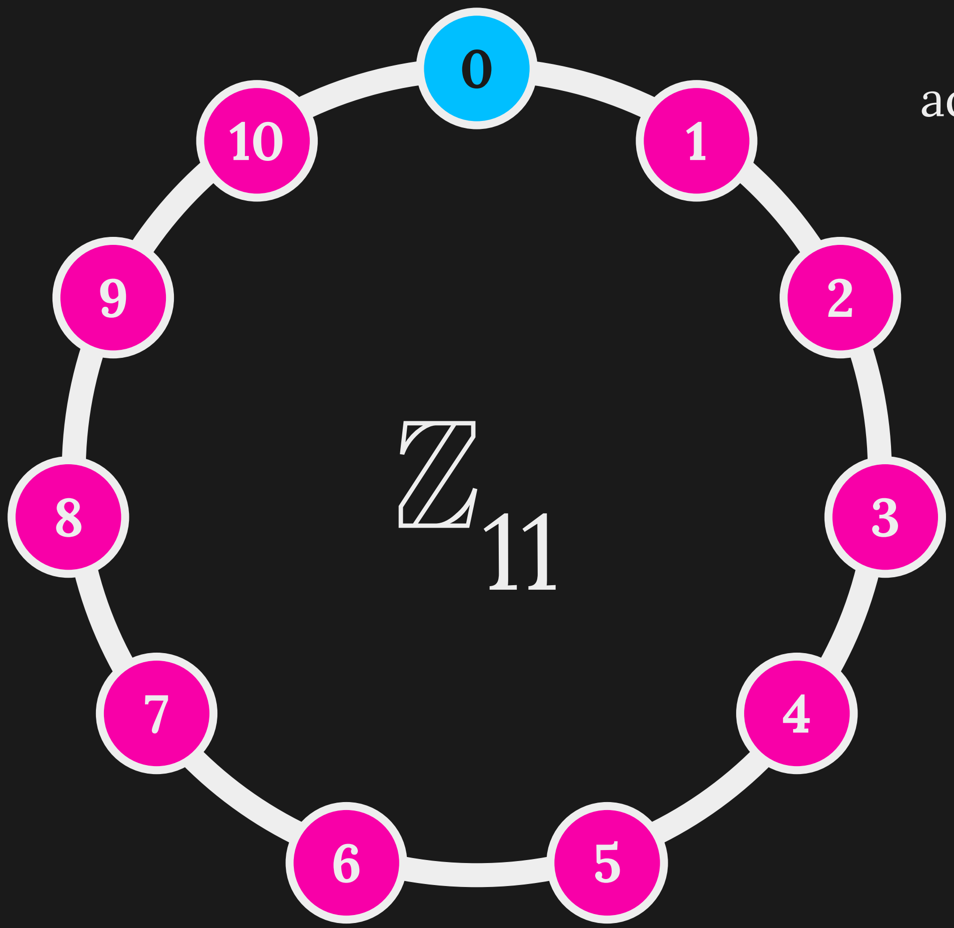
inverses

for every a in the group, there's a b that makes $a + b = 0$ true

associativity

$1 + (4 + 2) = (1 + 4) + 2$

@eli.holderness.dev



identity element

adding 0 doesn't change an element

inverses

for every a in the group, there's a b that makes $a + b = 0$ true

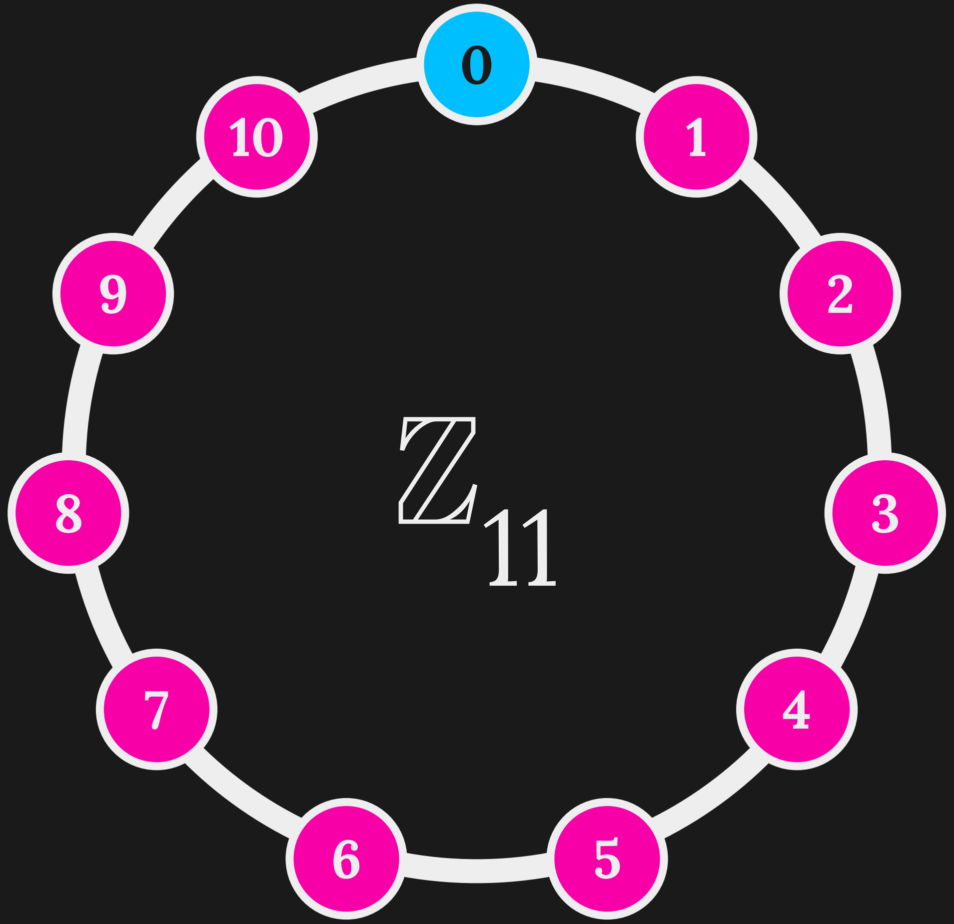
associativity

$1 + (4 + 2) = (1 + 4) + 2$

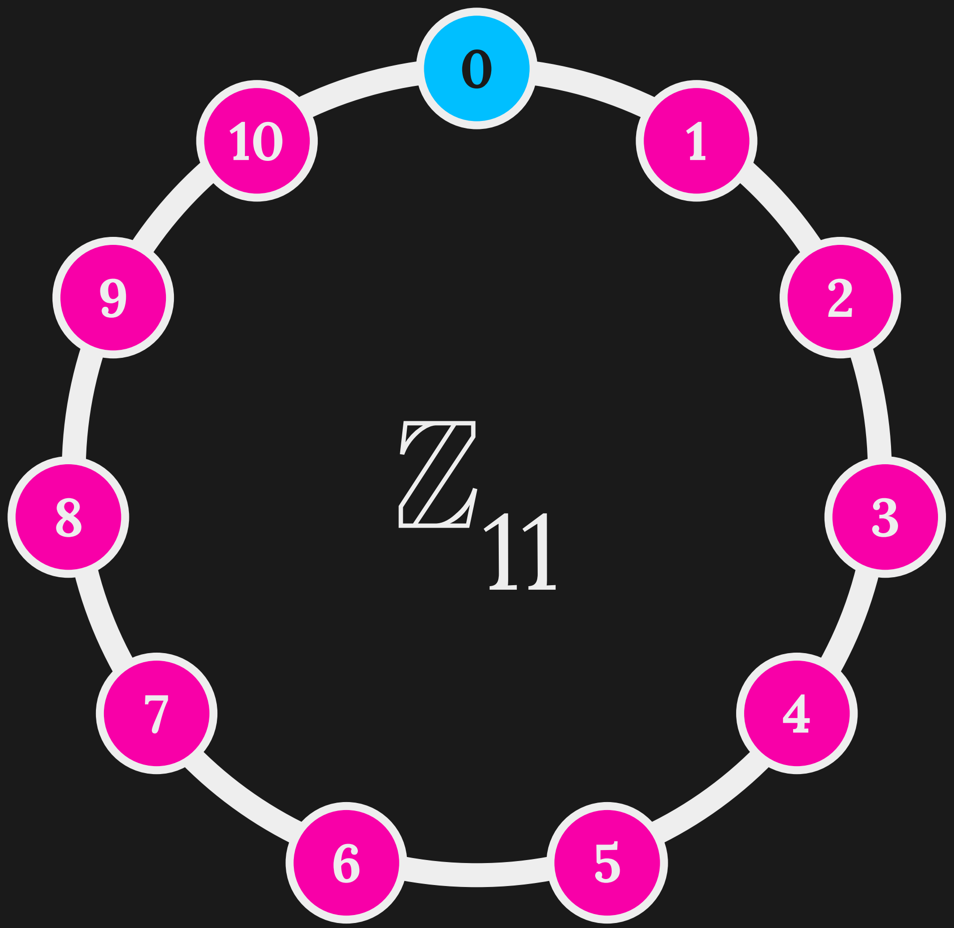
closure

If a and b are in the group and $a + b = c$, then c is in the group

@eli.holderness.dev

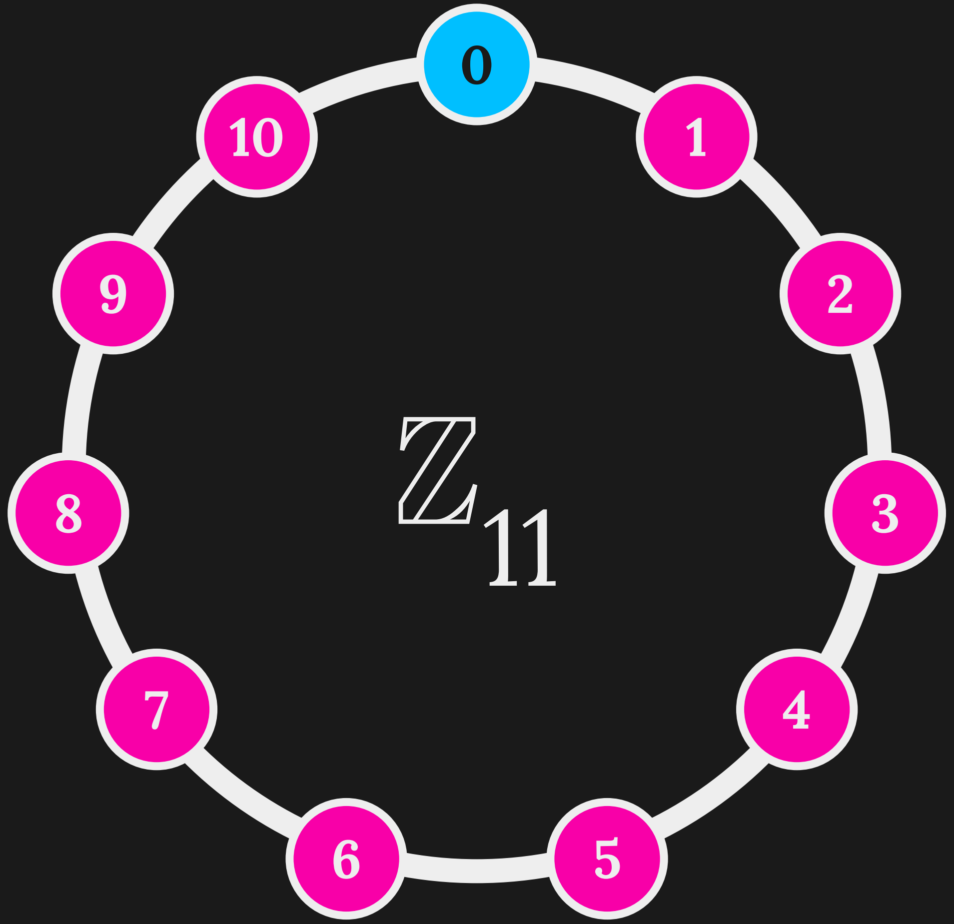


@eli.holderness.dev

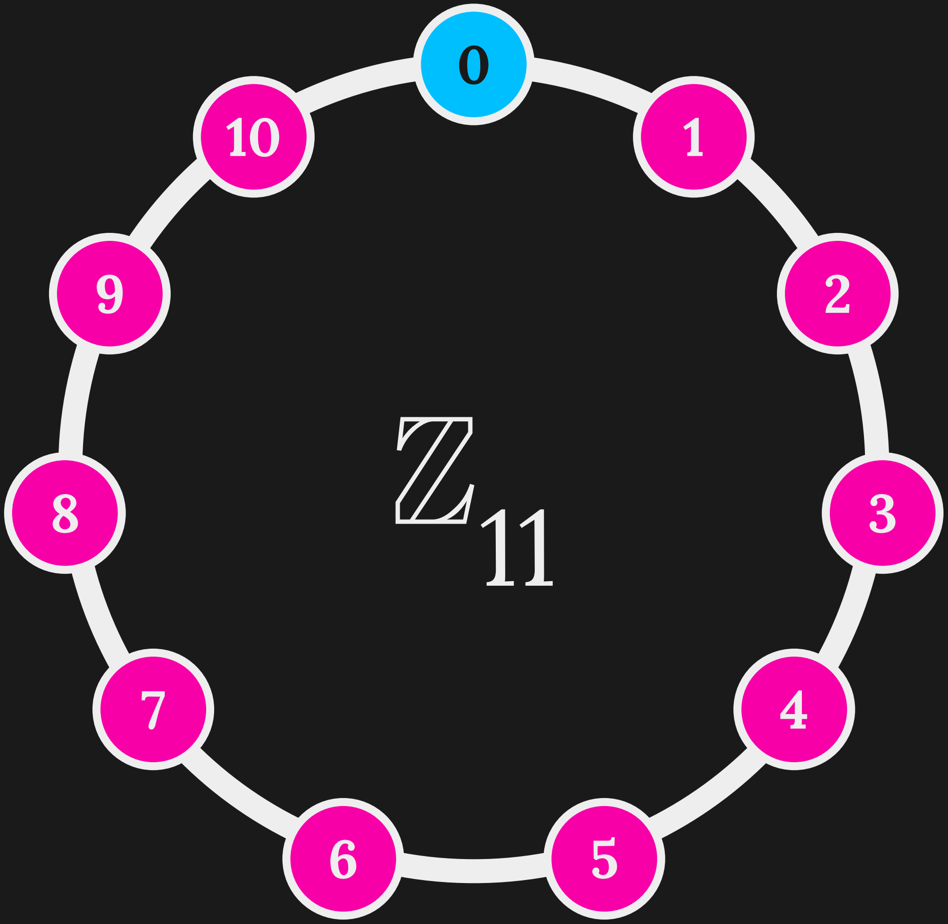


4 x 13 = 52

@eli.holderness.dev

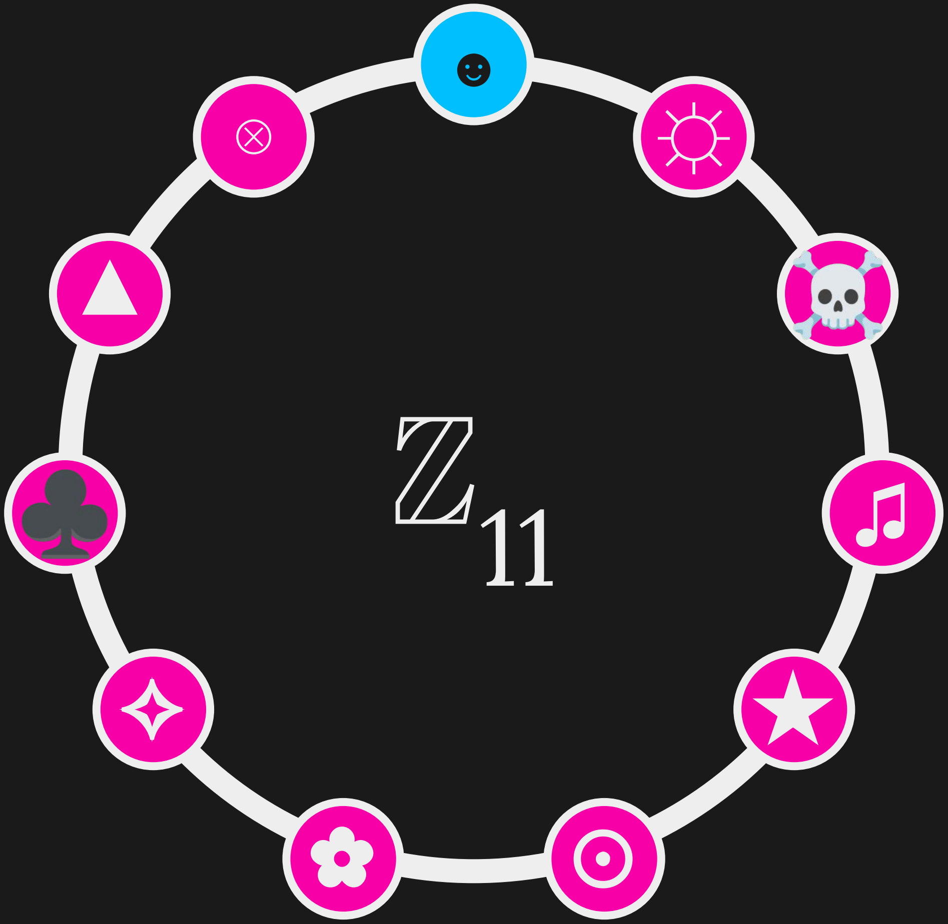


$$4 \times 13 = 52$$
$$= (4 \times 11) + 8$$
$$= 8$$



$$4 \times 13 = 52$$
$$= (4 \times 11) + 8$$
$$= 8$$

you can multiply an element of the group by something that is NOT in the group



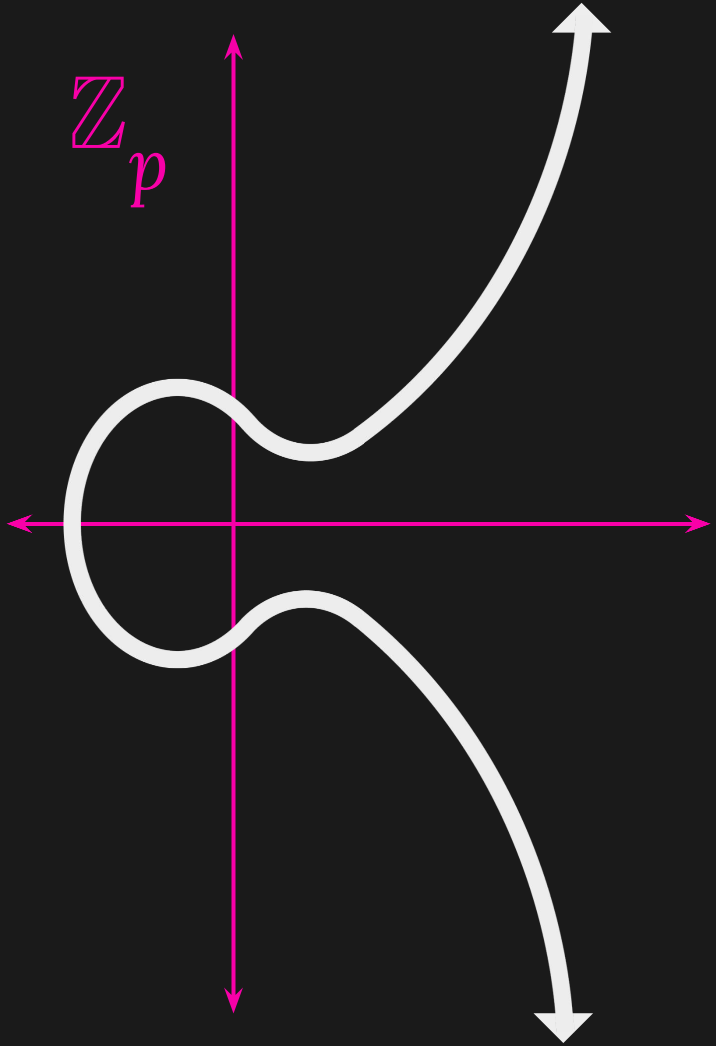
 $\times\ 13 = 52$

$= (4 \times 11) + 8$

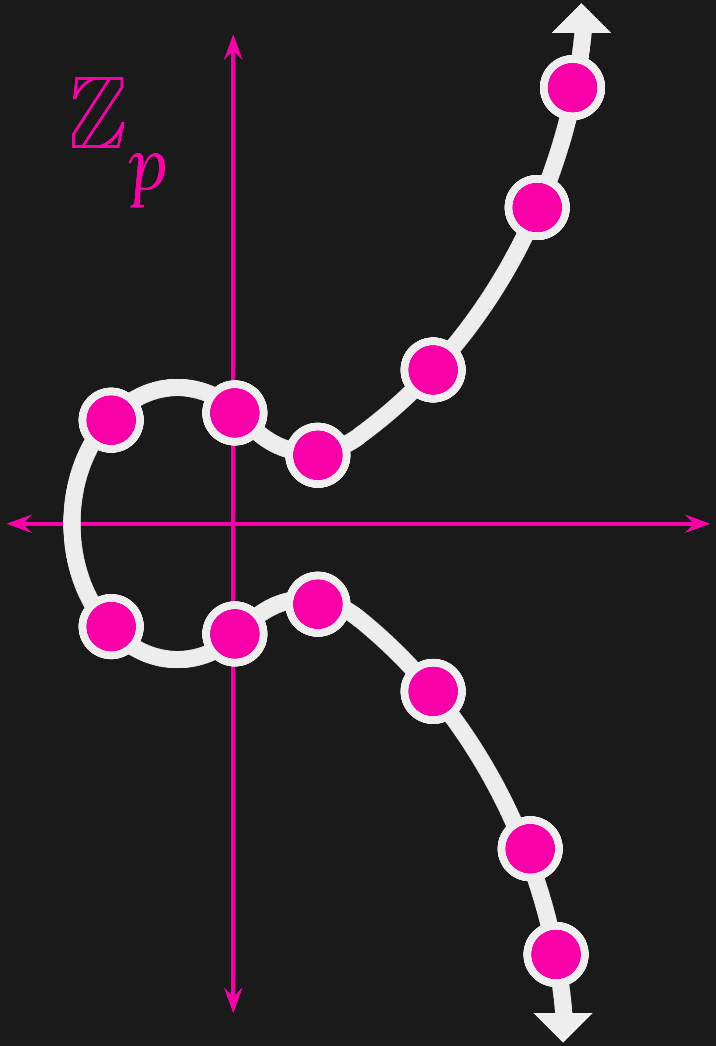
$=$ 

you can multiply an element of the group by something that is NOT in the group

Elliptic Curve Cryptography

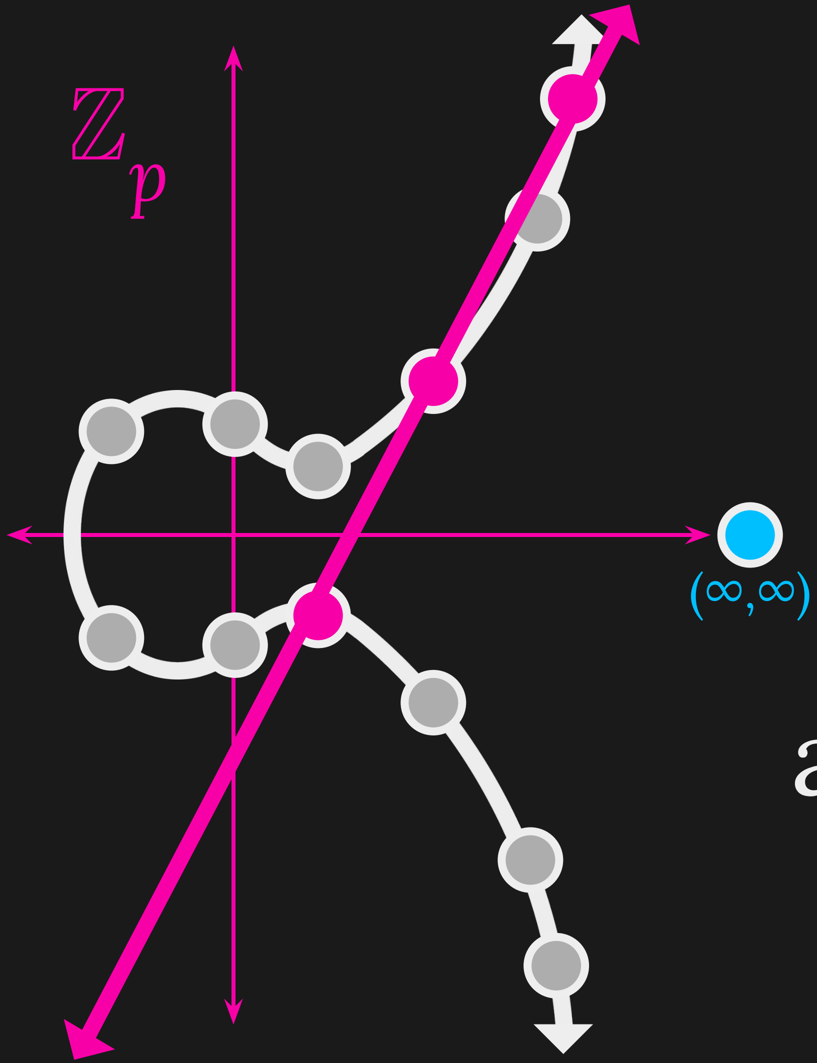


$$y^2 \equiv x^3 + \mathrm{a}x + \mathrm{b}$$



$$y^2 \equiv x^3 + ax + b$$

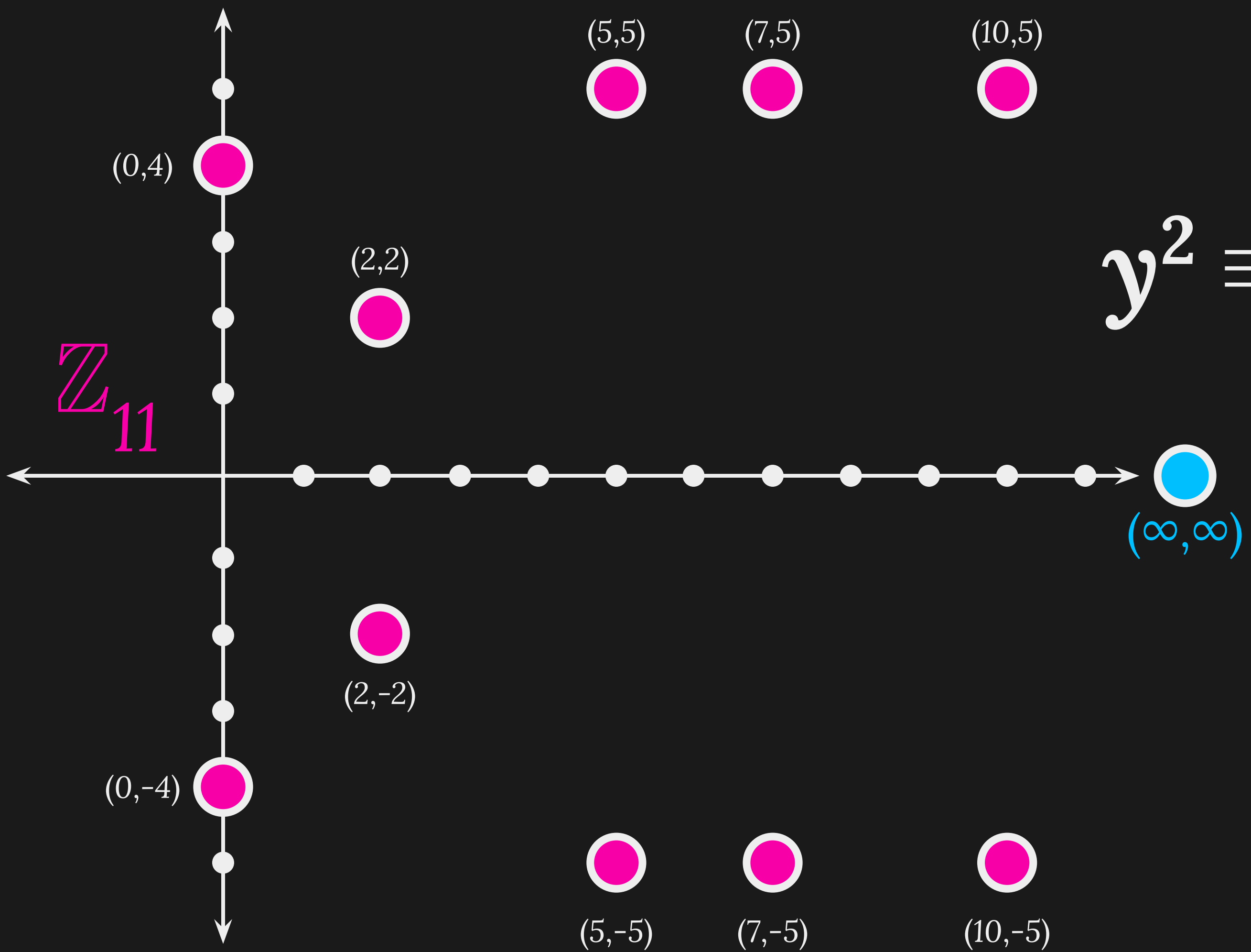
where x and y are in $\mathbb{Z}_p$



$$y^2 \equiv x^3 + ax + b$$

where x and y are in $\mathbb{Z}_p$

and three collinear points 'sum' to O



$$y^2 \equiv x^3 + x + 5$$

elliptic curve domain parameters over $\boldsymbol{F}_p$

$$T = (p, a, b, \mathrm{G}, n, h)$$

Certicom Research: http://www.secg.org/sec2-v2.pdf

elliptic curve domain parameters over F_p

$$T = (p, a, b, G, n, h)$$

an integer defining
the field F_p

Certicom Research: http://www.secg.org/sec2-v2.pdf

@eli.holderness.dev

elliptic curve domain parameters over F_p

$$T = (p, a, b, \mathrm{G}, n, h)$$

two elements of F_p defining

$$\mathrm{E}: y^2 \equiv x^3 + ax + b$$

Certicom Research: http://www.secg.org/sec2-v2.pdf

@eli.holderness.dev

elliptic curve domain parameters over $\boldsymbol{F}_p$

$$T = (p, a, b, G, n, h)$$

a point on $E(F_p)$ written as

$$G = (x_G, y_G)$$

Certicom Research: http://www.secg.org/sec2-v2.pdf

@eli.holderness.dev

elliptic curve domain parameters over $\boldsymbol{F}_p$

$$T = (p, a, b, \mathrm{G}, n, h)$$

the *order* of G in $E(F_p)$ - i.e.,

$$n \times \mathrm{G} = \mathbf{O}$$

Certicom Research: http://www.secg.org/sec2-v2.pdf

@eli.holderness.dev

elliptic curve domain parameters over F_p

$$T = (p, a, b, \mathrm{G}, n, h)$$

the *cofactor* of G in $E(F_p)$, which is $|E(F_p)| \,/\, n$

Certicom Research: http://www.secg.org/sec2-v2.pdf

elliptic curve domain parameters over $\boldsymbol{F}_p$

$$T = (p, a, b, \mathrm{G}, n, h)$$

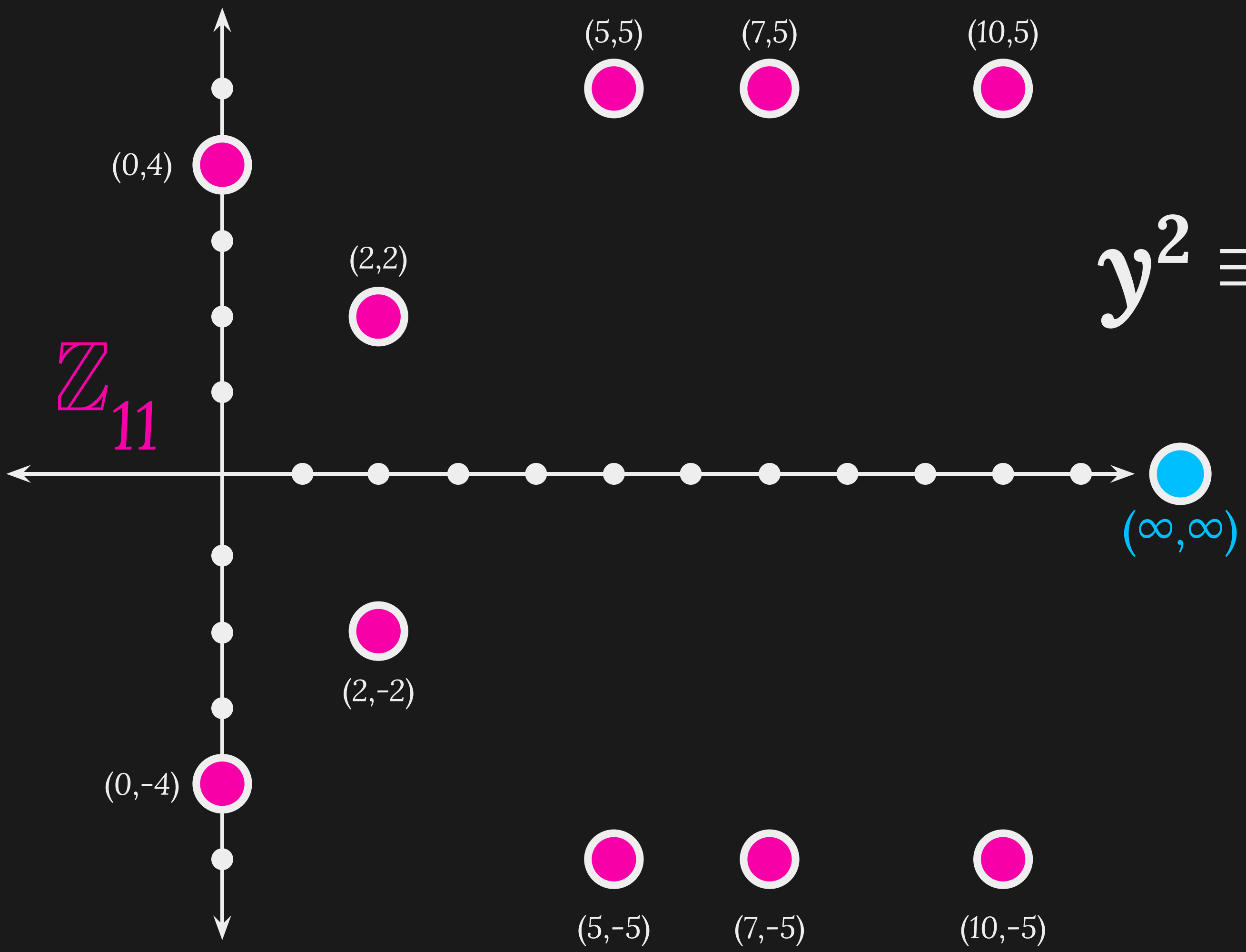
or more properly, $orb(\mathrm{G})$

the *cofactor* of G in $E(F_p)$, which is

$$|E(F_p)| \,/\, n$$

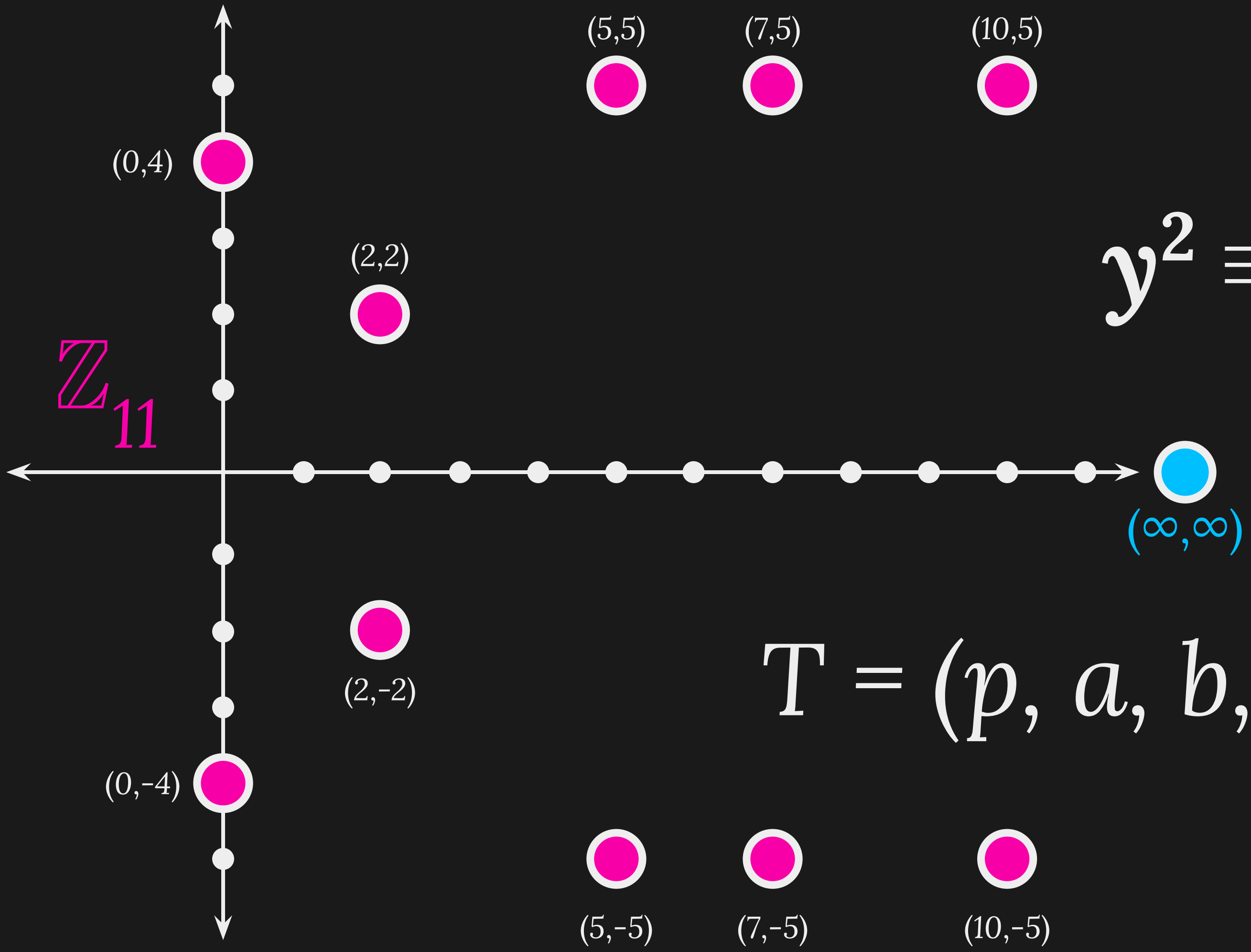
Certicom Research: http://www.secg.org/sec2-v2.pdf

@eli.holderness.dev



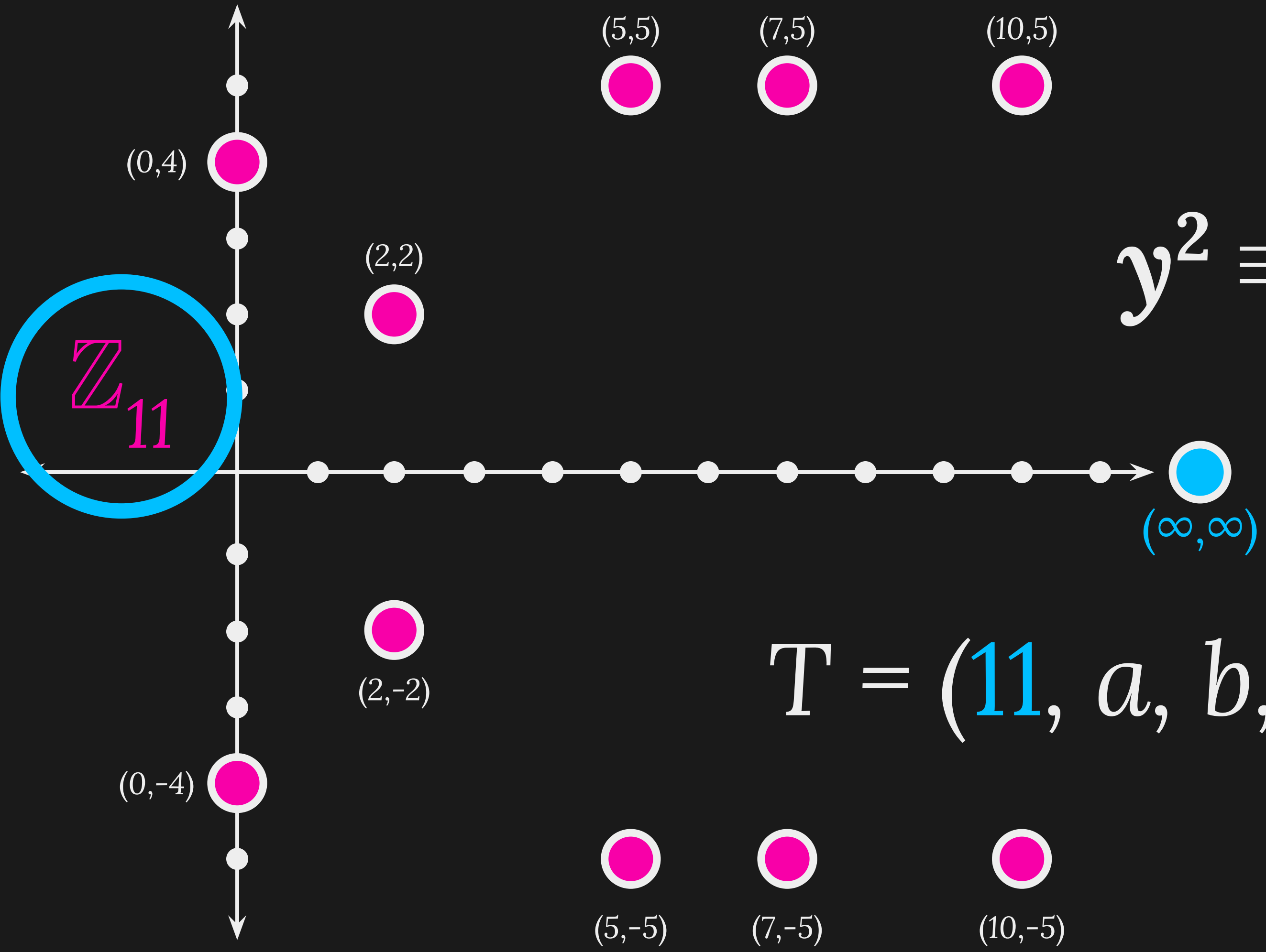
$$y^2 \equiv x^3 + x + 5$$

@eli.holderness.dev



$$y^2 \equiv x^3 + x + 5$$

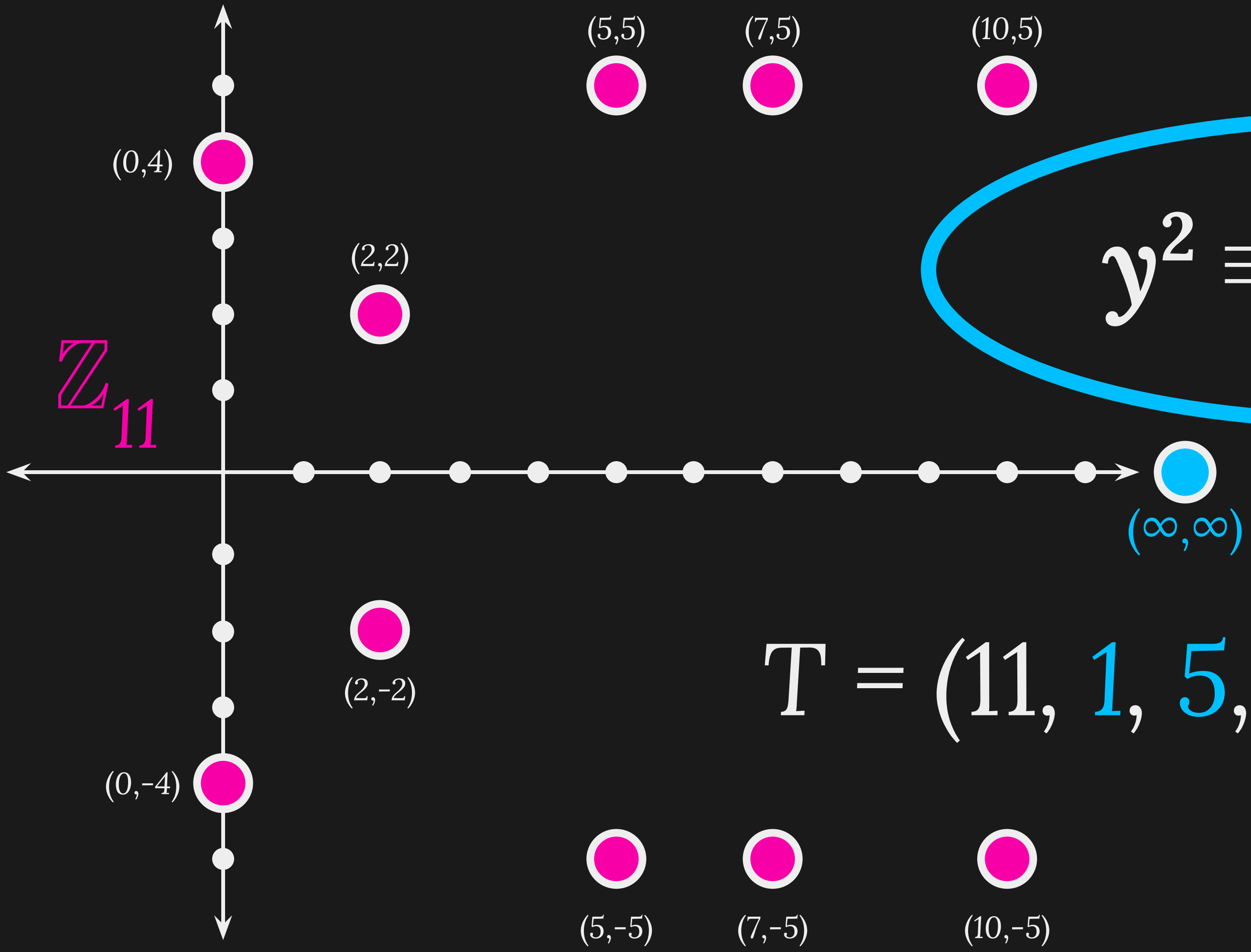
$$T = (p, a, b, G, n, h)$$



$$y^2 \equiv x^3 + x + 5$$

$$T = (11, a, b, G, n, h)$$

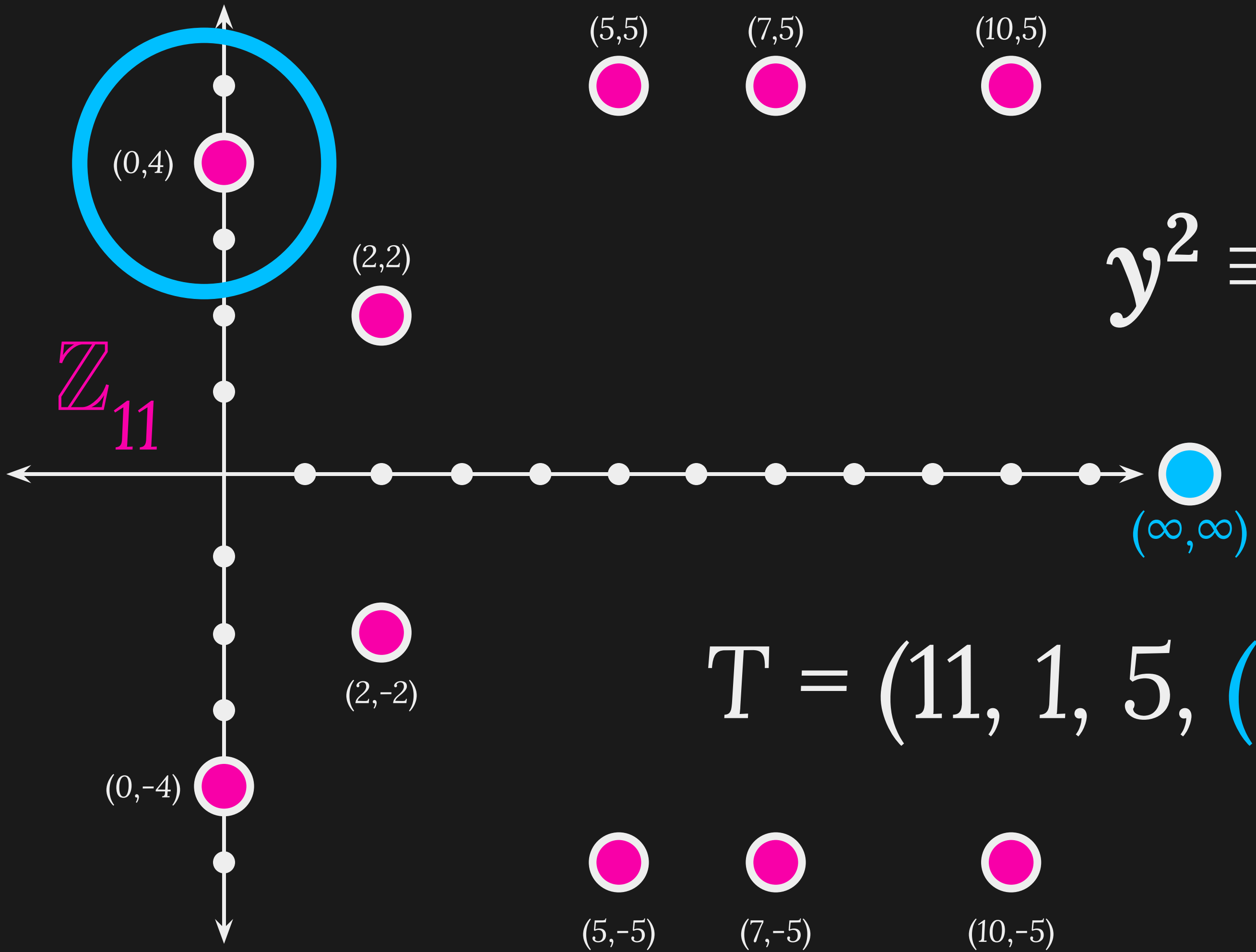
@eli.holderness.dev



$$y^2 \equiv x^3 + x + 5$$

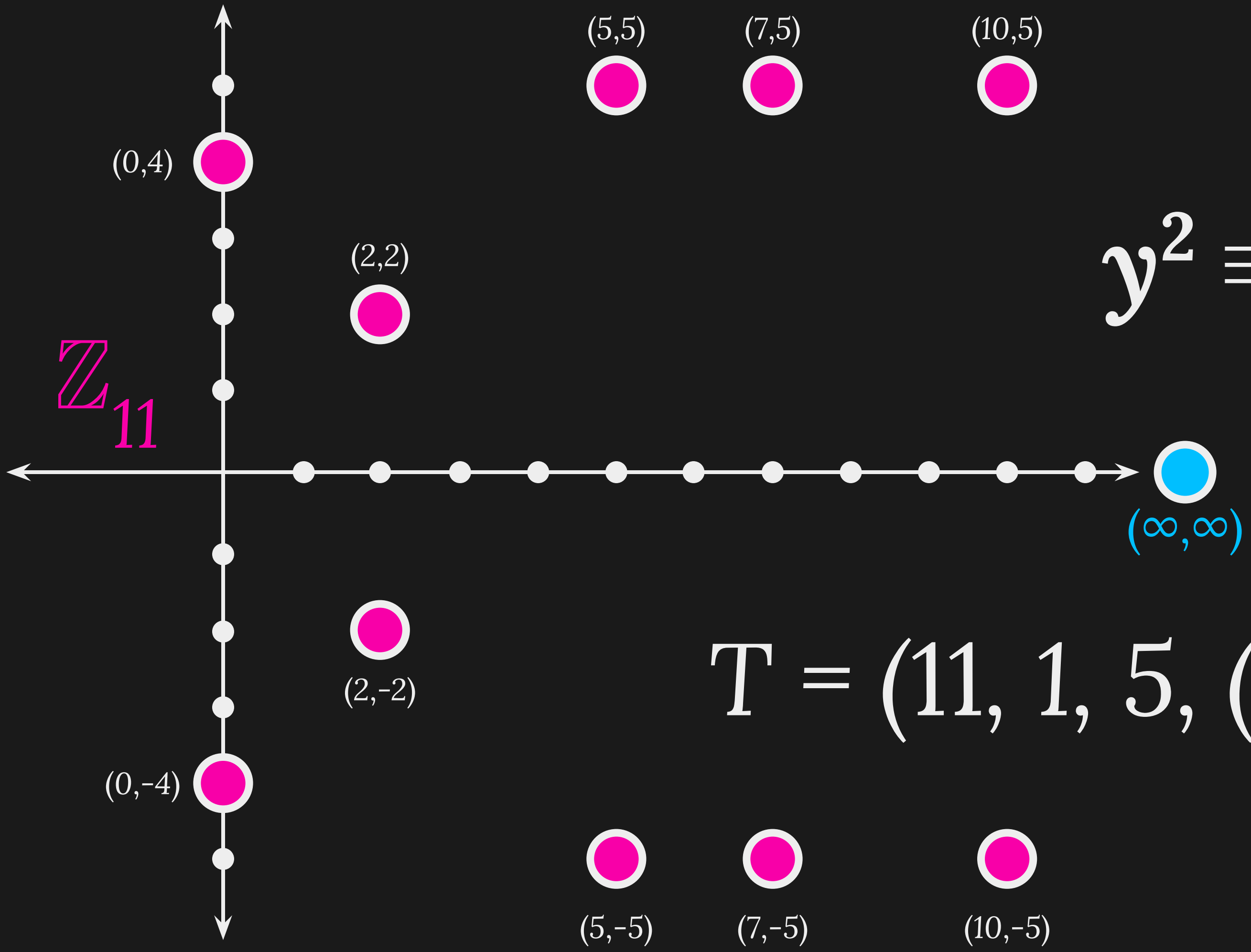
$$T = (11, 1, 5, G, n, h)$$

@eli.holderness.dev



$$y^2 \equiv x^3 + x + 5$$

$$T = (11,\ 1,\ 5,\ (0,4),\ n,\ h)$$



$$y^2 \equiv x^3 + x + 5$$

$$T = (11,\ 1,\ 5,\ (0,4),\ 11,\ 1)$$

@eli.holderness.dev

$1 \times G = (0, 4)$

$2 \times G = (5, 5)$

$3 \times G = (10, 5)$

$4 \times G = (2, -2)$

$5 \times G = (7, -5)$

$6 \times G = (7, 5)$

$7 \times G = (2, 2)$

$8 \times G = (10, -5)$

$9 \times G = (5, -5)$

$10 \times G = (0, -4)$

$$11 \times G = (\infty, \infty)$$

@eli.holderness.dev

$$1 \times G = (0, 4)$$
$$2 \times G = (5, 5)$$
$$3 \times G = (10, 5)$$
$$4 \times G = (2, -2)$$
$$5 \times G = (7, -5)$$
$$6 \times G = (7, 5)$$
$$7 \times G = (2, 2)$$
$$8 \times G = (10, -5)$$
$$9 \times G = (5, -5)$$
$$10 \times G = (0, -4)$$

$$11 \times G = (\infty, \infty)$$

@eli.holderness.dev

worked example

worked example with $T = (11, 1, 5, (0,4), 11, 1)$

worked example with $T = (11, 1, 5, (0,4), 11, 1)$

$d_{PK} = 3$

Pick a random number d_{PK} from $[1, \ldots\ n-1] = [1, \ldots\ 10]$.
Let's pick 3. This is our private key.

worked example with $T = (11, 1, 5, (0,4), 11, 1)$

$d_{PK} = 3 \quad Q_{PK} = (10, 5)$

Pick a random number d_{PK} from $[1,\dots n\text{-}1] = [1,\dots 10]$.
Let's pick 3. This is our private key.

Calculate $Q_{PK} = d_{PK} \times G$, which in our case is
$3 \times (0,4) = (10, 5)$. This is our public curve point.

worked example with $T = (11, 1, 5, (0,4), 11, 1)$

$d_{PK} = 3 \quad Q_{PK} = (10, 5)$

We have some binary message, e, to sign. Let's say we want to sign the message 01001110 01000100 01000011.

worked example with $T = (11, 1, 5, (0,4), 11, 1)$

$z = 3$ $d_{PK} = 3$ $Q_{PK} = (10, 5)$

We have some binary message, *e*, to sign. Let's say we want to sign the message 01001110 01000100 01000011.

The size of our group is 11, or 1101 in binary - 4 bits long. Take the last 4 bits of our message: 0011. Call it z.

worked example with $T = (11, 1, 5, (0,4), 11, 1)$

$z = 3$ $d_{PK} = 3$ $Q_{PK} = (10, 5)$

Pick another random number k from $[1,...n-1]$. This time let's choose 5. This must be random per signature.

worked example with $T = (11, 1, 5, (0,4), 11, 1)$

$k^{-1} = 9$ $z = 3$ $d_{PK} = 3$ $Q_{PK} = (10, 5)$

Pick another random number k from $[1,\ldots n-1]$. This time let's choose 5. This must be random per signature.

Find its inverse k^{-1} in $\mathbf{F}_{11}$, which is 9.

worked example with $T = (11, 1, 5, (0,4), 11, 1)$

$x_k = 7, y_k = -5$ $k^{-1} = 9$ $z = 3$ $d_{PK} = 3$ $Q_{PK} = (10, 5)$

Pick another random number k from $[1,...n-1]$. This time let's choose 5. This must be random per signature.

Find its inverse k^{-1} in $\mathbf{F}_{11}$, which is 9.

Calculate $k \times G = 5 \times (0,4) = (7, -5)$. Take its coordinates, so we have $x_k = 7, y_k = -5$

@eli.holderness.dev

worked example with $T = (11, 1, 5, (0,4), 11, 1)$

$x_k = 7, y_k = -5$ $k^{-1} = 9$ $z = 3$ $d_{PK} = 3$ $Q_{PK} = (10, 5)$

Now calculate r and s, where

$r \equiv x_k \bmod n$ and $s \equiv k^{-1}(z + r * d_{PK}) \bmod n$

@eli.holderness.dev

worked example with $T = (11, 1, 5, (0,4), 11, 1)$

$x_k = 7, y_k = -5$ $k^{-1} = 9$ $z = 3$ $d_{PK} = 3$ $Q_{PK} = (10, 5)$

Now calculate r and s, where

$r \equiv x_k \bmod n$ and $s \equiv k^{-1}(z + r * d_{PK}) \bmod n$

This gives us $r = 7$ and $s = 7$, and this is our signature:

$(r,s) = (7, 7)$.

@eli.holderness.dev

worked example with $T = (11, 1, 5, (0,4), 11, 1)$

$x_k = 7, y_k = -5$ $k^{-1} = 9$ $z = 3$ $d_{PK} = 3$ $Q_{PK} = (10, 5)$

Now calculate r and s, where

$r \equiv x_k \bmod n$ and $s \equiv k^{-1}(z + r * d_{PK}) \bmod n$

This gives us $r = 7$ and $s = 7$, and this is our signature:

$(r,s) = (7, 7)$.

If either r or s are 0, we have to go back and pick a different k.

@eli.holderness.dev

worked example with $T = (11, 1, 5, (0,4), 11, 1)$

$x_k = 7, y_k = -5$ $k^{-1} = 9$ $z = 3$ $d_{PK} = 3$ $Q_{PK} = (10, 5)$

We've now generated a signature $(r,s) = (7, 7)$ over the binary message 01001110 01000100 01000011.

Let's verify it!

worked example with $T = (11, 1, 5, (0,4), 11, 1)$

$r = 7, s = 7$ $\qquad$ $Q_{PK} = (10, 5)$

worked example with $T = (11, 1, 5, (0,4), 11, 1)$

$z = 3$ $r = 7, s = 7$ $Q_{PK} = (10, 5)$

We have the message, 01001110 01000100 01000011.
Take the last 4 bits as we did before to get $z = 3$.

worked example with $T = (11, 1, 5, (0,4), 11, 1)$

$u_1 = 2, u_2 = 5$ $z = 3$ $r = 7, s = 7$ $Q_{PK} = (10, 5)$

We have the message, 01001110 01000100 01000011.
Take the last 4 bits as we did before to get $z = 3$.

Calculate $u_1 \equiv zs^{-1} \bmod n$: $u_1 \equiv 3*8 \equiv 2 \bmod 11$
Calculate $u_2 \equiv rs^{-1} \bmod n$: $u_2 \equiv 7*7 \equiv 5 \bmod 11$

worked example with $T = (11, 1, 5, (0,4), 11, 1)$

$u_1 = 2, u_2 = 5$ $z = 3$ $r = 7, s = 7$ $Q_{PK} = (10, 5)$

Calculate a new point on the curve, $(x, y) = u_1 \times G + u_2 \times Q_{PK}$

@eli.holderness.dev

worked example with $T = (11, 1, 5, (0,4), 11, 1)$

$u_1 = 2, u_2 = 5$ $z = 3$ $r = 7, s = 7$ $Q_{PK} = (10, 5)$

Calculate a new point on the curve, $(x, y) = u_1 \times G + u_2 \times Q_{PK}$

$$u_1 \times G = 2 \times (0,4)$$

$$u_2 \times Q_{PK} = 5 \times (10,5) = 5 \times (3 \times (0,4)) = 4 \times (0,4)$$

@eli.holderness.dev

worked example with $T = (11, 1, 5, (0,4), 11, 1)$

$u_1 = 2, u_2 = 5$ $z = 3$ $r = 7, s = 7$ $Q_{PK} = (10, 5)$

Calculate a new point on the curve, $(x, y) = u_1 \times G + u_2 \times Q_{PK}$

$$u_1 \times G = 2 \times (0,4)$$

$$u_2 \times Q_{PK} = 5 \times (10,5) = 5 \times (3 \times (0,4)) = 4 \times (0,4)$$

so $(x, y) = 2 \times (0,4) + 4 \times (0,4) = 6 \times (0,4) = (7,5)$

worked example with $T = (11, 1, 5, (0,4), 11, 1)$

$u_1 = 2, u_2 = 5$ $z = 3$ $r = 7, s = 7$ $Q_{PK} = (10, 5)$

Calculate a new point on the curve, $(x, y) = u_1 \times G + u_2 \times Q_{PK}$

$$u_1 \times G = 2 \times (0,4)$$

$$u_2 \times Q_{PK} = 5 \times (10,5) = 5 \times (3 \times (0,4)) = 4 \times (0,4)$$

so $(x, y) = 2 \times (0,4) + 4 \times (0,4) = 6 \times (0,4) = (7,5)$

The signature is valid if $x = r \bmod n$, which it is!

@eli.holderness.dev

comparison with RSA

comparison with RSA

smaller key size per security

comparison with RSA

smaller key size per security

smaller payload size

comparison with RSA

smaller key size per security

smaller payload size

faster computation

@eli.holderness.dev

@eli.holderness.dev

Quantum Computing & Shor's Algorithms

the Integer Factorisation problem

if pq = N with p & q prime, find p and q given only N

the Integer Factorisation problem

if pq = N with p & q prime, find p and q given only N

the Discrete Logarithm problem

if g generates a subgroup of a finite field F, and y is another member of F, find x such that $g^x = y$

the Integer Factorisation problem

if pq = N with p & q prime, find p and q given only N

the Discrete Logarithm problem

if g generates a subgroup of a finite field F, and y is another member of F, find x such that $g^x = y$

the Elliptic Curve Discrete Logarithm problem

if G generates a subgroup of an elliptic curve over a field F, and P is another member of that elliptic curve, find k such that $P = kG$

Shor's order-finding algorithm

for a given number N, and any number a between 1 and N, we can find the smallest r such that $a^r \equiv 1 \bmod N$, in polynomial time

Shor’s order-finding algorithm

Shor’s order-finding algorithm

let $N = 323$. Choose $a = 11$.

Shor’s algorithm gives us that $11^{48} \equiv 1 \bmod 323$

Shor's order-finding algorithm

let N = 323. Choose a = 11.

Shor's algorithm gives us that $11^{48} \equiv 1 \bmod 323$

$11^{48} - 1 \equiv 0 \bmod 323$, so $(11^{24} - 1)(11^{24} + 1) \equiv 0 \bmod 323$,

which is equivalent to $323 \mid (11^{24} - 1)(11^{24} + 1)$

Shor's order-finding algorithm

let N = 323. Choose a = 11.

Shor's algorithm gives us that $11^{48} \equiv 1 \bmod 323$

$11^{48} - 1 \equiv 0 \bmod 323$, so $(11^{24} - 1)(11^{24} + 1) \equiv 0 \bmod 323$,

which is equivalent to $323 \mid (11^{24} - 1)(11^{24} + 1)$

we know 323 doesn't divide $11^{24} - 1$, or else we'd have

$11^{24} \equiv 1 \bmod 323$

Shor's order-finding algorithm

let N = 323. Choose a = 11.
Shor's algorithm gives us that $11^{48} \equiv 1 \bmod 323$

$11^{48} - 1 \equiv 0 \bmod 323$, so $(11^{24} - 1)(11^{24} + 1) \equiv 0 \bmod 323$,
which is equivalent to $323 \mid (11^{24} - 1)(11^{24} + 1)$

we know 323 doesn't divide $11^{24} - 1$, or else we'd have
$11^{24} \equiv 1 \bmod 323$

so at least some of the factors of 323 must also
divide $11^{24} + 1$

@eli.holderness.dev

Shor's order-finding algorithm

given that at least some of the factors of 323
must also divide $11^{24} + 1$

Shor's order-finding algorithm

given that at least some of the factors of 323
must also divide $11^{24} + 1$

calculate $gcd(323, 11^{24} + 1) = 17$,
which is computationally efficient on classical computers

Shor's order-finding algorithm

given that at least some of the factors of 323 must also divide $11^{24} + 1$

calculate $gcd(323, 11^{24} + 1) = 17$, which is computationally efficient on classical computers

find that $17 \mid 323$ and $323 = 17 * 19$.

Shor's order-finding algorithm

given that at least some of the factors of 323
must also divide $11^{24} + 1$

calculate $gcd(323, 11^{24} + 1) = 17$,
which is computationally efficient on classical computers

find that $17 \mid 323$ and $323 = 17 * 19$.

this breaks RSA!

the Integer Factorisation problem

if pq = N with p & q prime, find p and q given only N

the Discrete Logarithm problem

if g generates a subgroup of a finite field F, and y is another member of F, find x such that $g^x = y$

the Elliptic Curve Discrete Logarithm problem

if G generates a subgroup of an elliptic curve over a field F, and P is another member of that elliptic curve, find k such that $P = kG$

~~the Integer Factorisation problem~~

~~if pq = N with p & q prime, find p and q given only N~~

the Discrete Logarithm problem

if g generates a subgroup of a finite field F, and y is another member of F, find x such that $g^x = y$

the Elliptic Curve Discrete Logarithm problem

if G generates a subgroup of an elliptic curve over a field F, and P is another member of that elliptic curve, find k such that $P = kG$

~~the Integer Factorisation problem~~

~~if pq = N with p & q prime, find p and q given only N~~

~~the Discrete Logarithm problem~~

~~if g generates a subgroup of a finite field F, and y is another member of F, find x such that $g^x = y$~~

~~the Elliptic Curve Discrete Logarithm problem~~

~~if G generates a subgroup of an elliptic curve over a field F, and P is another member of that elliptic curve, find k such that $P = kG$~~

~~the Integer Factorisation problem~~

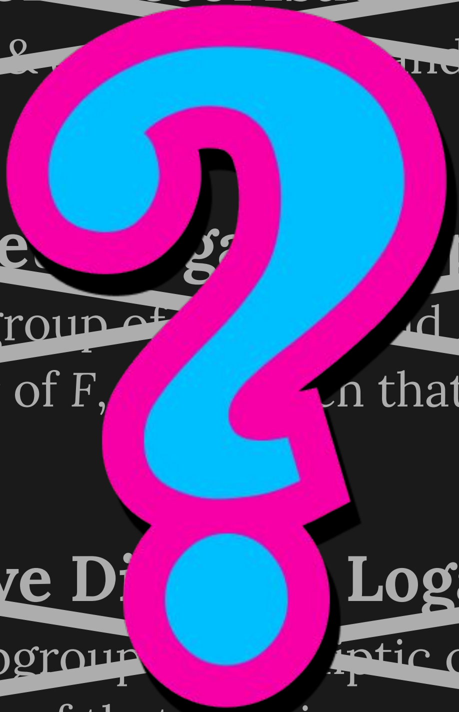
~~if pq = N with p & [illegible] and q given only N~~

~~the Discre[illegible]ga[illegible] problem~~

~~if g generates a subgroup o[illegible]d F, and y is another member of F, [illegible]n that $g^x = y$~~

~~the Elliptic Curve Di[illegible] Logarithm problem~~

~~if G generates a subgrou[illegible]iptic curve over a field F, and P is another member of that elliptic curve, find k such that $P = kG$~~



5

Post-quantum Cryptography

the isogeny-finding problem

given two elliptic curves between which we know there exists an isogeny, find the mapping that describes it

the isogeny-finding problem

given two elliptic curves between which we know there exists an isogeny, find the mapping that describes it

SIKE and SIDH, which are considered insecure

the isogeny-finding problem

given two elliptic curves between which we know there exists an isogeny, find the mapping that describes it

SIKE and SIDH, which are considered insecure

CSIDH

Quantum Security Analysis of CSIDH and Ordinary Isogeny-based Schemes

Xavier Bonnetain[1,2] and André Schrottenloher[2]

[1] Sorbonne Université, Collège Doctoral, F-75005 Paris, France
[2] Inria, France

Abstract. CSIDH is a recent proposal by Castryck, Lange, Martindale, Panny and Renes for post-quantum non-interactive key-exchange. It is similar in design to a scheme by Couveignes, Rostovtsev and Stolbunov

https://who.rocq.inria.fr/Xavier.Bonnetain/pdfs/csidh-attack.pdf

Quantum Security Analysis of CSIDH and Ordinary Isogeny-based Schemes

Xavier Bonnetain[1,2] and André Schrottenloher[2]

[1] Sorbonne Université, Collège Doctoral, F-75005 Paris, France
[2] Inria, France

Abstract. CSIDH is a recent proposal by Castryck, Lange, Martindale, Panny and Renes for post-quantum non-interactive key-exchange. It is similar in design to a scheme by Couveignes, Rostovtsev and Stolbunov

7 Conclusion

We presented a comprehensive quantum security assessment of CSIDH. In particular, when compared to the cost of a classical key-exchange, we showed that the parameters set in [6] actually seem to provide only around half of the expected security, as summarized in Table 7.

https://who.rocq.inria.fr/Xavier.Bonnetain/pdfs/csidh-attack.pdf

the isogeny-finding problem

given two elliptic curves between which we know there exists an isogeny, find the mapping that describes it

SIKE and SIDH, which are considered insecure

CSIDH, which should also be considered insecure

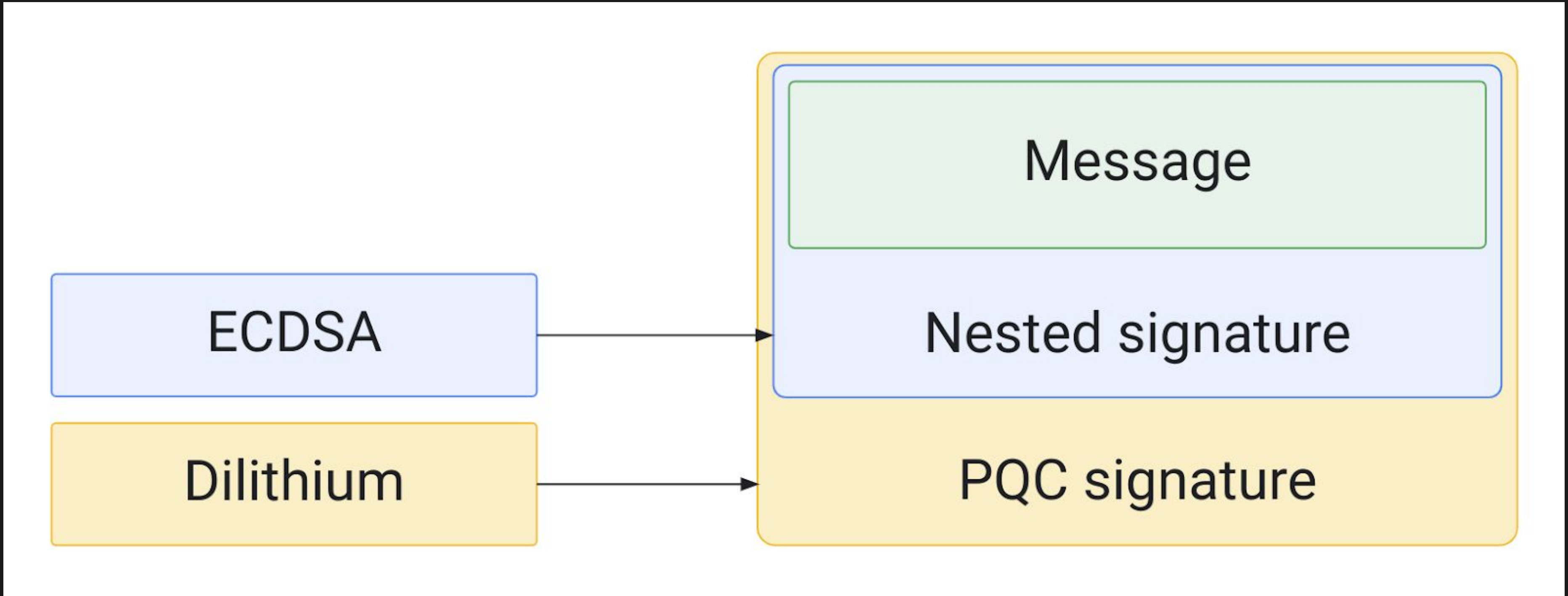
the Learning With Errors problem

introducing noise to encodings and using probability to decode

the Learning With Errors problem

introducing noise to encodings and using probability to decode

CRYSTALS-Kyber (key encapsulation) and
CRYSTALS-Dilithium (signatures)



https://security.googleblog.com/2023/08/toward-quantum-resilient-security-keys.html

@eli.holderness.dev

In Chrome, you can now enable 'X25519Kyber768' for key exchange during TLS

32 bits generated by X25519

32 bits generated by Kyber768

https://openquantumsafe.org/

what I hope to see

what I hope to see

more diverse quantum-resilient cryptosystems

what I hope to see

more diverse quantum-resilient cryptosystems

quantum-resilient hardware tokens

what I hope to see

more diverse quantum-resilient cryptosystems

quantum-resilient hardware tokens

wider accessibility & rollout

wrapping up

how we got here

how we got here

RSA & ECDSA

how we got here

RSA & ECDSA

...and how quantum breaks them

how we got here

RSA & ECDSA

…and how quantum breaks them

what’s next

Asymmetric Cryptography: A Deep Dive

Eli Holderness
@eli@hachyderm.io
they/them/theirs

sources: history

https://www.redhat.com/en/blog/brief-history-cryptography

sources: RSA + group theory

https://ee.stanford.edu/~hellman/publications/24.pdf

https://weakdh.org/imperfect-forward-secrecy-ccs15.pdf

https://en.wikipedia.org/wiki/Padding_(cryptography)

sources: ECC

https://scholar.rose-hulman.edu/cgi/viewcontent.cgi?article=1389&context=rhumj

http://koclab.cs.ucsb.edu/teaching/ecc/eccPapers/Washington-ch04.pdf

http://www.secg.org/sec2-v2.pdf

sources: QC & Shor

https://research.kudelskisecurity.com/2021/08/24/quantum-attack-resource-estimate-using-shors-algorithm-to-break-rsa-vs-dh-dsa-vs-ecc/

https://arxiv.org/pdf/quant-ph/9508027.pdf

https://www.omnicalculator.com/math/power-modulo

sources: PQC

https://security.googleblog.com/2023/08/toward-quantum-resilient-security-keys.html

https://csidh.isogeny.org/

https://sike.org/

https://eprint.iacr.org/2019/725

https://blog.chromium.org/2023/08/protecting-chrome-traffic-with-hybrid.html

https://www.ietf.org/archive/id/draft-tls-westerbaan-xyber768d00-02.html

https://openquantumsafe.org/

https://eprint.iacr.org/2022/1225.pdf