Cryptography

Lecture 12

Elliptic curve cryptography (ECC)

- Motivations
- Elliptic curves over $\mathbb{R}$.
- Addition; geometric and algebraic view.
- Elliptic curves over $\mathbb{Z}_p^*$.
- Cryptographic primitives.

Motivations

Subexponential algorithms

For both factoring of integers and discrete logs in $\mathbb{Z}_p^*$ we have seen subexponential algorithms, i.e. a doubling of the number of bits of security requires more than a doubling of the key lengths.

NIST key recommendations (SP 800-57, Part 1, January 2016)

Key sizes with approximately equal amount of work to break.

Security	Lifetime	Symm. encr.	RSA	DH subgrp
80	Deprecated	3DES (2 keys)	1024	160/1024
112	Until 2030	3DES	2048	224/2048
128	Beyond 2030	AES-128	3072	256/3072
256		AES-256	15360	512/15360

Implications for the good guys

Table on previous slide shows key sizes necessary to achieve acceptable security.

Note that many cryptographic applications require protection for long periods of time, requiring large key sizes (e.g. 3072 bits). But this also means that

- encryption/decryption/signing time and power consumption for public key operations become a problem in many applications.
- key/signature/certificate sizes become a problem in many applications.

Can we do better?

Discrete logarithms, revisited

The generic algorithms for discrete log problems in general cyclic groups of prime order q (e.g. Pollard's ρ) have (expected) complexity $\sqrt{q}$, which is exponential in size of q ($2^{k/2}$ for k bit primes).

Known subexponential algorithm (index calculus) works only in $\mathbb{Z}_p$.

One can prove that generic algorithms, i.e. those that work in any cyclic group, cannot have complexity below $\sqrt{q}$.

The quest


WANTED!!
Cyclic groups for Diffie−Hellman etc.
Requirements:
− can be generated in different sizes
(in particular prime sizes ca 2^160 and up)
− convenient repr of elements as bit strings
− fast group operation on computers
− no known fast discrete log algorithm

Enter Elliptic Curve Cryptography (ECC)

One more column in NIST's table

The NIST report also compares with ECC algorithms:

Security	Lifetime	...	ECC key size
80	Deprecated	...	≥ 160
112	Until 2030	...	≥ 224
128	Beyond 2030	...	≥ 256
256		...	≥ 512

For 128 bits security, ECC algorithms are ≈ 10 times better than traditional public key algorithms in both time and space.

NSA's view in 2005

"NSA has determined that beyond the 1024-bit public key cryptography in common use today, rather than increase key sizes beyond 1024 bits, a switch to elliptic curve technology is warranted."

256 bits security??

When do we need 256 bits?

If 112 bits security is enough through 2030, and 128 bits beyond 2030, why have they standardized 256 bit keys?

Grover's algorithm

This algorithm, which only works on quantum computers, can perform unordered search in a set with N elements in time $O(\sqrt{N})$, using brute force.

So, if quantum computers become a reality, 256 bit keys give 128 bits security against brute force attacks on AES.

Do you believe in quantum computers?

NSA requires 256 bit keys to encrypt classified information . . .

NSA Suite B

In 2005, NSA announced its Suite B of cryptographic algorithms, intended to provide the basis for US government use of cryptography. Suite B includes

- AES block cipher with 128 and 256 bit keys.
- SHA-256 hash functions.
- ECDSA (256 and 384 bit moduli) for digital signatures.
- EC D-H (256 and 384 bit moduli) for key negotiation.

New NSA recommendations, August 2015

- AES block cipher with 256 bit keys.
- SHA-384 hash functions.
- ECDSA (384 bit) or 3072 bit RSA for digital signatures.
- EC D-H (384 bit) or 3072 bit D-H for key negotiation.

More work initiated on quantum-resistent cryptography.

What are elliptic curves?

First answer/example: P-256 from FIPS 186-4

Elements are (some) pairs (x, y) where
$x, y \in$
$\mathbb{Z}_{115792089210356248762697446949407573530086143415290314195533631308867097853951}$.
If $x_1 \neq x_2$, then $(x_1, y_1) * (x_2, y_2) = (x_3, y_3)$ where

$$\begin{aligned} m &= (y_2 - y_1)/(x_2 - x_1) \\ x_3 &= m^2 - x_1 - x_2 \\ y_3 &= m(x_1 - x_3) - y_1. \end{aligned}$$

(similar formula when $x_1 = x_2$).
Group order is
115792089210356248762697446949407573529996955224135760342422259061068512044369.

Generator is
(48439561293906451759052585252797914202762949526041747995844080717082404635286,
36134250956749795798585127919587881956611106672985015071877198253568414405109).

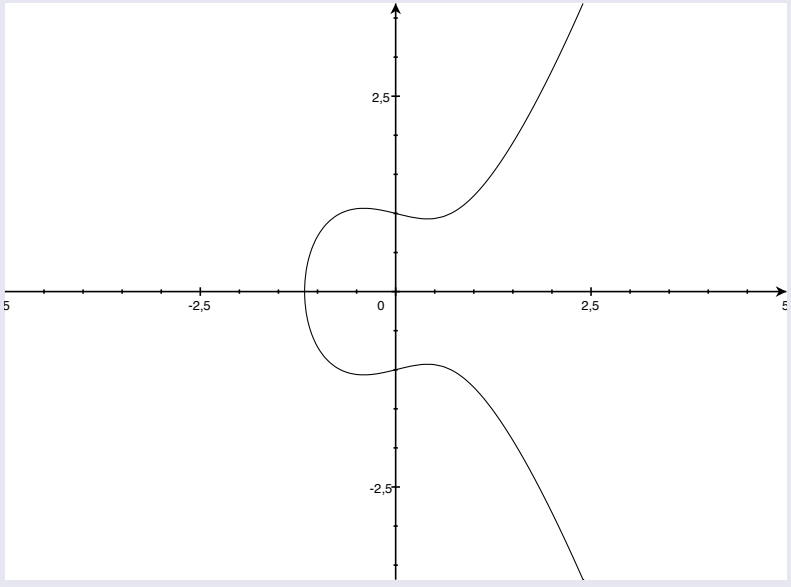
The above is enough for computation – but we need better intuitions.

What are elliptic curves?

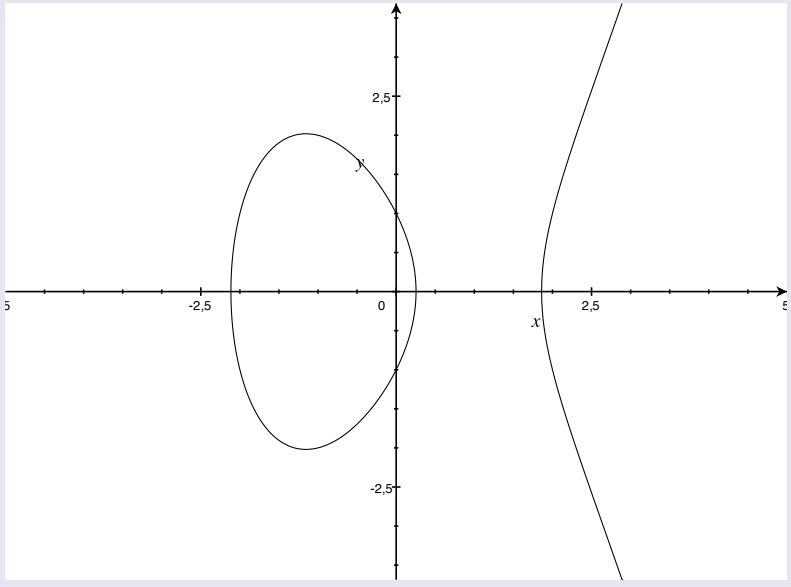
Second answer: Elliptic curves over $\mathbb{R}$

An elliptic curve over $\mathbb{R}$ is specified by two real parameters b and c. The curve is the set of points (x, y) that satisfy $y^2 = x^3 + bx + c$.

$b = -0.5, c = 1$



$b = -4, c = 1$



Simple observations

- The curve has one or two connected components depending on whether the equation RHS = 0 has one or three roots (RHS = right hand side).
- We exclude the singular case where the RHS has two roots (one of them double); thus we require $4b^3 + 27c^2 \neq 0$.
- The curve is symmetric around the x-axis and is differentiable everywhere.

How can one define a (commutative and associative) binary operation $*$ on such a curve?

Given two points P and Q on the curve, we want to specify a third $P * Q$.

A geometric idea

If $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ determine a line with equation $y = mx + r$, then that line will cross the curve in a third point, since

$$(mx + r)^2 = x^3 + bx + c$$

has three roots. (We cannot have the one-root case, since we started from two points).

Maybe this point could be $P * Q$?

Remaining problems

1. What if $P = Q$?
2. What if P and Q determine a vertical line (equation $x = a$)?
3. What if the third root x_3 is equal to one of x_1 or x_2?
4. Will this operation be associative?

Fixing the problems

The following modification tries to solve the problems from the previous slide.

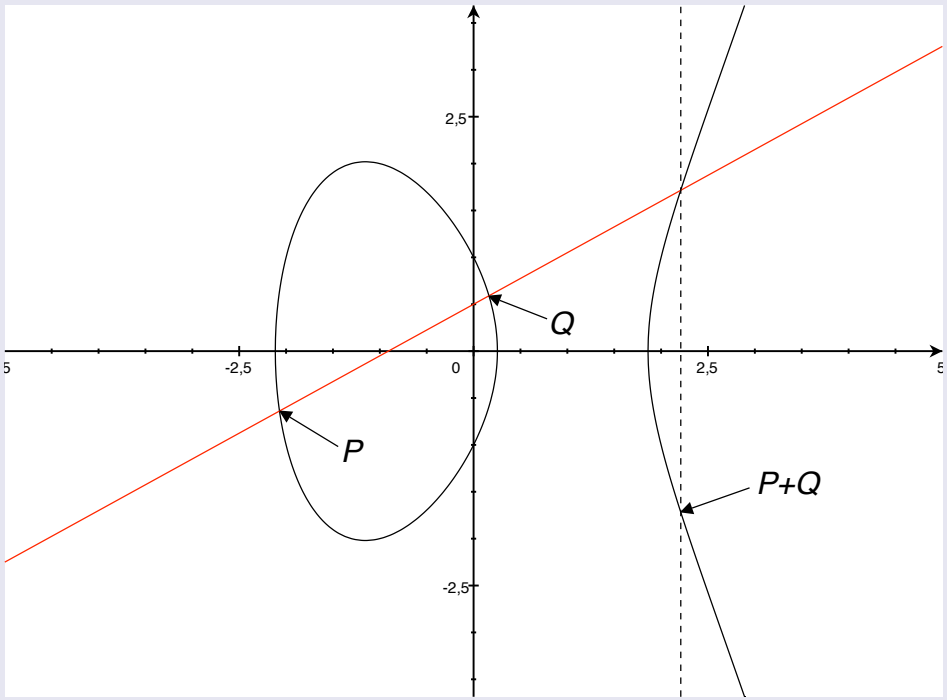
1. If $P = Q$, use the tangent of the curve at P as the line to determine the third point; will work if the tangent is not vertical.
2. If the line is vertical: We add one more point, "the point at infinity" ∞, to the set and prescribe that if $x_1 = x_2$ (and $y_2 = -y_1$), then $P * Q = \infty$.

   New problem: What is $P * \infty$?
   Answer: We put $P * \infty = \infty * P = P$, so ∞ is the unit wrt $*$.
3. If $x_3 = x_1$: We do not want $P * Q = P$, since that would force Q to be the unit wrt $*$. So we let $P * Q = (x_1, -y_1)$, and do this mirroring for all points!

   We need a new picture of the geometric intuition.
   From now on, we also follow tradition and call the operation "addition", denoted by +.

The geometric view of addition

The following picture illustrates how to find $P + Q$ when P and Q determine a non-vertical line:



Additional rules

- When $P = Q$, use tangent.
- When line is vertical, $P + Q = \infty$.
- $P + \infty = \infty + P = P$.

Still remaining

+ is clearly commutative, but what about associativity?

Algebraic formulas for addition

The case when $x_1 \neq x_2$

The slope m of the line is $m = (y_2 - y_1)/(x_2 - x_1)$.

We rewrite $(mx + r)^2 = x^3 + bx + c$ as
$x^3 - m^2x^2 + (b - 2mr)x + (c - r^2) = 0$.
The LHS is $(x - x_1)(x - x_2)(x - x_3)$,
so $m^2 = x_1 + x_2 + x_3$, i.e. $x_3 = m^2 - x_1 - x_2$.

Since P is on the line, $r = y_1 - mx_1$,
which gives $y_3 = -(m(x_3 - x_1) + y_1)$,
($-$ because of mirroring), i.e. $y_3 = m(x_1 - x_3) - y_1$.

When $P = Q$

Differentiating gives $2yy' = 3x^2 + b$, so slope at P is $m = (3x_1^2 + b)/(2y_1)$.

Associativity

Now it is just tedious work to check associativity.

Elliptic curves for cryptography?

Are these groups what we wanted?

- The number of points on a curve is infinite, not a large prime.
- Addition and point representation require real numbers.

Can we do similar things without real numbers?

Third answer: Elliptic curves over a finite field

We can consider the equation $y^2 = x^3 + bx + c$ where the coefficients b and c and also x and y are elements not of $\mathbb{R}$, but of any field.

We will do so only for the field $\mathbb{Z}_p$ with p a (large) prime.

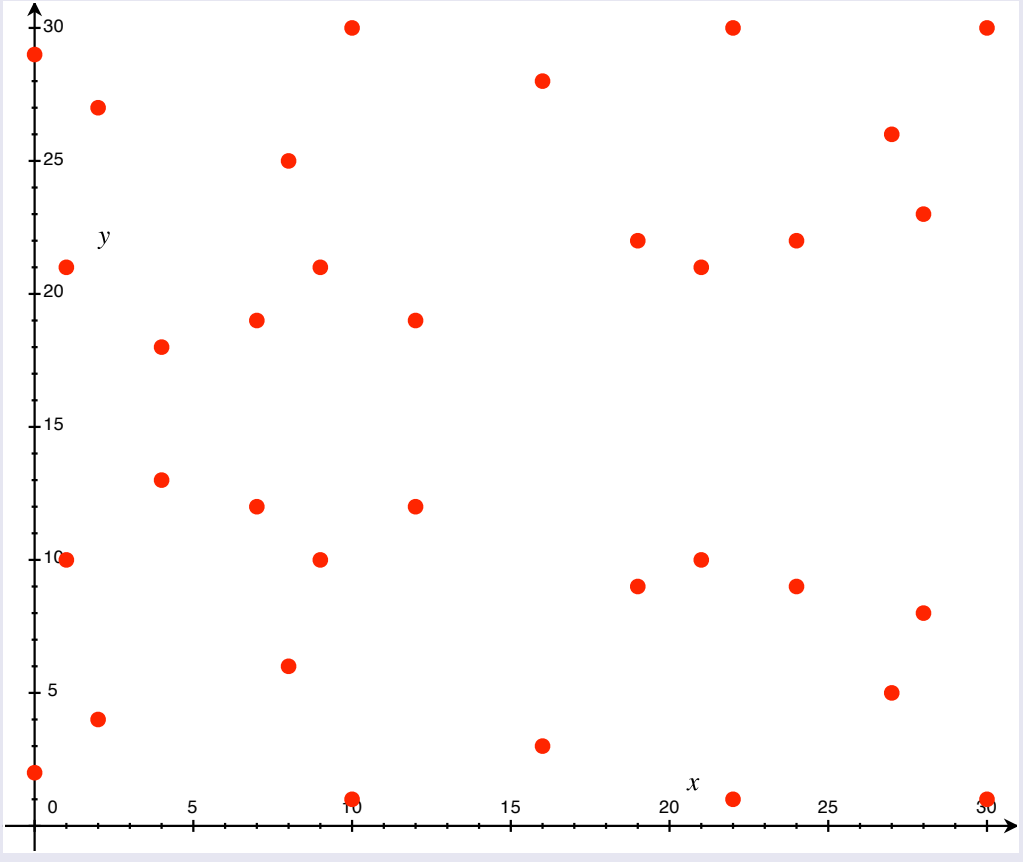
Example

We consider $y^2 = x^3 + 1$ in $\mathbb{Z}_{11}$.
For this small p we can tabulate the solutions:

x	0	1	2	3	4	5	6	7	8	9	10
x^3+1	1	2	9	6	10	5	8	3	7	4	0
y	1, 10	-	3, 8	-	-	4, 7	-	5, 6	-	2, 9	0

So, this “curve” consists of the 12 points (0,1), (0,10), (2,3), (2,8), (5,4), (5,7), (7,5), (7,6), (9,2), (9,9), (10,0) and ∞.

A somewhat larger example



This is

$$y^2 = x^3 + 2x + 4$$

in $\mathbb{Z}_{31}$.

Group has 35 elements (including ∞, not shown in graph).

What about symmetry around the x-axis?

Addition

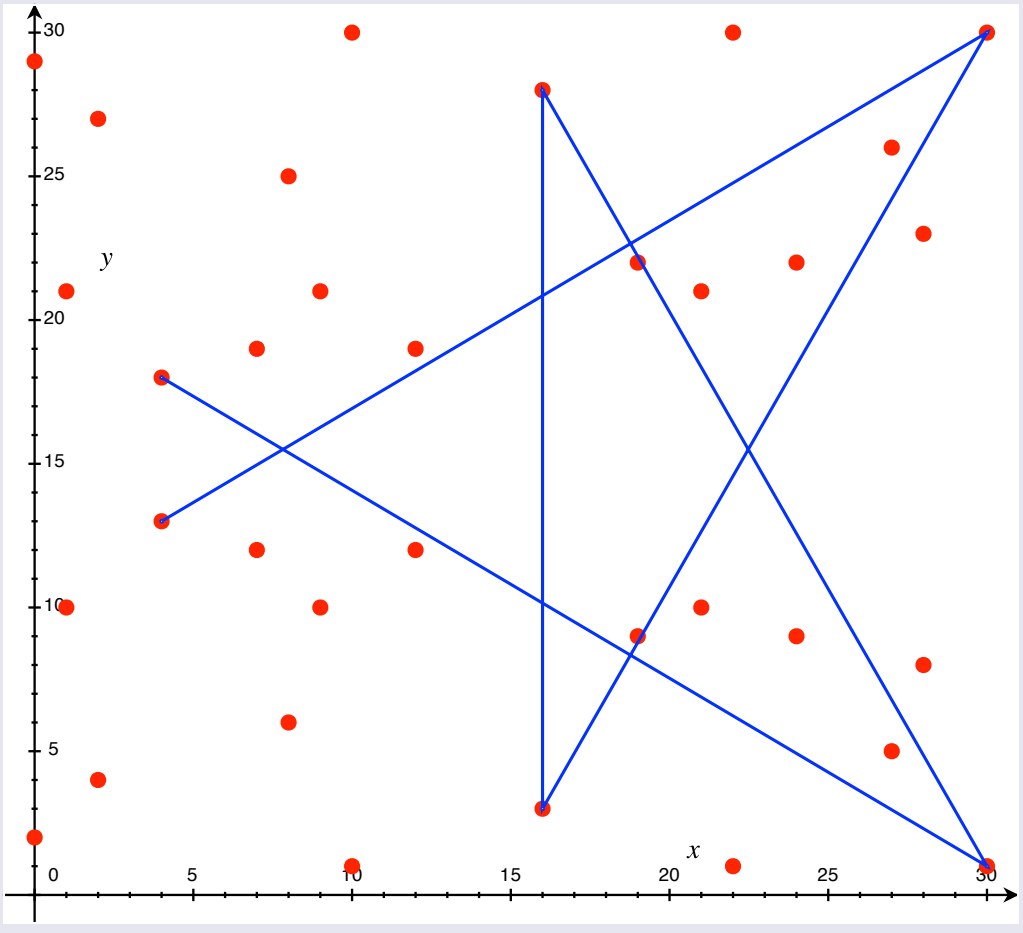
The formulas for addition make sense also when we work in $\mathbb{Z}_p$, even though the notion of a "curve" is non-intuitive. So, we define addition in the same way:

- If $x_1 \neq x_2$, we have $(x_1, y_1) + (x_2, y_2) = (x_3, y_3)$ where

$$\begin{aligned} m &= (y_2 - y_1)/(x_2 - x_1) \\ x_3 &= m^2 - x_1 - x_2 \\ y_3 &= m(x_1 - x_3) - y_1. \end{aligned}$$

- $(x1, y1) = (x2, y2)$, the value of m is changed to $m = (3x_1^2 + b)/(2y_1)$.
- If $x_1 = x_2$ and $y_1 = -y_2 \neq 0$, then $(x_1, y_1) + (x_2, y_2) = \infty$.

A cyclic subgroup



The lines connect the points of the subgroup generated by $(4, 18)$.

Subgroup has 7 elements (including ∞, not shown in graph).

Can you now guess the setting we will work in?

Elliptic curve cryptography (ECC)

Setting

We choose a subgroup of prime order of an elliptic curve over $\mathbb{Z}_p$ for some large prime p.
Here we can pose the discrete log and Diffie-Hellman problems.

Best known attacks are Pollard algorithms; we need subgroup size $\approx 2^{256}$ for 128 bits security.

Binary fields

There is an alternative setting (binary fields), which we ignore.

Questions

1. How do we find the elliptic curve?
2. How do we find the subgroup and its generator (here called base point)?
3. How many points are there on an elliptic curve over $\mathbb{Z}_p$?

Answers to questions

1. Not easy. In practice, use one of the curves in some published standard, such as FIPS186-4 (DSS standard from NIST), SEC2 (industry consortium), ANSI X9.62 etc. Curves are provided for different values of p; bit sizes include (160, 192,) 224, 256, 384 and 521 bits.
2. The curves in the standards are typically already of prime order, so no subgroup is needed.
3. Intuition: each $x \in \mathbb{Z}_p$ generates either zero or two points, depending on if the value of the RHS is a square or not. We should expect $\approx p$ points.

   Strict estimate (Hasse's theorem, not proved here): $|N - (p+1)| < 2\sqrt{p}$.

ECC Diffie-Hellman

To test our understanding, let us define elliptic curve Diffie-Hellman.

Key setup

A community of users agree on domain parameters, i.e.

- an elliptic curve (i.e. p, b and c must be specified),
- a base point P, that generates a (sub-)group of prime order q.

Typically one selects a published set of parameters.

Key agreement

Alice chooses $a \overset{R}{\leftarrow} \{1, 2, \ldots, q-1\}$ at random, computes $A = aP$ and sends A to Bob.
Bob chooses $b \overset{R}{\leftarrow} \{1, 2, \ldots, q-1\}$ at random, computes $B = bP$ and sends B to Alice.
Both Alice and Bob can compute $(x, y) = (ab)P$.

Other ECC primitives

Carried over from $\mathbb{Z}_p^*$

Many discrete log-based primitives have a direct ECC counterpart, e.g.

- Elgamal encryption.
- DSA signatures.

Keep in mind the analogies

Traditional ($\mathbb{Z}_p^*$)	ECC
$x \cdot y$	$P + Q$
g^2	$2P$
g^n	nP

Summary

New setting,
more sophisticated group,
but same main ideas,
only more efficient!

Points, keys and messages

Keys

After ECC Diffie-Hellman the parties have agreed on a common secret P, a point on the elliptic curve. To create e.g. a 128 bit session key from this, one uses a key derivation function, which typically hashes the coordinates to create a key.

Messages

To encrypt a message m (a bitstring) using ECC Elgamal, one needs to interpret m as a point on the curve.
One possibility: we try to create a point (x, y) where $x = 80m + j$ for a value j in $0, 1, 2, \ldots 79$: Compute $x^3 + bx + c$ and see if it has a square root; then we found a point. Otherwise try next j. The probability that all j will fail is around 2^{-80}.

ECC deployment

Examples

- ECC key exchange standardized in TLS.
  Available in most web browsers and servers.
- AACS (content protection for BluRay) uses ECC key agreement.
- Bitcoin uses ECDSA signatures.
- ...

General

Still many old systems around without ECC support.
Until recently the expectation was that in a few years ECC will be everywhere.

But how to interpret NSA's new position?

Real Life Cryptography: the Dual EC bit generator

- A 2006 NIST standard included a random bit generator based on ECC. It employed the NIST curve P-256 and two specified points P and Q on that curve.
- Early analysis showed that the generator was slow and had some statistical weaknesses. Why was it standardized?
- In 2007, it was noted that if an attacker knew t such that $P = tQ$, the generator would be broken. How were P and Q chosen by NIST? Several cryptographers strongly advised against use of the generator.
- In September 2013, New York Times reported that documents leaked by Edward Snowden revealed that NSA indeed knew such t.
- In December 2013 Reuters revealed that NSA had paid RSA Security $10 millions to make the Dual EC generator the default in its BSafe software.