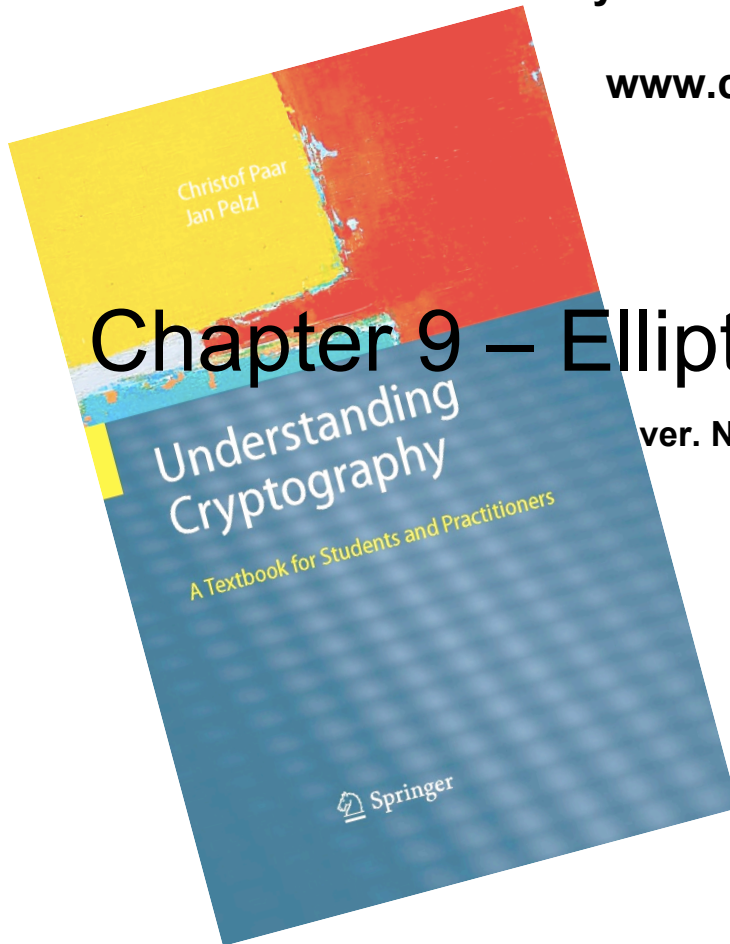


Understanding Cryptography

by Christof Paar and Jan Pelzl

www.crypto-textbook.com



Chapter 9 – Elliptic Curve Cryptography

ver. November 3rd, 2009

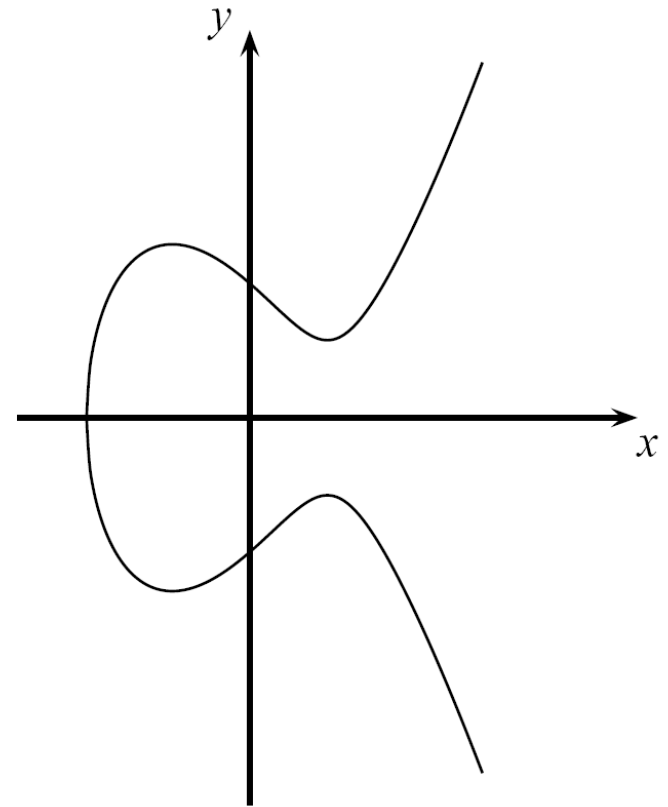
These slides were prepared by Tim Güneysu, Christof Paar and Jan Pelzl

■ Some legal stuff (sorry): Terms of Use

- The slides can be used free of charge. All copyrights for the slides remain with Christof Paar and Jan Pelzl.
- The title of the accompanying book “Understanding Cryptography” by Springer and the author’s names must remain on each slide.
- If the slides are modified, appropriate credits to the book authors and the book title must remain within the slides.
- It is not permitted to reproduce parts or all of the slides in printed form whatsoever without written consent by the authors.

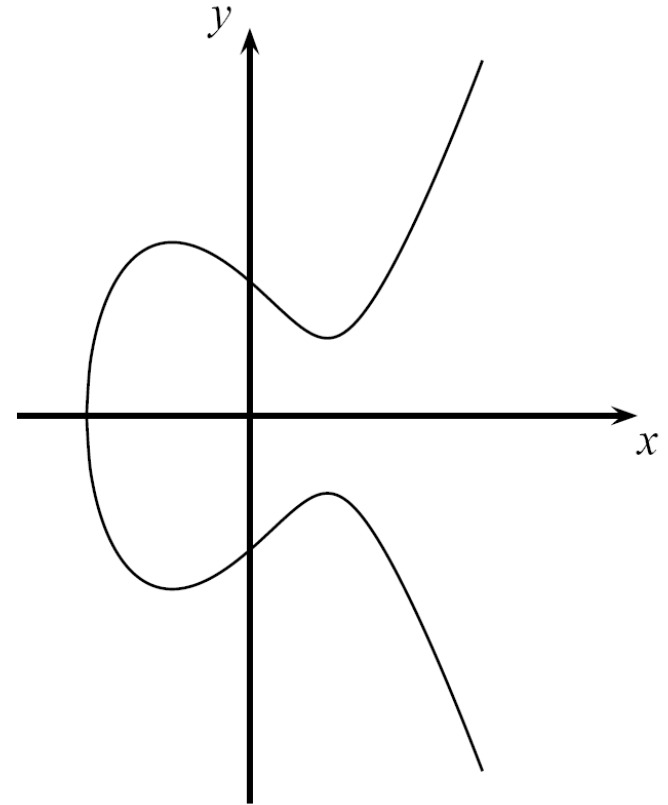
■ Content of this Chapter

- Introduction
- Computations on Elliptic Curves
- The Elliptic Curve Diffie-Hellman Protocol
- Security Aspects
- Implementation in Software and Hardware



■ Content of this Chapter

- **Introduction**
- Computations on Elliptic Curves
- The Elliptic Curve Diffie-Hellman Protocol
- Security Aspects
- Implementation in Software and Hardware



■ Motivation

■ Problem:

Asymmetric schemes like RSA and Elgamal require exponentiations in integer rings and fields with parameters of more than 1000 bits.

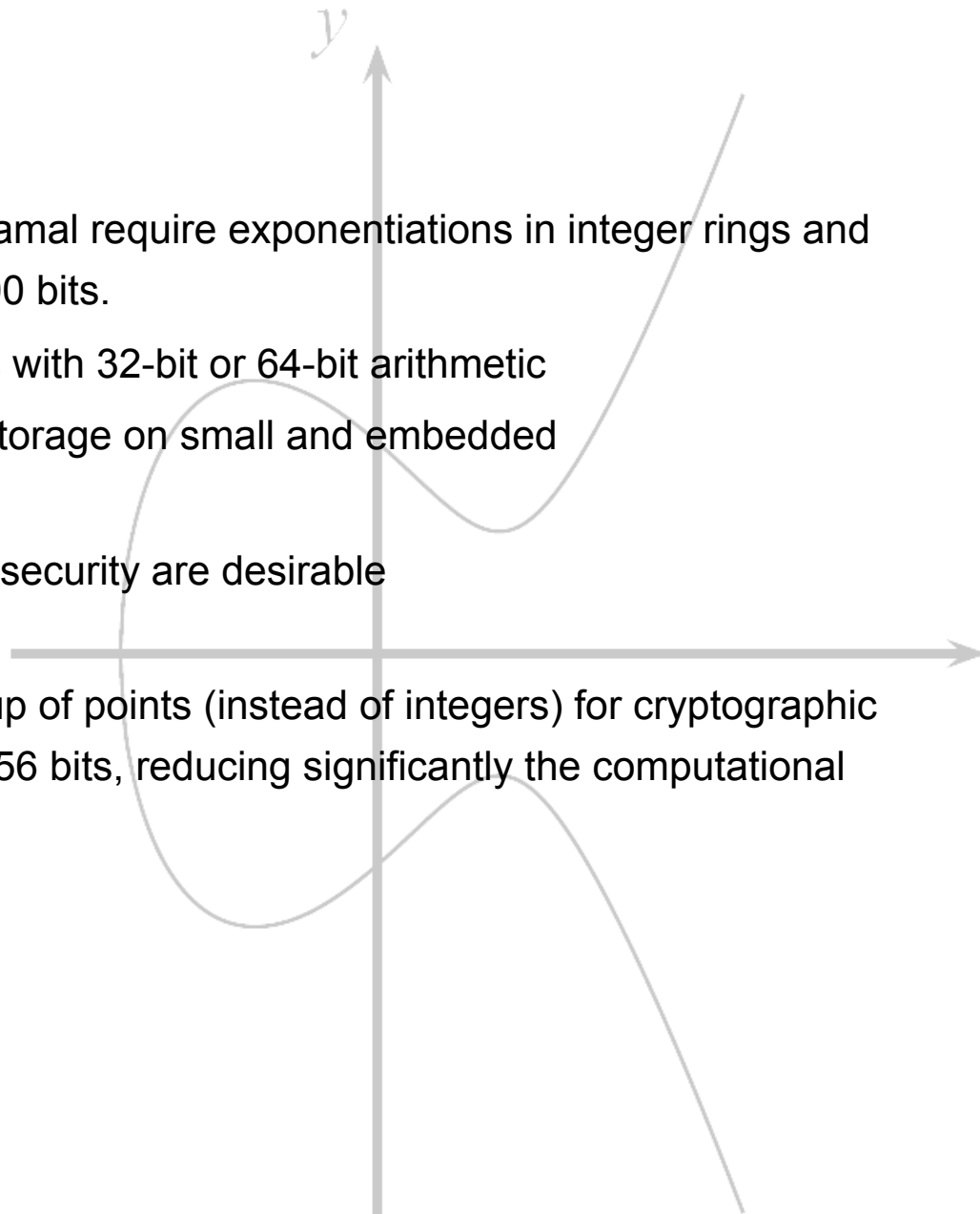
- High computational effort on CPUs with 32-bit or 64-bit arithmetic
- Large parameter sizes critical for storage on small and embedded

■ Motivation:

Smaller field sizes providing equivalent security are desirable

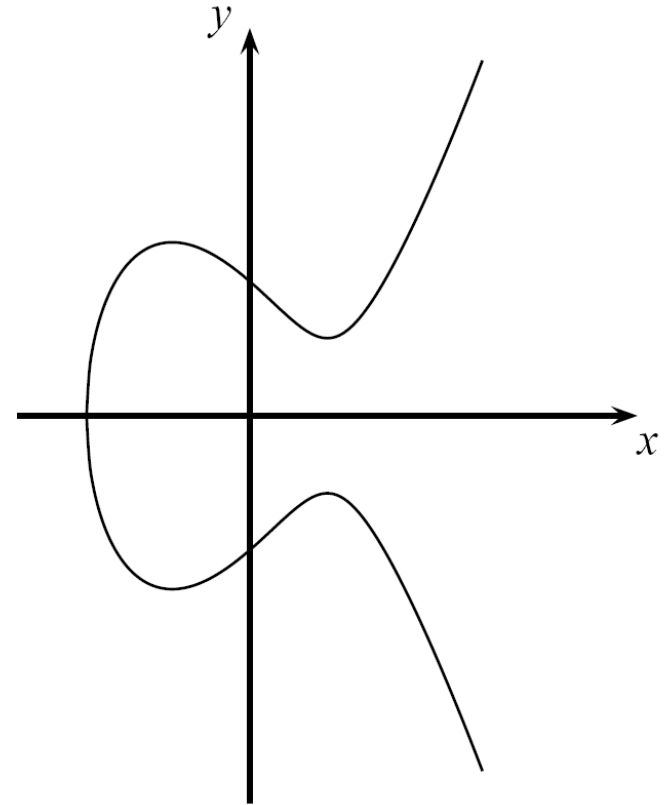
■ Solution:

Elliptic Curve Cryptography uses a group of points (instead of integers) for cryptographic schemes with coefficient sizes of 160-256 bits, reducing significantly the computational effort.



■ Content of this Chapter

- Introduction
- **Computations on Elliptic Curves**
- The Elliptic Curve Diffie-Hellman Protocol
- Security Aspects
- Implementation in Software and Hardware



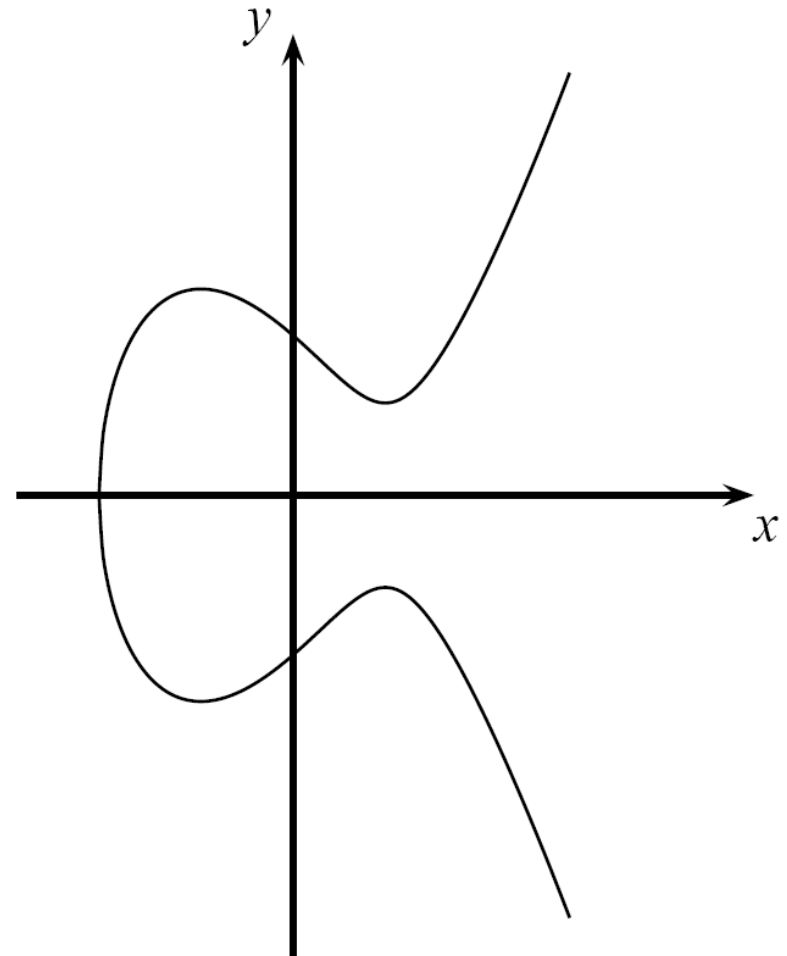
■ Computations on Elliptic Curves

- Elliptic curves are polynomials that define points based on the (simplified) Weierstraß equation:

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

for parameters a, b that specify the exact shape of the curve

- On the real numbers and with parameters $a, b \in \mathbb{R}$, an elliptic curve looks like this →
- Elliptic curves can not just be defined over the real numbers \mathbb{R} but over many other types of finite fields.



Example: $y^2 = x^3 - 3x + 3$ over \mathbb{R}

■ Computations on Elliptic Curves (ctd.)

- In cryptography, we are interested in elliptic curves module a prime p :

Definition: Elliptic Curves over prime fields

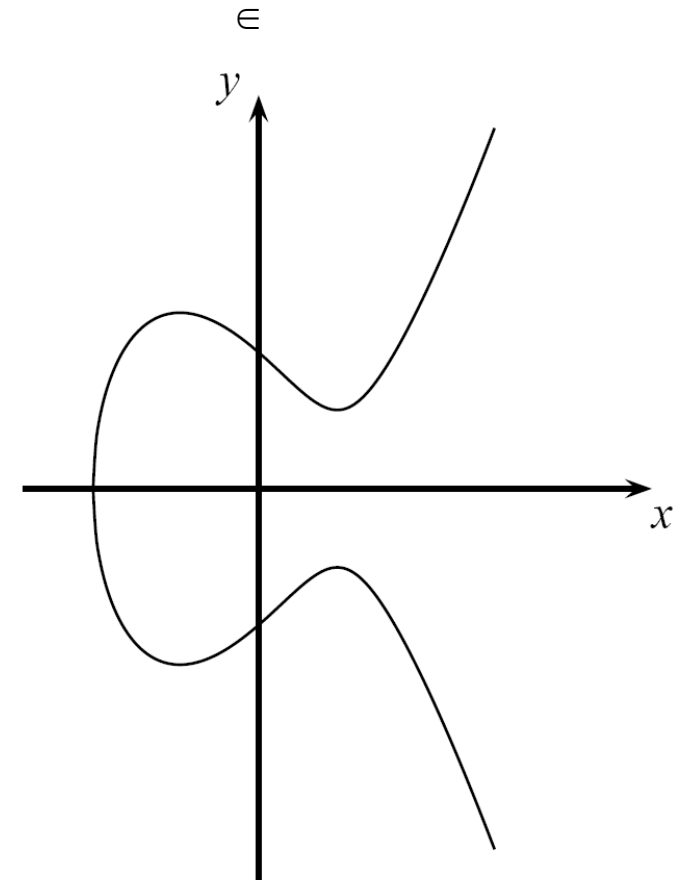
The elliptic curve over Z_p , $p > 3$ is the set of all pairs $(x, y) \in Z_p$ which fulfill

$$y^2 = x^3 + ax + b \bmod p$$

together with an imaginary point of infinity θ , where $a, b \in Z_p$ and the condition

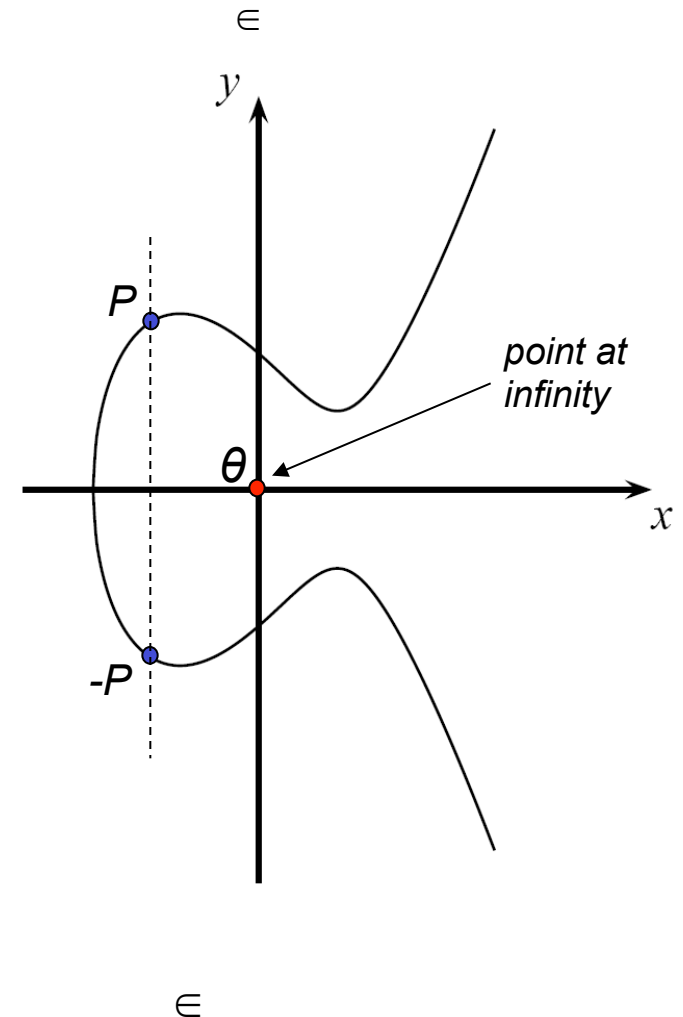
$$4a^3 + 27b^2 \neq 0 \bmod p.$$

- Note that $Z_p = \{0, 1, \dots, p-1\}$ is a set of integers with modulo p arithmetic



■ Computations on Elliptic Curves (ctd.)

- Some special considerations are required to convert elliptic curves into a group of points
 - *In any group, a special element is required to allow for the identity operation, i.e., given $P \in E$: $P + \theta = P = \theta + P$*
 - *This identity point (which is not on the curve) is additionally added to the group definition*
 - *This (infinite) identity point is denoted by θ*
- Elliptic Curves are symmetric along the x-axis
 - Up to two solutions y and $-y$ exist for each quadratic residue x of the elliptic curve
 - For each point $P = (x, y)$, the inverse or negative point is defined as $-P = (x, -y)$



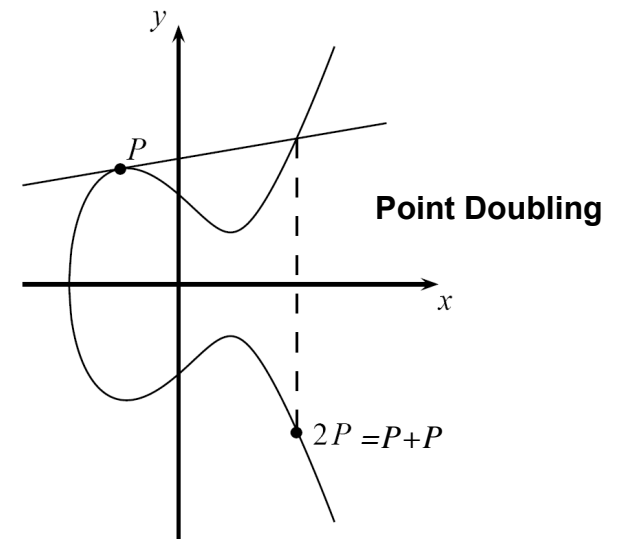
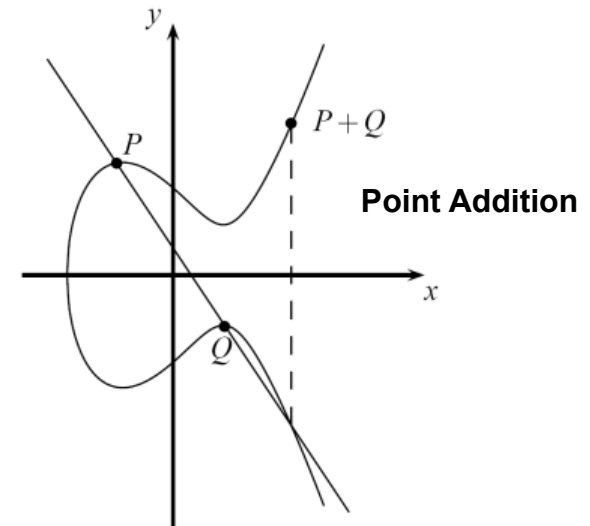
■ Computations on Elliptic Curves (ctd.)

- Generating a *group of points* on elliptic curves based on point addition operation $P+Q = R$, i.e.,
 $(x_P, y_P) + (x_Q, y_Q) = (x_R, y_R)$
- Geometric Interpretation of point addition operation
 - Draw straight line through P and Q ; if $P=Q$ use tangent line instead
 - Mirror third intersection point of drawn line with the elliptic curve along the x -axis
- Elliptic Curve Point Addition and Doubling Formulas

$$x_3 = s_2 - x_1 - x_2 \bmod p \text{ and } y_3 = s(x_1 - x_3) - y_1 \bmod p$$

where

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



■ Computations on Elliptic Curves (ctd.)

- **Example:** Given $E: y^2 = x^3 + 2x + 2 \pmod{17}$ and point $P = (5, 1)$

Goal: Compute $2P = P + P = (5, 1) + (5, 1) = (x_3, y_3)$

$$s = \frac{3x_1^2 + a}{2y_1} = (2 \cdot 1)^{-1}(3 \cdot 5^2 + 2) = 2^{-1} \cdot 9 \equiv 9 \cdot 9 \equiv 13 \pmod{17}$$

$$x_3 = s^2 - x_1 - x_2 = 13^2 - 5 - 5 = 159 \equiv 6 \pmod{17}$$

$$y_3 = s(x_1 - x_3) - y_1 = 13(5 - 6) - 1 = -14 \equiv 3 \pmod{17}$$

Finally $2P = (5, 1) + (5, 1) = (6, 3)$

■ Computations on Elliptic Curves (ctd.)

- The points on an elliptic curve and the point at infinity θ form cyclic subgroups

$$2P = (5,1) + (5,1) = (6,3)$$

$$3P = 2P + P = (10,6)$$

$$4P = (3,1)$$

$$5P = (9,16)$$

$$6P = (16,13)$$

$$7P = (0,6)$$

$$8P = (13,7)$$

$$9P = (7,6)$$

$$10P = (7,11)$$

$$11P = (13,10)$$

$$12P = (0,11)$$

$$13P = (16,4)$$

$$14P = (9,1)$$

$$15P = (3,16)$$

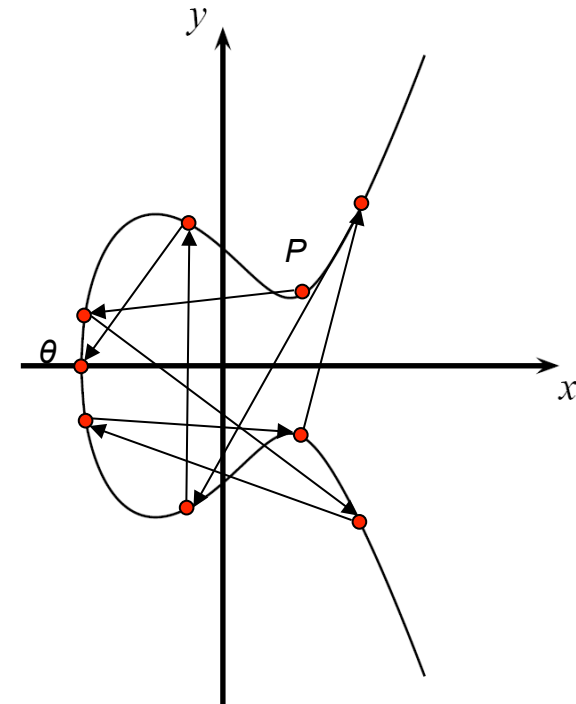
$$16P = (10,11)$$

$$17P = (6,14)$$

$$18P = (5,16)$$

$$19P = \theta$$

This elliptic curve has order $\#E = |E| = 19$ since it contains 19 points in its cyclic group.



■ Number of Points on an Elliptic Curve

- How many points can be on an arbitrary elliptic curve?
 - Consider previous example: $E: y^2 = x^3 + 2x + 2 \pmod{17}$ has 19 points
 - However, determining the point count on elliptic curves in general is hard
- But Hasse's theorem bounds the number of points to a restricted interval

Definition: Hasse's Theorem:

Given an elliptic curve module p , the number of points on the curve is denoted by $\#E$ and is bounded by

$$p+1-2\sqrt{p} \leq \#E \leq p+1+2\sqrt{p}$$

- **Interpretation:** The number of points is „close to“ the prime p
- **Example:** To generate a curve with about 2^{160} points, a prime with a length of about 160 bits is required

■ Elliptic Curve Discrete Logarithm Problem

- Cryptosystems rely on the hardness of the Elliptic Curve Discrete Logarithm Problem (ECDLP)

Definition: Elliptic Curve Discrete Logarithm Problem (ECDLP)

*Given a primitive element P and another element T on an elliptic curve E .
The ECDL problem is finding the integer d , where $1 \leq d \leq \#E$ such that*

$$\underbrace{P + P + \dots + P}_{d \text{ times}} = dP = T.$$

- Cryptosystems are based on the idea that d is large and kept secret and attackers cannot compute it easily
- If d is known, an efficient method to compute the point multiplication dP is required to create a reasonable cryptosystem
 - Known Square-and-Multiply Method can be adapted to Elliptic Curves
 - The method for efficient point multiplication on elliptic curves: Double-and-Add Algorithm

■ Double-and-Add Algorithm for Point Multiplication

■ Double-and-Add Algorithm

Input: Elliptic curve E , an elliptic curve point P and a scalar d with bits d_i

Output: $T = dP$

Initialization:

$T = P$

Algorithm:

FOR $i = t-1$ DOWNT0 0

$T = T + T \bmod n$

 IF $d_i = 1$

$T = T + P \bmod n$

RETURN (T)

Example: $26P = (11010_2)P = (d_4d_3d_2d_1d_0)_2 P$.

Step

#0

$P = 1_2P$

initial setting

#1a

$P+P = 2P = 10_2P$

DOUBLE (bit d_3)

#1b

$2P+P = 3P = 10^2 P + 1_2P = 11_2P$

ADD (bit $d_3=1$)

#2a

$3P+3P = 6P = 2(11_2P) = 110_2P$

DOUBLE (bit d_2)

#2b

$6P+6P = 12P = 2(110_2P) = 1100_2P$

no ADD ($d_2 = 0$)

#3a

$12P+P = 13P = 1100_2P + 1_2P = 1101_2P$

DOUBLE (bit d_1)

#3b

$12P+P = 13P = 1100_2P + 1_2P = 1101_2P$

ADD (bit $d_1=1$)

#4a

$13P+13P = 26P = 2(1101_2P) = 11010_2P$

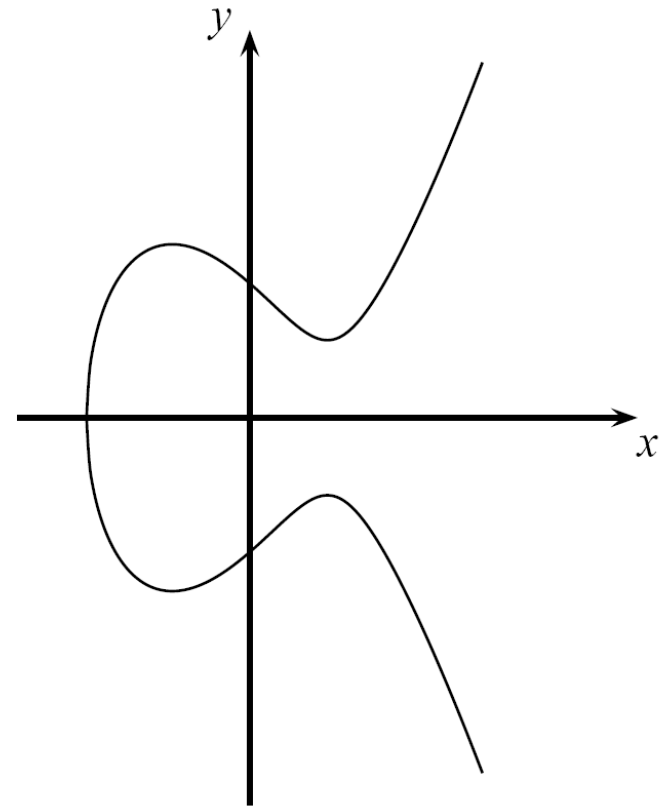
DOUBLE (bit d_0)

#4b

no ADD ($d_0 = 0$)

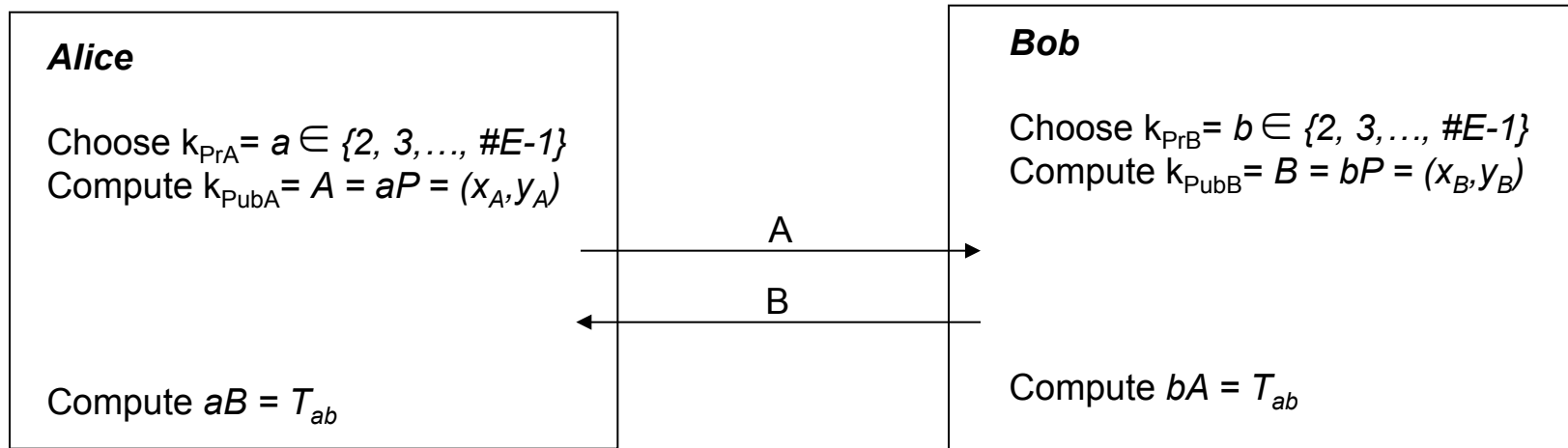
■ Content of this Chapter

- Introduction
- Computations on Elliptic Curves
- **The Elliptic Curve Diffie-Hellman Protocol**
- Security Aspects
- Implementation in Software and Hardware



■ The Elliptic Curve Diffie-Hellman Key Exchange (ECDH)

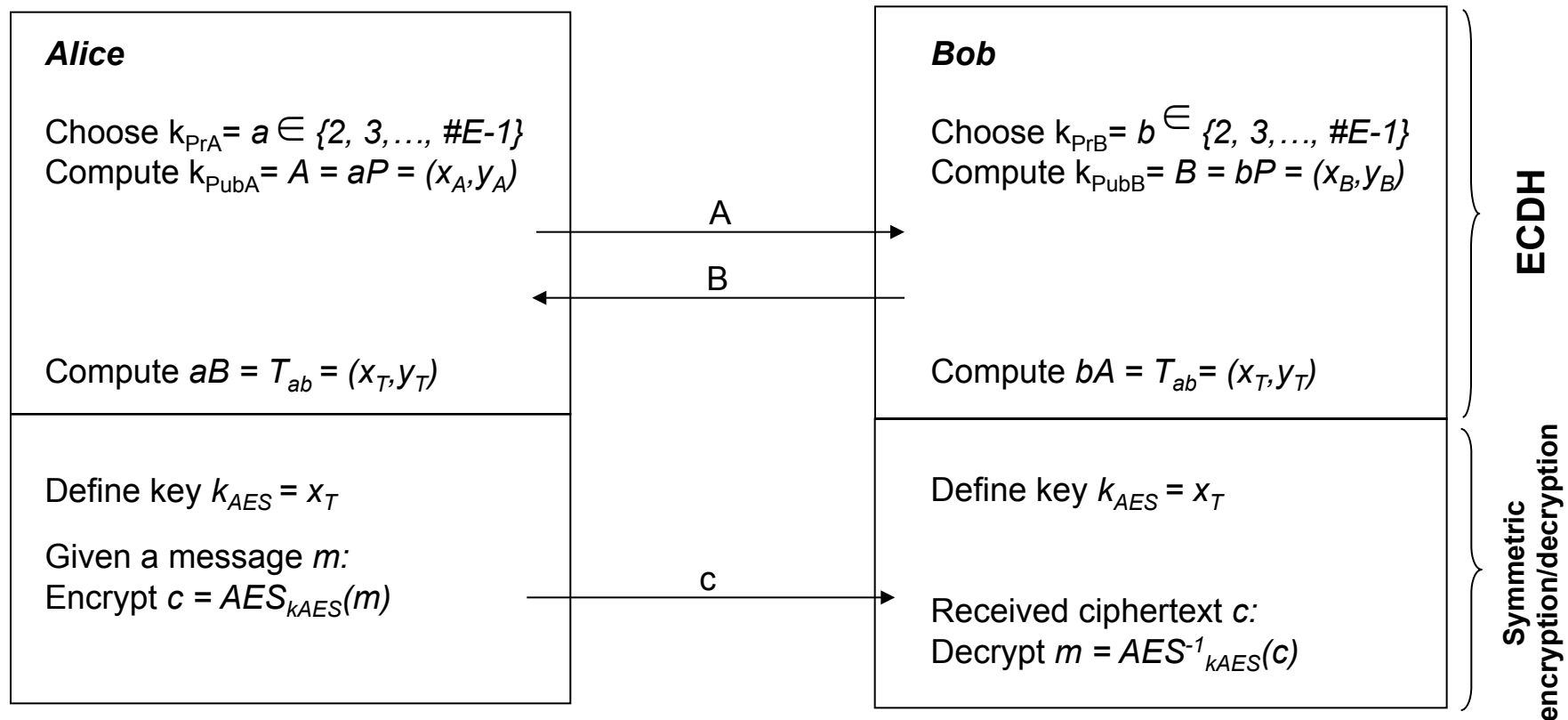
- Given a prime p , a suitable elliptic curve E and a point $P=(x_P, y_P)$
- The Elliptic Curve Diffie-Hellman Key Exchange is defined by the following protocol:



- Joint secret between Alice and Bob: $T_{AB} = (x_{AB}, y_{AB})$
- Proof for correctness:
 - Alice computes $aB = a(bP) = abP$
 - Bob computes $bA = b(aP) = abP$ since group is associative
- One of the coordinates of the point T_{AB} (usually the x-coordinate) can be used as session key (often after applying a hash function)

■ The Elliptic Curve Diffie-Hellman Key Exchange (ECDH) (ctd.)

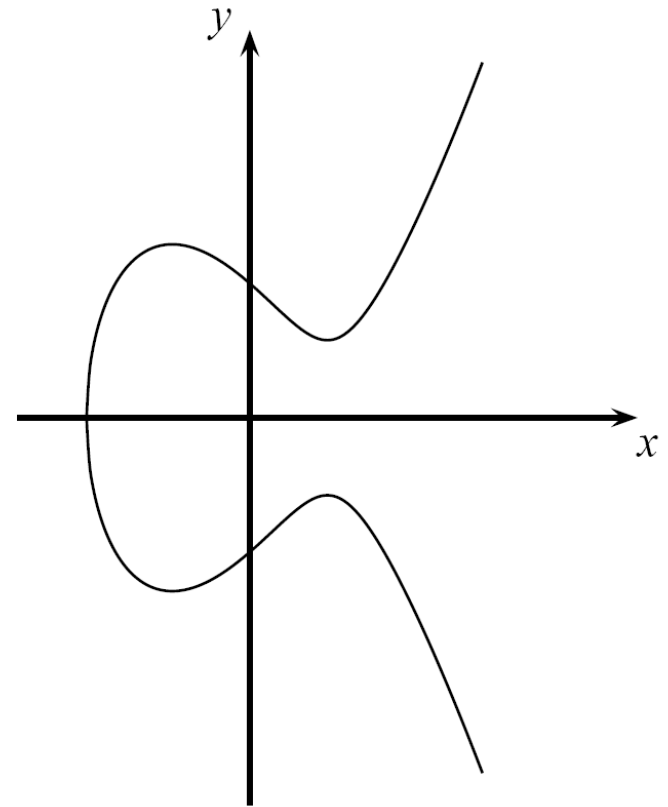
- The ECDH is often used to derive session keys for (symmetric) encryption
- One of the coordinates of the point T_{AB} (usually the x-coordinate) is taken as session key



- In some cases, a hash function (see next chapters) is used to derive the session key

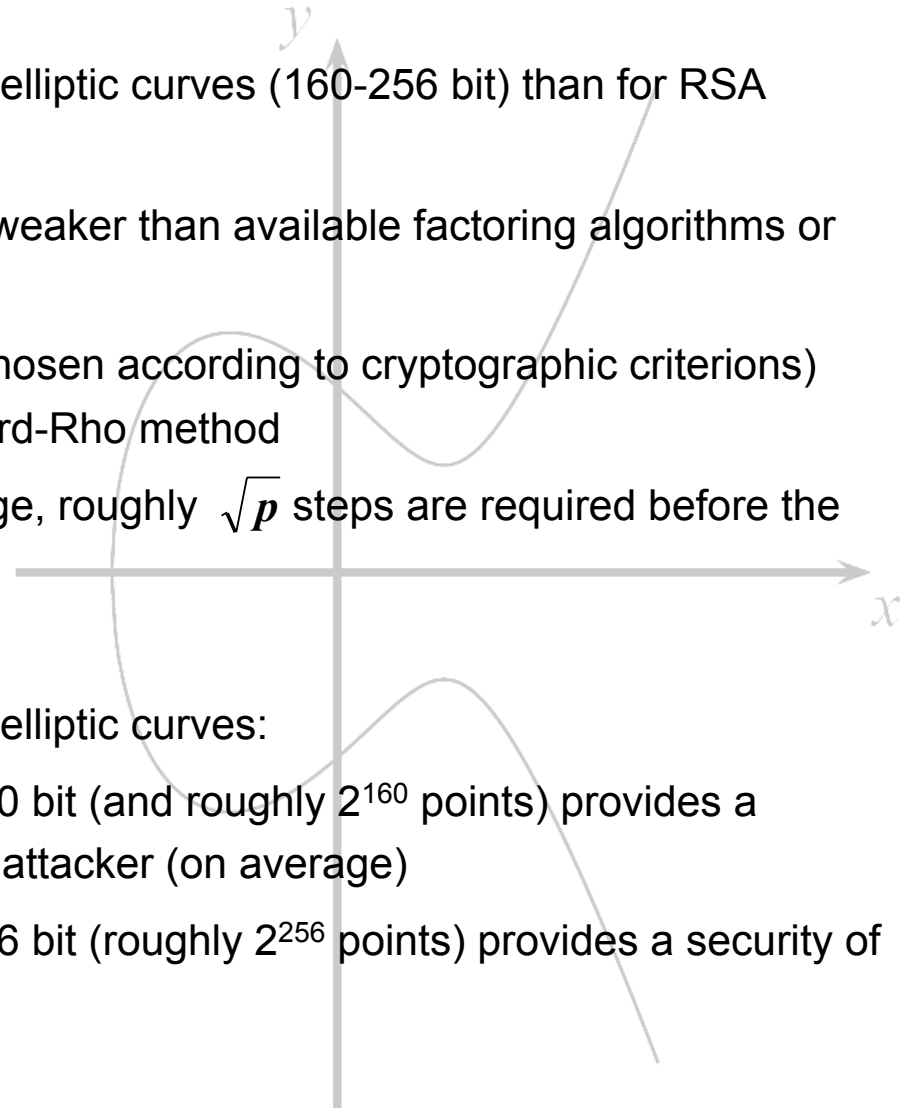
■ Content of this Chapter

- Introduction
- Computations on Elliptic Curves
- The Elliptic Curve Diffie-Hellman Protocol
- **Security Aspects**
- Implementation in Software and Hardware



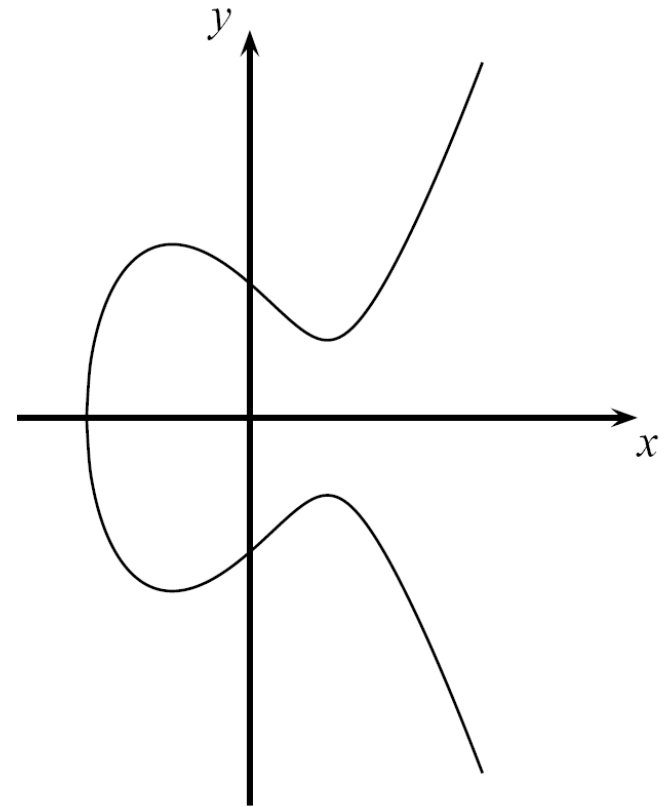
■ Security Aspects

- Why are parameters significantly smaller for elliptic curves (160-256 bit) than for RSA (1024-3076 bit)?
 - Attacks on groups of elliptic curves are weaker than available factoring algorithms or integer DL attacks
 - Best known attacks on elliptic curves (chosen according to cryptographic criterions) are the Baby-Step Giant-Step and Pollard-Rho method
 - Complexity of these methods: on average, roughly \sqrt{p} steps are required before the ECDLP can be successfully solved
- Implications to practical parameter sizes for elliptic curves:
 - An elliptic curve using a prime p with 160 bit (and roughly 2^{160} points) provides a security of 2^{80} steps that required by an attacker (on average)
 - An elliptic curve using a prime p with 256 bit (roughly 2^{256} points) provides a security of 2^{128} steps on average



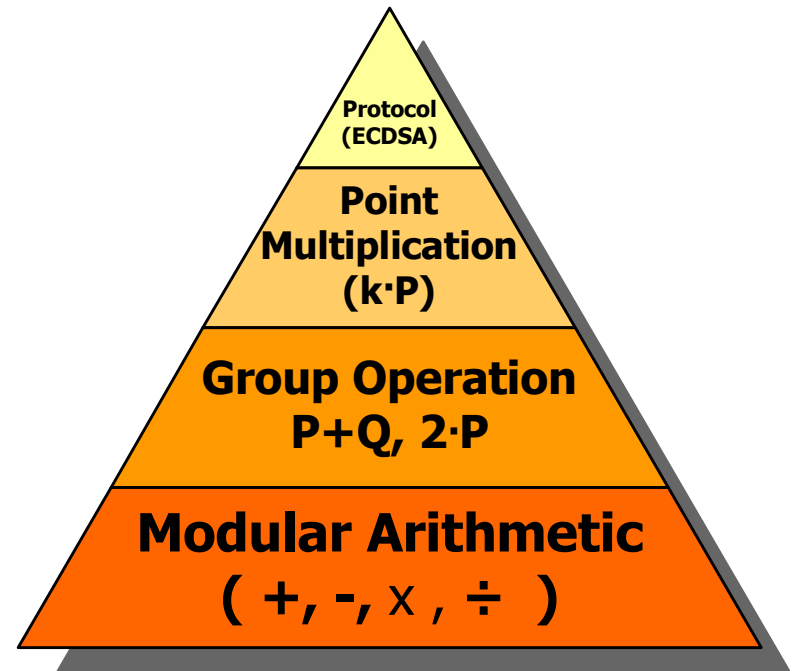
■ Content of this Chapter

- Introduction
- Computations on Elliptic Curves
- The Elliptic Curve Diffie-Hellman Protocol
- Security Aspects
- **Implementation in Software and Hardware**



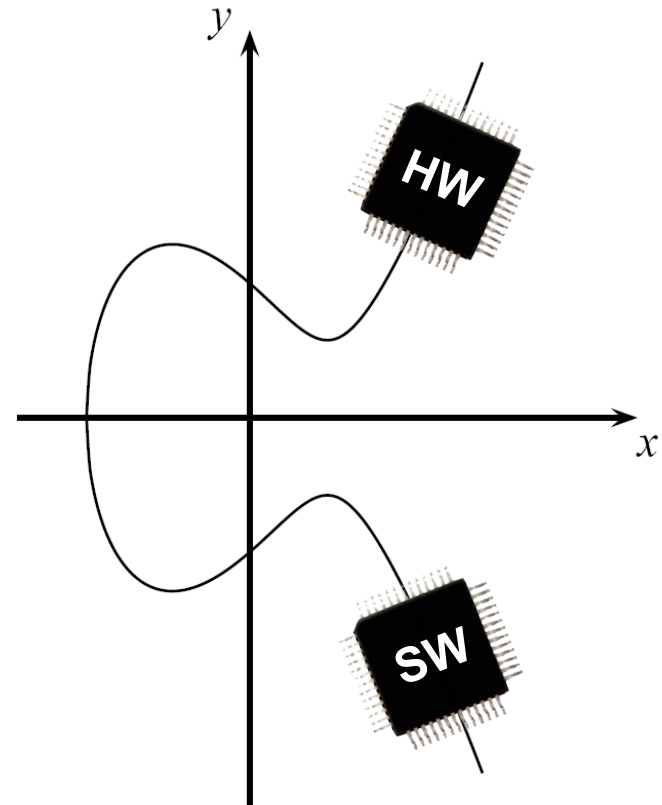
■ Implementations in Hardware and Software

- Elliptic curve computations usually regarded as consisting of four layers:
 - Basic modular arithmetic operations are computationally most expensive
 - Group operation implements point doubling and point addition
 - Point multiplication can be implemented using the Double-and-Add method
 - Upper layer protocols like ECDH and ECDSA
- Most efforts should go in optimizations of the modular arithmetic operations, such as
 - Modular addition and subtraction
 - Modular multiplication
 - Modular inversion



■ Implementations in Hardware and Software

- Software implementations
 - Optimized 256-bit ECC implementation on 3GHz 64-bit CPU requires about 2 ms per point multiplication
 - Less powerful microprocessors (e.g, on SmartCards or cell phones) even take significantly longer ($>10\text{ ms}$)
- Hardware implementations
 - High-performance implementations with 256-bit special primes can compute a point multiplication in a few hundred microseconds on reconfigurable hardware
 - Dedicated chips for ECC can compute a point multiplication even in a few ten microseconds



■ Lessons Learned

- Elliptic Curve Cryptography (ECC) is based on the discrete logarithm problem. It requires, for instance, arithmetic modulo a prime.
- ECC can be used for key exchange, for digital signatures and for encryption.
- ECC provides the same level of security as RSA or discrete logarithm systems over \mathbb{Z}_p with considerably shorter operands (approximately 160–256 bit vs. 1024–3072 bit), which results in shorter ciphertexts and signatures.
- In many cases ECC has performance advantages over other public-key algorithms.
- ECC is slowly gaining popularity in applications, compared to other public-key schemes, i.e., many new applications, especially on embedded platforms, make use of elliptic curve cryptography.