Elliptic Curve Cryptosystems

→ Elliptic Curve Cryptography (ECC) works well with RSA and DLP where ECC is based on DLP → enables same security of RSA or DLP requiring smaller operands

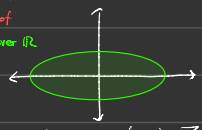
→ Since ECC is based on DLP → we need to find a cyclic group to build the cryptosystem off of

By "curves" ⇒ set of points (x,y) → sol^{n} of polynomial equations

Ex: Say we take $x^2+y^2=r^2$ over $\mathbb{R}$:

Ex: Say we do $ax^2+by^2=c^2$ over $\mathbb{R}$

we get an ellipse



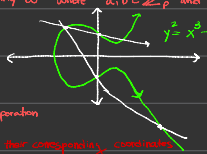
→ Elliptic Curve: Special type of polynomial equation over finite fields like Prime Galois Fields $GF(p)$

→ Formal Def^{n} of Elliptic Curves: Elliptic Curve over $\mathbb{Z}_p$ where $p>3$ and is prime, is set of all pairs $(x,y)\in\mathbb{Z}_p$ s.t. $y^2\equiv x^3+a\cdot x+b \bmod p$ together with an imaginary point of infinity ∞ where $a,b\in\mathbb{Z}_p$ and $4a^3+27b^2\neq 0 \bmod p$

This doesn't look like a curve over $\mathbb{Z}_p$ but we could draw it over $\mathbb{R}$

requires curve is nonsingular → no self-intersections or vertices → $-16(4a^3+27b^2)>0$

Ex:

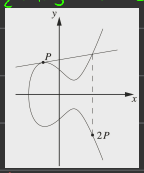


→ Properties of Elliptic Curves

1. Symmetric about x-axis.

$\forall x_i\ (\ y_i=\pm\sqrt{x_i^3+ax_i+b}\ )$ — solve when $y=0$

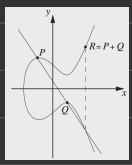
2. There are $1\text{ or }3$ intersections with the x-axis



→ To find curve with large cyclic group → Find a group ⇒ define operation

→ Group Operations: Say points are elements representing their corresponding coordinates

Addition: $P+Q=R \equiv (x_1,y_1)+(x_2,y_2)=(x_3,y_3)$



Tangent-&-Chord:

$R=P+Q$ where $P\neq Q$: Draw line through P and Q and obtain 3rd PoI between Elliptic Curve and Line. Mirror PoI in x-axis → R

$R=P+Q$ where $P=Q$: Draw tangent through P and obtain 2nd PoI between Elliptic Curve & Line. Mirror PoI in x-axis → R

Group requirements

Cowpert third pt: If 2 already given while only using $+,-,\times,\div$ ⇒ set of points fulfill closure, associativity, existence of identity, existence of inverse

→ Elliptic Curve Point Addition and Doubling:

$$x_3\equiv s^2-x_1-x_2 \bmod p$$

$$y_3\equiv s(x_1-x_3)-y_1 \bmod p$$

$$s\equiv\begin{cases}\frac{y_2-y_1}{x_2-x_1} \bmod p & \text{if } P\neq Q\\ \frac{3x_1^2+a}{2y_1} \bmod p & \text{if } P=Q\end{cases}$$

slope

→ Existence of Identity: $P+\mathcal{O}=P\ \forall$ Points P → define abstract point at infinity as Identity $\mathcal{O}$ → forced towards $+\infty$ along y-axis or $-\infty$ along y-axis

→ Existence of Inverse: $P+(-P)=\mathcal{O}\longrightarrow -P=(x_P,-y_P)$ over $\mathbb{R}$

$-P=(x_P,p-y_P)$ where $-y_P\equiv p-y_P \bmod p$

→ Theorem: The points on an elliptic curve together with $\mathcal{O}$ form a group with cyclic subgroups → Under certain conditions, all points of the curve form a cyclic group → pick some point P as generator and perform Point Doubling

→ To set up DLP we need to know the order of this group

→ Hasse's Theorem: Given Elliptic curve E mod p the number of points $p+1-2\sqrt{p}\le \#E\le p+1+2\sqrt{p}$ → If we need 2^{256} elem we need prime p of length 256 bits

→ Elliptic Curve Discrete Logarithmic Problem (ECDLP): Given an Elliptic Curve E with Generator P and element T: DLP is finding $d\in\mathbb{Z},\ 1\le d\le \#E$ s.t. $dP=T$ → d = private key, T = point on curve (x_T,y_T)

Similar to RSA

Point Multiplication Algorithm as Sums of Point Additions

Doubling results in left shift of d

Input: $E,P,d=\sum_{i=0}^{t}d_i2^i$ with $d_i\in\{0,1\}$ and $d_t=1$ → Geometric Interpretation of ECDLP: Given starting P, we compute $2P, 3P, \dots, dP=T$ → To break this cryptosystem, attacker needs to figure out how often we jumped on curve → d

Output: $T=dP$

$T=P$

For $i=t-1$ to 0:

$T=T+T$

If $d_i=1$: $T=T+P$

Return T

→ Diffie-Hellman Key Exchange can be done using ECC → ECDH Protocol:

Domain Parameters:

1. Choose prime p and $E: y^2\equiv x^3+ax+b \bmod p$
2. Choose primitive element $P=(x_P,y_P)$

$\{(x_P,y_P),a,b,P\}$

Alice	Bob
$k_{PrA}=\{2,3,\dots,\#E-1\}$	$k_{PrB}=b\in\{2,3,\dots,\#E-1\}$
$k_{PubA}=aP=A=(x_A,y_A)$	$k_{PubB}=bP=B=(x_B,y_B)$
$\xrightarrow{A}$	
	$\xleftarrow{B}$
$aB=T_{AB}$	$bA=T_{AB}$

→ We use EC because ECDLP has an excellent one-way f^{n} → Oscar has: E,p,P,A,B He wants $T_{AB}=a\cdot b\cdot P$

Proof: Alice computes $a(bP)$ while Bob computes $b(aP)$ Valid via associativity of $\cdot$

If E chosen with care, best ECDLP attack is weaker than best algs for DLP mod p & best Factoring Attack to RSA

$a=\log_P A \qquad b=\log_P B$

Ex: Index Calc. Attack not applicable

T_{AB} → generate session key for AES algorithm

(x_{AB},y_{AB}) are not independent of each other ⇒ either could be used in derivation of session key

In practice, is hashed & used as symmetric key

Shanks' Baby Steps and Pollard's Rho: Since # of steps $\approx\sqrt{\#E}$, group of at least order 2^{256} used → P must at least 256 bits long

At Security only achieved if Ephemeral keys used!!

→ Before ECC used, curve with good properties need to be identified → Core: cyclic group/subgroup formed by points has prime order

→ When implementing ECC ⇒ 4 layers.

→ In practice, ECC is lightweight for IoT devices → high compact ECC engines need 10k gates & run at $1/10^{th}$ of millisecond for point mult

Bottom Layer: Finite Field operations performed (ex: E over $GF(p)$ for prime p)

Next Layer: Group Operations like Point Doubling, Point Addition

Next Layer: Scalar multiplication using Point Multiplication → highly optimized 256-bit ECC on 3-GHz, 64-bit CPU takes $\approx$ 2ms to 10ms → For high performance apps like servers need large throughput ∴ need 40–100 µs

Top Layer: Implements ECDH or ECDSA

→ Computational Complexity is cubic in bit length of prime used → smaller than RSA or DLP even though implementation longer than AES, 3DES

→ smaller but better security