

Theory of Blockchain



Session 4:

Asymmetric Cryptography - Part 2

Module 1 – Introduction to Elliptic
Curves + ECDSA

Elliptic Curve Cryptography

- **Elliptic curve cryptography (ECC)** is an approach to public-key cryptography based on the algebraic structure of elliptic curves over finite fields. ECC requires smaller keys compared to non-EC cryptography (i.e. based on plain Galois fields) to provide equivalent security.
- Elliptic curves can be used for digital signatures, pseudo-random number generation, etc.

The Elliptic Curve Digital Signature Algorithm (ECDSA) is used in the majority of cryptocurrencies, including Bitcoin and Ethereum.

Elliptic Curve Cryptography

For almost every **public-key cryptosystem**, there is an alternative based on ECC. So ECC is another domain for implementation of public-key schemes.



- The ECC schemes are usually faster or more secure with the same key size. Consequently, ECC is particularly appropriate for resource-limited embedded devices (e.g. smart cards).
- It has its own mathematics: ADDITION, MULTIPLICATION, ...

What's a Group?

- » Set of elements
- » Operation: $*$
- » Closed under $*$
- » Inverses
- » Identity: e

Example:

Integer numbers with $*$ = Addition

- $1+5=6$ (still an integer \rightarrow closed)
- $5+(-5)=0$ (there is an inverse for every integer)
- The identity element is 0 (i.e. $e=0$)
- $1+(5+4)=(1+5)+4$ (It's associative too)
- $1+5=5+1$ (It's Abelian/Commutative too)

Elliptic curve points also form a group with the $+$ operation, as we will see.

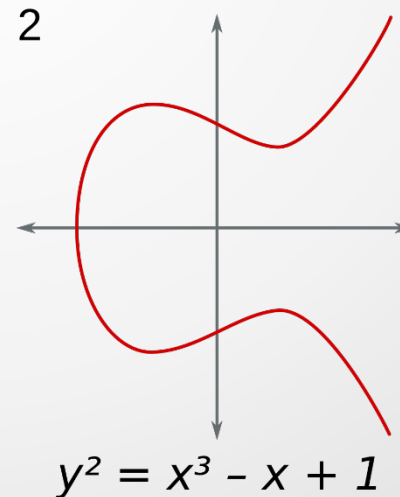
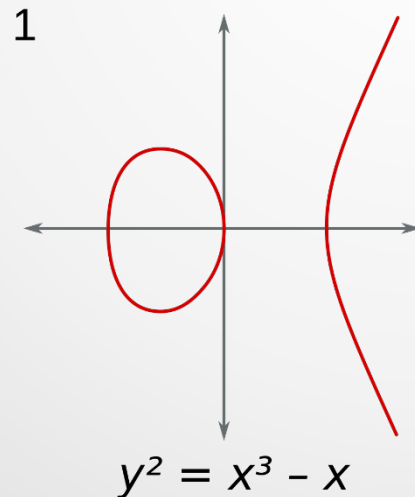
Highly suggested video: <https://www.youtube.com/watch?v=QudbrUcVPxk>

What's an Elliptic Curve (EC)

EC is the set of points described by the equation:

$$y^2 = x^3 + ax + b$$

Where $4a^3 + 27b^2 \neq 0$. (this is required to exclude singular curves).



Non-singularity

EC must be non-singular. Non-singularity means that the curve has no cusps (sharp tip), self-intersections, or isolated points.



On the left, a curve with a cusp ($y^2=x^3$). On the right, a curve with a self-intersection ($y^2=x^3-3x+2$). None of them is a valid EC for cryptography.

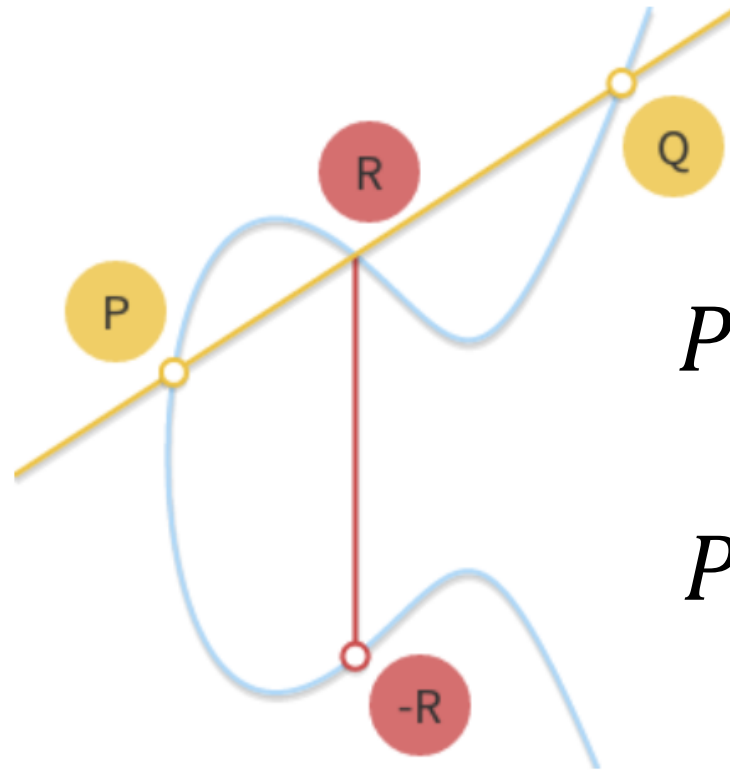
A bit of mathematics:

We can define a group over elliptic curves. Specifically:

- The elements of the group are the points of an elliptic curve;
- The identity element is the point at infinity O ; (e.g. $y = \infty$)
- The inverse of a point $P = (x, y)$ is the one symmetric about the x-axis, that is $-P = (x, -y)$;
- Addition is given by the following rule: given three aligned, non-zero points P , Q and R , their sum is $P+Q+R=O$.

These two minuses are different !

“+” operation on ECs



$$P + Q + R \triangleq 0$$

→

$$P + Q \triangleq -R$$

Draw the line through P and Q . The line intersects a third point R . The point symmetric to it, $-R$, is the result of $P + Q$.

Similarly we can imagine where 0 lies from $P+0=P \rightarrow y=\inf$

Addition on ECs:

- If P and Q are distinct, the line through them has the **slope**:

$$m = \frac{y_P - y_Q}{x_P - x_Q}$$

- If P=Q, the tangent line will have the **slope**:

$$m = \frac{3x_P^2 + a}{2y_P}$$

- The intersection of this line with the curve is $R = (x_R, y_R)$

$$x_R = m^2 - x_P - x_Q$$

$$y_R = y_P + m(x_R - x_P)$$

$$P + Q = -R \rightarrow (x_P, y_P) + (x_Q, y_Q) = (x_R, -y_R)$$

Example

$$y^2 = x^3 - 7x + 10$$

$$P=(1,2) \quad Q=(3,4) \quad : \quad P+Q=-R=?$$

$$m = \frac{y_P - y_Q}{x_P - x_Q} = \frac{2-4}{1-3} = 1$$

$$x_R = m^2 - x_P - x_Q = 1^2 - 1 - 3 = -3$$

$$y_R = y_P + m(x_R - x_P) = 2 + 1 \cdot (-3 - 1) = -2$$

$$\Rightarrow P + Q = -R = (-3, 2)$$

“×” operation on ECs

$$nP = \underbrace{P + P + \dots + P}_{n \text{ times}}$$

n is a natural number.

Written in that form, it may seem that computing nP requires n additions. If n has k binary digits, then this algorithm would be $O(2^k)$, which is not really good. But there exist faster algorithms. One of them is the **double and add** algorithm $\rightarrow O(\log n)$.

$O(2^k) = O(n)$

Double and Add Algorithm

It is of $O(\log n)$
or $O(k)$

Take $n = 151$. Its binary representation is 10010111_2 . This binary representation can be turned into a sum of powers of two:

$$\begin{aligned} 151 &= 1 \cdot 2^7 + 0 \cdot 2^6 + 0 \cdot 2^5 + 1 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 \\ &= 2^7 + 2^4 + 2^2 + 2^1 + 2^0 \end{aligned}$$

(We have taken each binary digit of n and multiplied it by a power of two.)

In view of this, we can write:

$$151 \cdot P = 2^7 P + 2^4 P + 2^2 P + 2^1 P + 2^0 P$$

What the double and add algorithm tells us to do is:

- Take P .
- *Double* it, so that we get $2P$.
- *Add* $2P$ to P (in order to get the result of $2^1 P + 2^0 P$).
- *Double* $2P$, so that we get $2^2 P$.
- *Add* it to our result (so that we get $2^2 P + 2^1 P + 2^0 P$).
- *Double* $2^2 P$ to get $2^3 P$.
- Don't perform any addition involving $2^3 P$.
- *Double* $2^3 P$ to get $2^4 P$.
- *Add* it to our result (so that we get $2^4 P + 2^2 P + 2^1 P + 2^0 P$).
- ...

Elliptic curves over \mathcal{F}_p

- Usually we restrict elliptic curves over finite fields (like \mathcal{F}_p). Originally the set of points on EC are:

$$\{(x, y) \in \mathbb{R}^2 \mid y^2 = x^3 + ax + b, \\ 4a^3 + 27b^2 \neq 0\} \cup \{0\}$$

which is restricted to \mathcal{F}_p as:

$$\{(x, y) \in (\mathbb{F}_p)^2 \mid y^2 \equiv x^3 + ax + b \pmod{p}, \\ 4a^3 + 27b^2 \not\equiv 0 \pmod{p}\} \cup \{0\}$$

where 0 is the identity element of + operation, and a and b are two integers in \mathcal{F}_p .

What is a Field?

Set F with 2 operations: $+$ \cdot
 $\langle F, + \rangle$ is a commutative group
 $\langle F^\times, \cdot \rangle$ is a commutative group
 $a \cdot (b + c) = a \cdot b + a \cdot c$
 $(b + c) \cdot a = b \cdot a + c \cdot a$

Example:

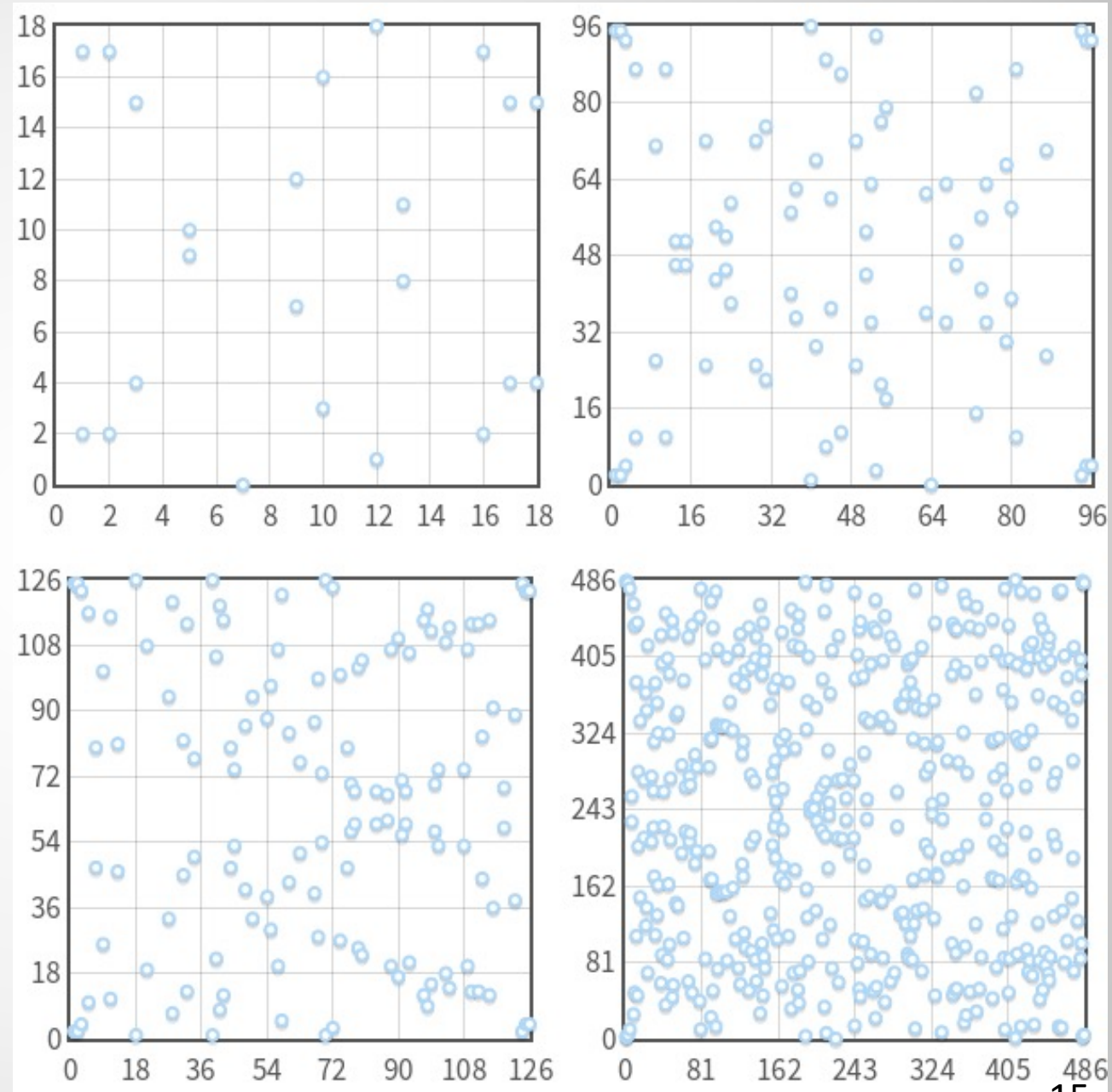
Integer numbers with $+$ = Addition
and \cdot = Multiplication

Elliptic curves in \mathcal{F}_p

The set of EC points in \mathcal{F}_p
with $p = 19, 97, 127, 487$

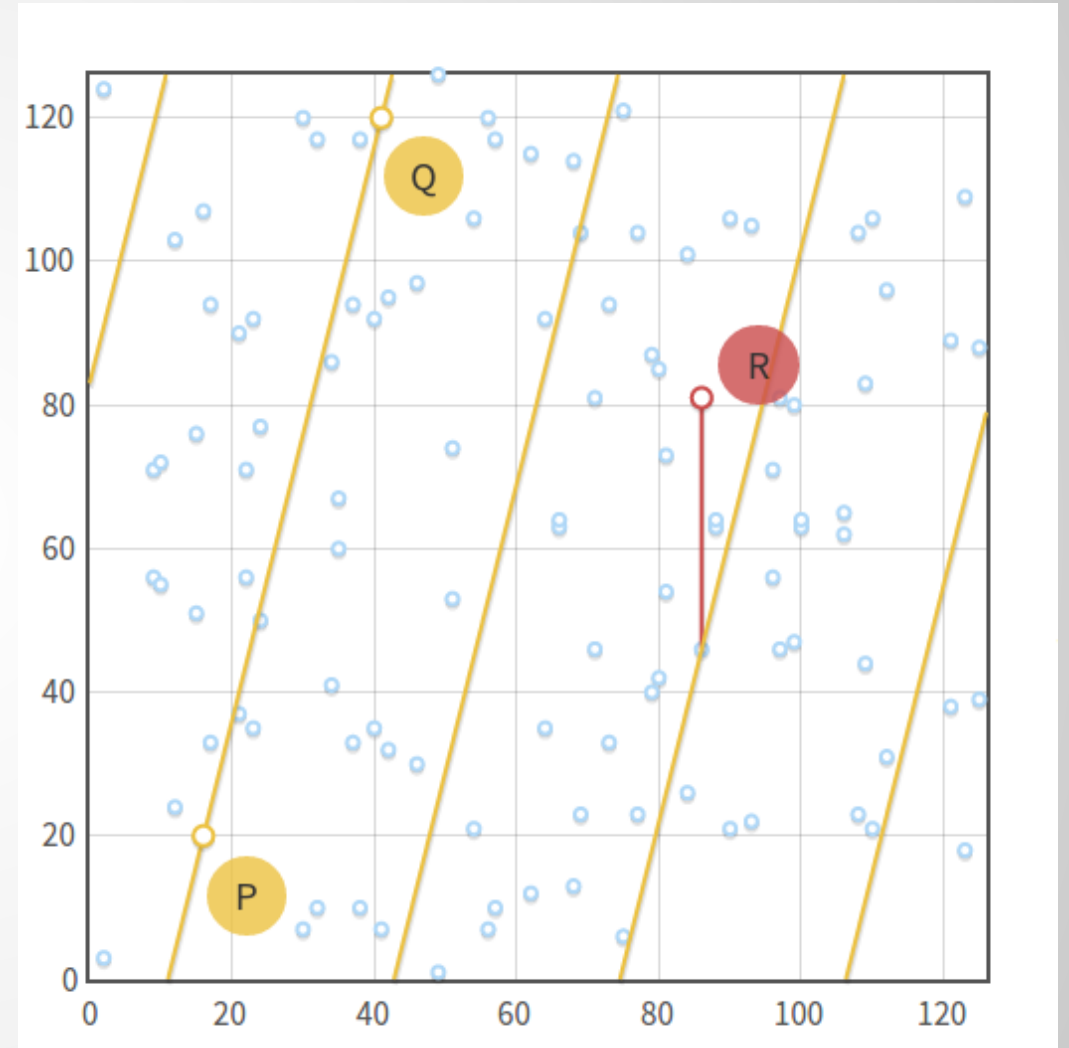
The EC equation is:

$$y^2 = x^3 - 7x + 10$$

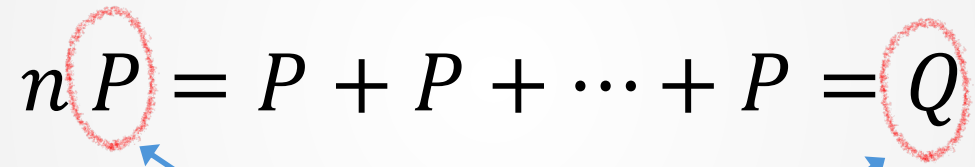


Elliptic curves in \mathcal{F}_p

Clearly, we need to change our definition of addition a bit in order to make it work in \mathcal{F}_p . With reals, we said that the sum of three aligned points was zero. We can keep this definition. Now, we can say that three points are aligned if there's a line that connects all of them. Of course, lines in \mathcal{F}_p are not the same as lines in \mathbb{R} . We can say, informally, that a line in \mathcal{F}_p is the set of points that satisfy the equation $ax + by + c \equiv 0 \pmod{p}$ (this is the standard line equation, with the addition of "mod p ").

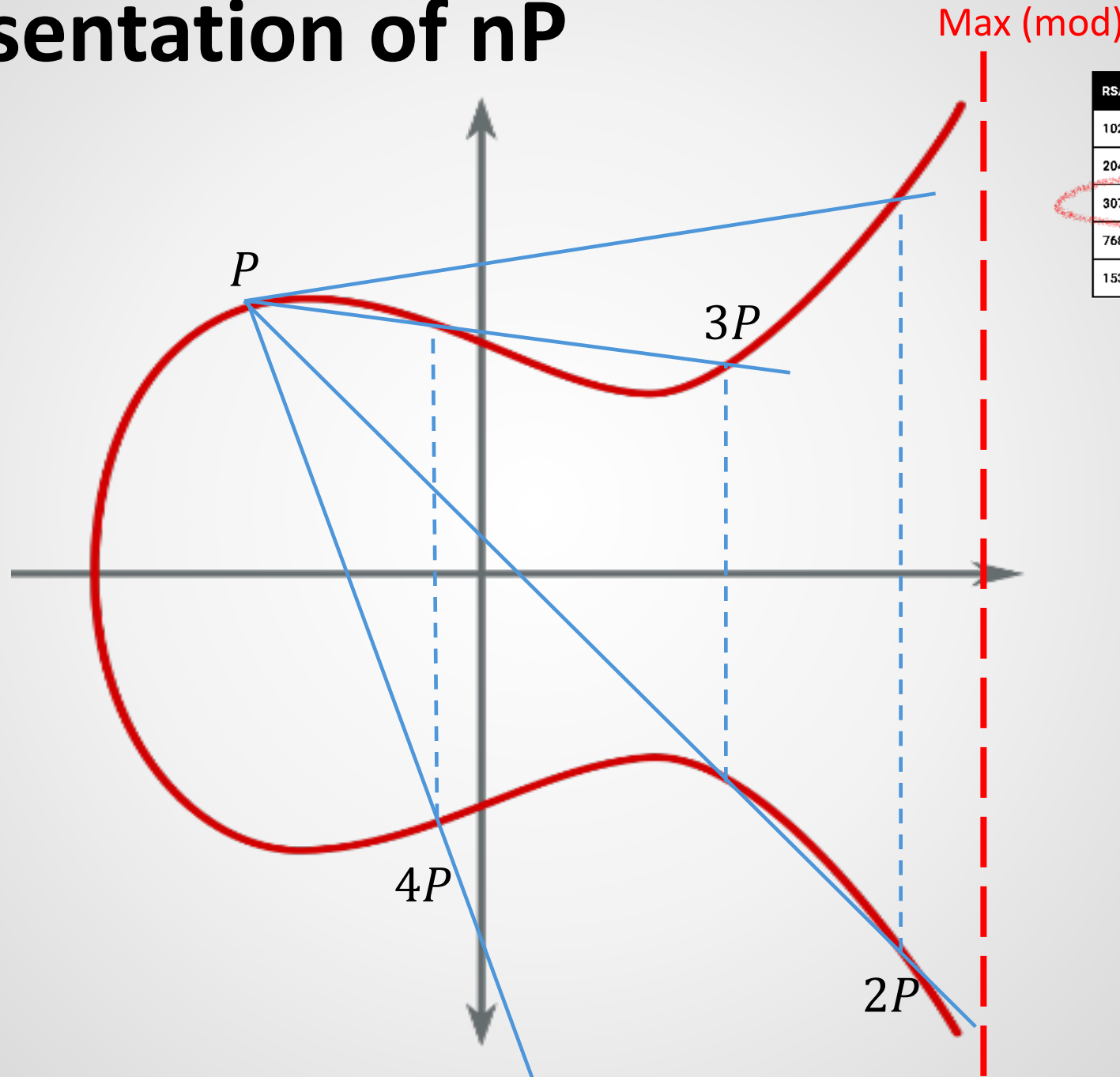


The Most Important Feature in EC Cryptography

$$nP = P + P + \dots + P = Q$$


Even if both Q and P are given, it's really hard to find n. It requires brute force. If \mathcal{F}_p becomes big, this brute force becomes impossible.

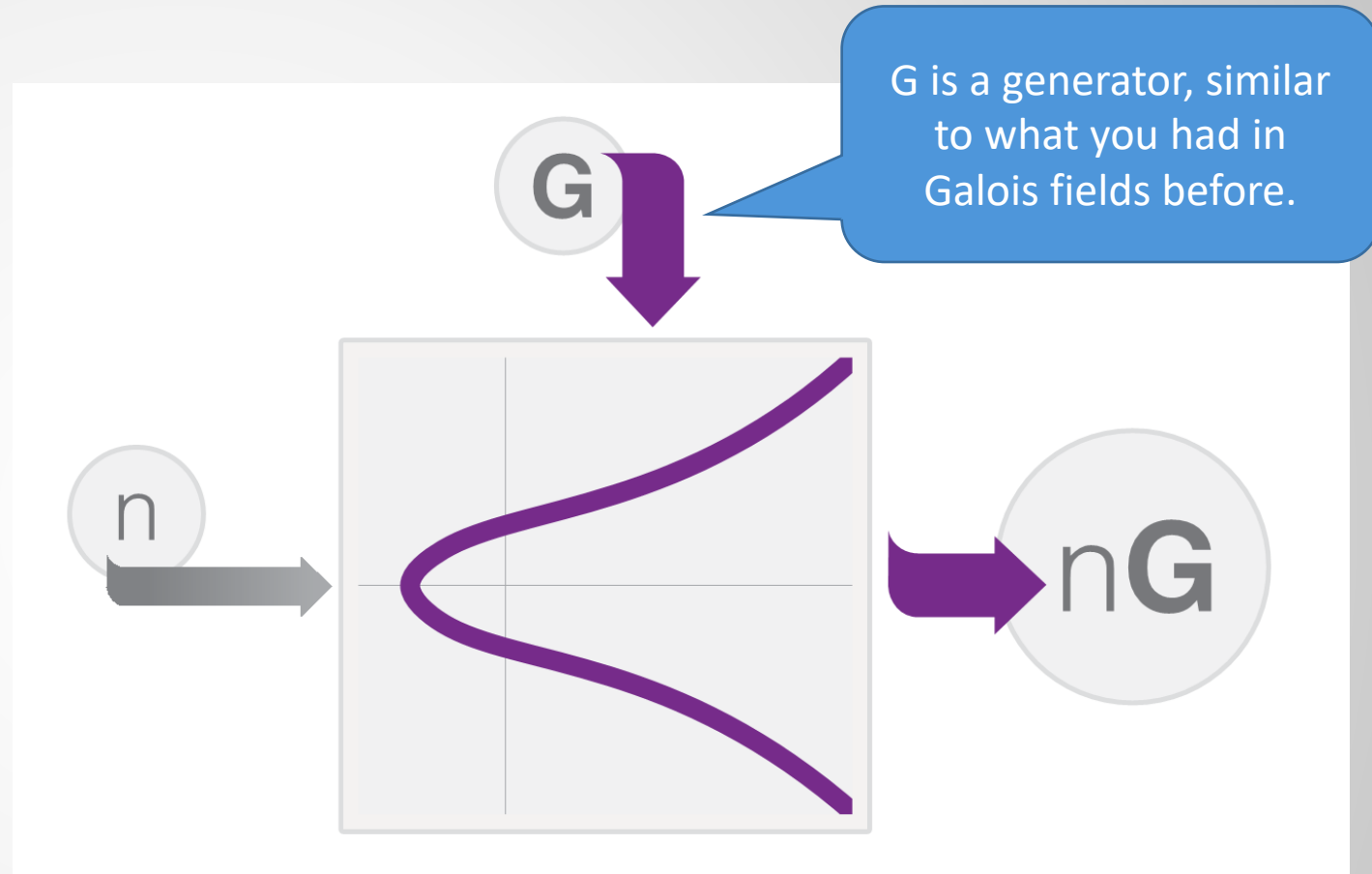
Visual Presentation of nP



RSA Size (bits)	Elliptic Curve Key Size (bits)
1024	160
2048	224
3072	256
7680	384
15360	521

Elliptic Curves Discrete Logarithm (one-way F)

- Given the point **G** and the EC, calculating **nG** is easy.
- But if one gives you the point **Q=nG**, it's hard to find **n**, even if you have **G**.

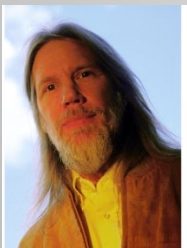
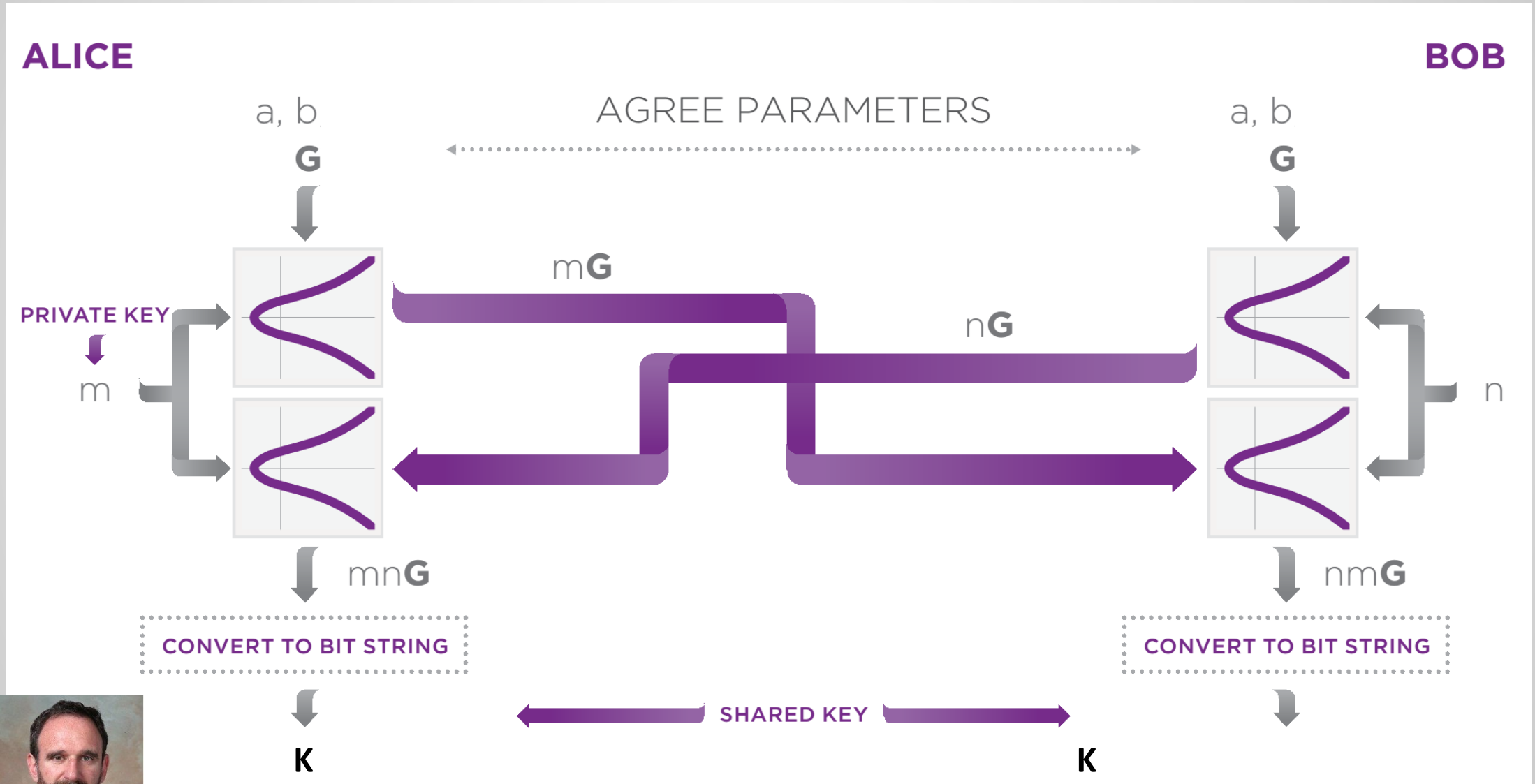


GF		ECC
<hr/>		
g^n	\approx	nG

→
Computationally **Easy**

←
Computationally **Hard**

Elliptic Curve Diffie-Hellman Key Agreement (ECDH)



Elliptic Curve Digital Signature Algorithm (ECDSA)

Let us say Alice has a **private key (integer) of k** , and a **public key of $Q = kP$** (where P is a point on EC).

Signature Generation:

1. Find the hash of the message: $h = H(m)$
2. (Per message) generate a random integer $1 \leq r \leq n - 1$
3. Compute $V = rP = (x, y)$
4. Calculate $s = r^{-1}(h + xk) \bmod n$
5. The signature is the (x, s) pair.

Elliptic Curve Digital Signature Algorithm

Signature Verification

1. Find the hash of the received message: $h' = H(m')$
2. Compute $u_1 = s^{-1}h' \bmod n$ and $u_2 = s^{-1}x \bmod n$
3. Compute $V' = (x_v, y_v) = u_1P + u_2Q$
4. The signature is valid if $x_v == x \bmod p$

Why?

$$V' = u_1P + u_2Q = s^{-1}hP + s^{-1}xQ = s^{-1}(h + xk)P$$

We had $s = r^{-1}(h + xk) \bmod n \Rightarrow r = s^{-1}(h + xk) \bmod n$, therefore:

$$V' = rP = V \Rightarrow x_v = x$$

We know the sender had the private key because he could factor Q into kP .

What Comes Next ...

- We learned about elliptic curves.
- We learned how the elliptic curve digital signature algorithm works. It is used in almost every famous cryptocurrency.
- In the next video, we explain how public keys are managed and distributed.

See you in the next video ...