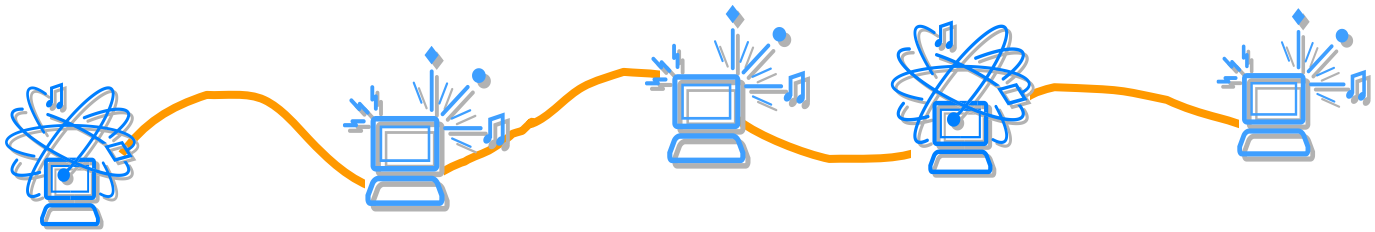
Segurança em Redes
Eliptic Curves Cryptography



Redes de Comunicação de Dados

Área Departamental de Engenharia da Eletrónica e das Telecomunicações e de Computadores

Instituto Superior de Engenharia de Lisboa

Bibliografia

Lecture Notes on “Computer and Network Security” by Avinash Kak
[https://engineering.purdue.edu/kak/compsec/]

ECC use

If you want to combine forward secrecy, with authentication, a commonly used algorithm today is **ECDHE-RSA**.

In ECDHE-RSA, RSA is used for certificate based authentication using the TLS/SSL protocol and ECDHE used for creating a one-time session key.

It is also possible to use DHE-RSA, which uses the regular Diffie-Hellman Exchange protocol for creating session keys, for the same purpose. However, it is likely to get greater security with ECDHE-RSA.

Elliptic curves – What are they?

First and foremost, elliptic curves have nothing to do with ellipses. Ellipses are formed by quadratic curves. Elliptic curves are always cubic.

The simplest possible “curves” are, of course, straight lines.

The next simplest possible curves are conics, these being quadratic forms of the following sort

$$\mathbf{ax^2 + bxy + cy^2 + dx + ey + f = 0}$$

If b^2-4ac is less than 0, then the curve is either an ellipse, or a circle, or a point, or the curve does not exist; if it is equal to 0, then we have either a parabola, or two parallel lines, or no curve at all; if it is greater than 0, then we either have a hyperbola or two intersecting lines. (Note that, by definition, a conic is the intersection of a plane with two cones that are joined at their tips.)

Elliptic curves – What are they?

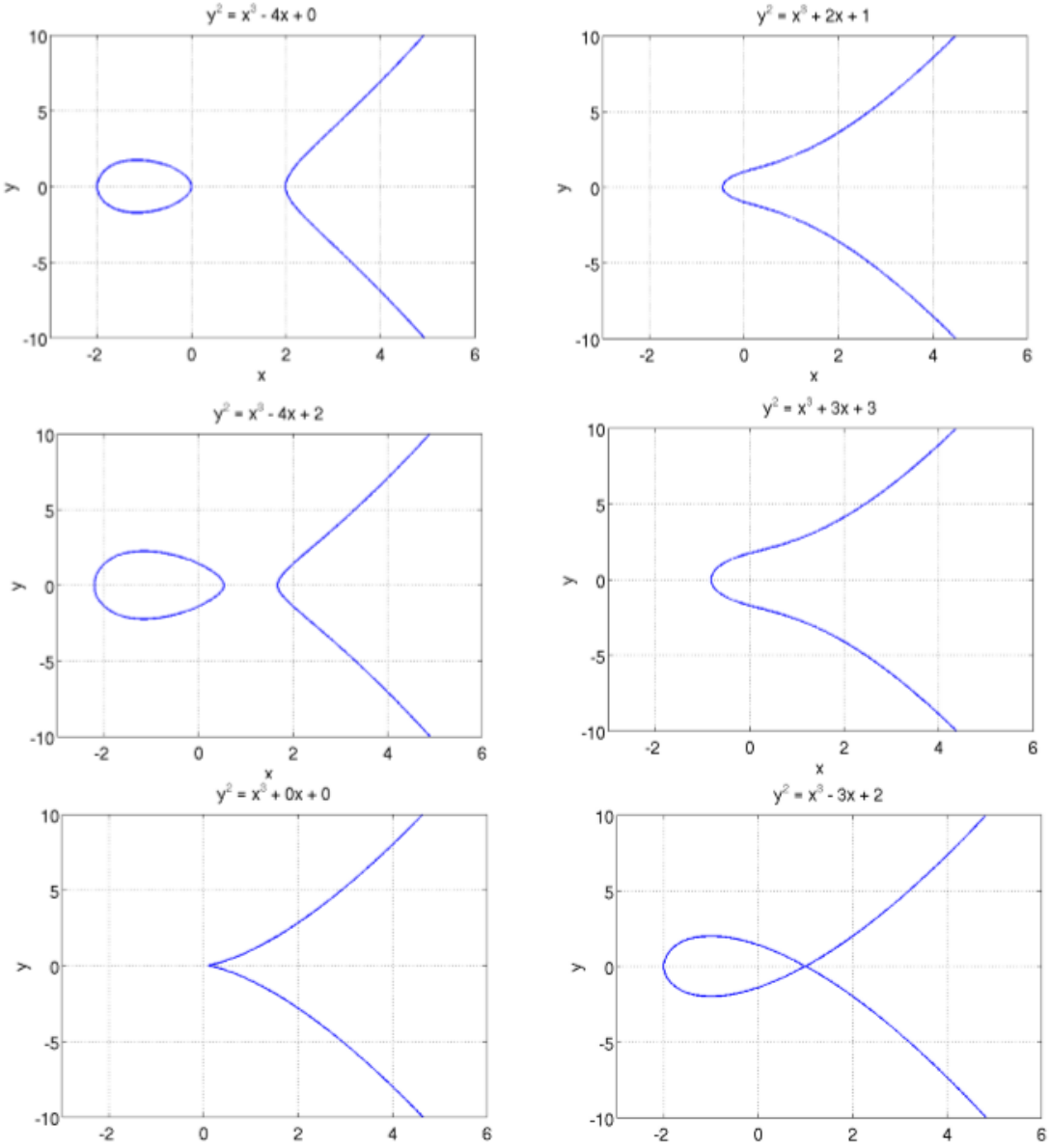
The next simplest possible curves are elliptic curves. An elliptic curve in its "standard form" is described by

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

for some fixed values for the parameters a and b. This equation is also referred to as **Weierstrass Equation** of characteristic 0.

The top four curves all look smooth (they do not have cusps, for example) because they all satisfy the following condition on the discriminant of the polynomial $f(x) = x^3 + ax + b$:

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

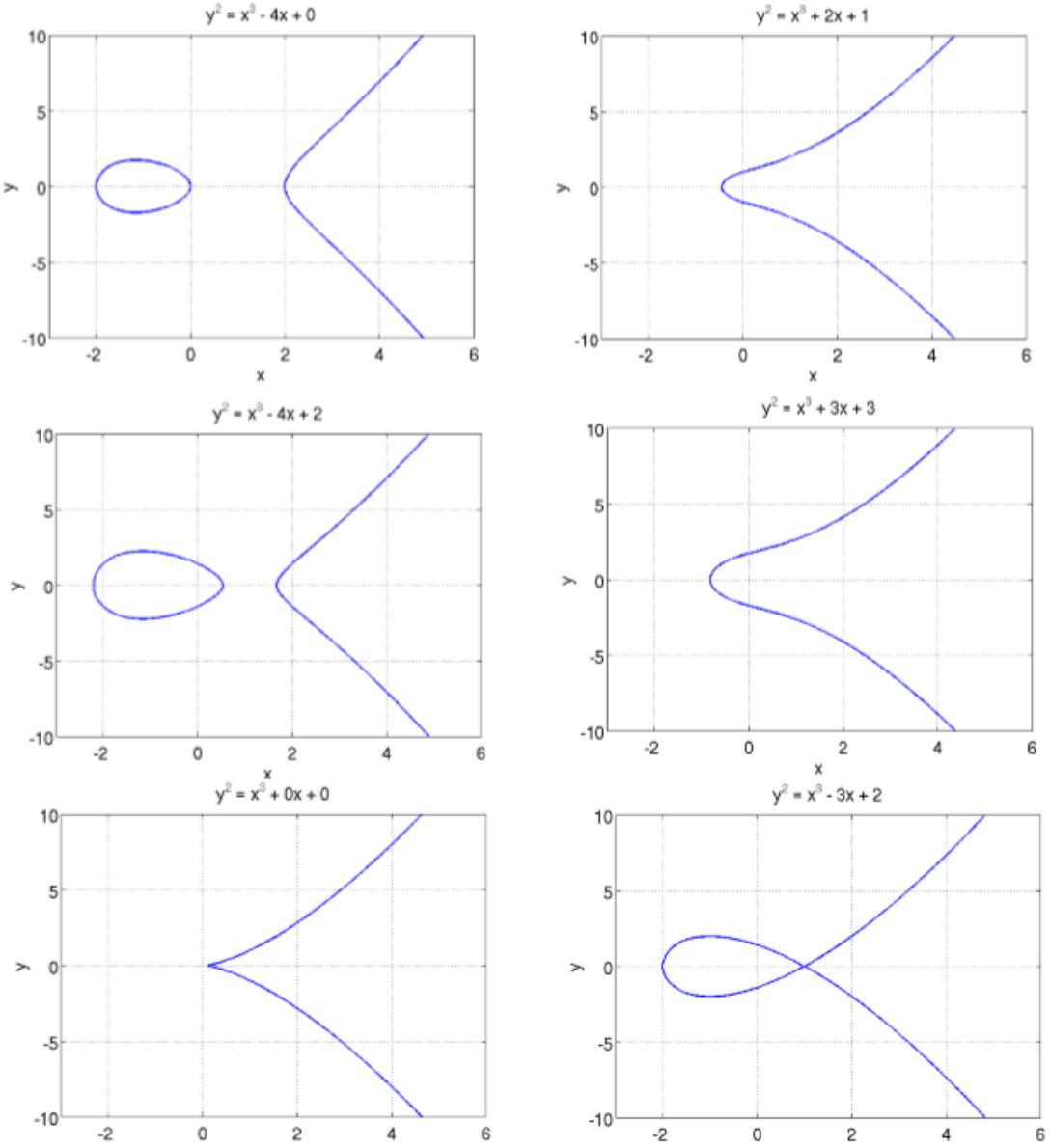


Elliptic curves for different values of the parameters a and b. (*This figure is from Lecture 14 of "Lecture Notes on Computer and Network Security" by Avi Kak.*)

Elliptic curves – What are they?

The bottom two examples in the figure show two elliptic curves for which the condition on the discriminant is violated. For the one on the left that corresponds to $f(x) = x^3$, all three roots of the cubic polynomial have coalesced into a single point and we get a cusp at that point. For the one on the right that corresponds to $f(x) = x^3-3x+2$, two of the roots have coalesced into the point where the curve crosses itself. These two curves are **singular**.

It is not safe to use singular curves for cryptography.



Elliptic curves for different values of the parameters a and b. (*This figure is from Lecture 14 of "Lecture Notes on Computer and Network Security" by Avi Kak.*)

Elliptic curves – What are they?

Note that since we can write

$$y = \pm \sqrt{x^3 + ax + b}$$

elliptic curves in their standard form will be symmetric about the x-axis.
It is difficult to comprehend the structure of the curves that involve polynomials of degree greater than 3.

To give the reader a taste of the parameters used in elliptic curves meant for real security, here is an example:

$$y^2 = x^3 + 3176890812513255034763174764138276932727469559 27x + 790528966078787587181205720257185354321006519 34$$

This elliptic curve is used in the Microsoft Windows Media Digital Rights Management Version 2.

Elliptic Curve Arithmetic

- Most of the products and standards that use public-key cryptography for encryption and digital signatures use RSA
  - The key length for secure RSA use has increased over recent years and this has put a heavier processing load on applications using RSA
- Elliptic curve cryptography (ECC) is showing up in standardization efforts including the IEEE P1363 Standard for Public-Key Cryptography
- Principal attraction of ECC is that it appears to offer equal security for a far smaller key size

Abelian Group

A set of elements with a binary operation, denoted by ●, that associates to each ordered pair (*a, b)* of elements in *G* an element

$(a \bullet b)$ in *G,* such that the following axioms are obeyed:

(A1) Closure: If *a* and *b* belong to *G*, then $a \bullet b$ is also in G

(A2) Associative: a●(b●c) = (a●b) ●c for all a, b, a in G

(A3) Identity element: There is an element **e** in *G* such that

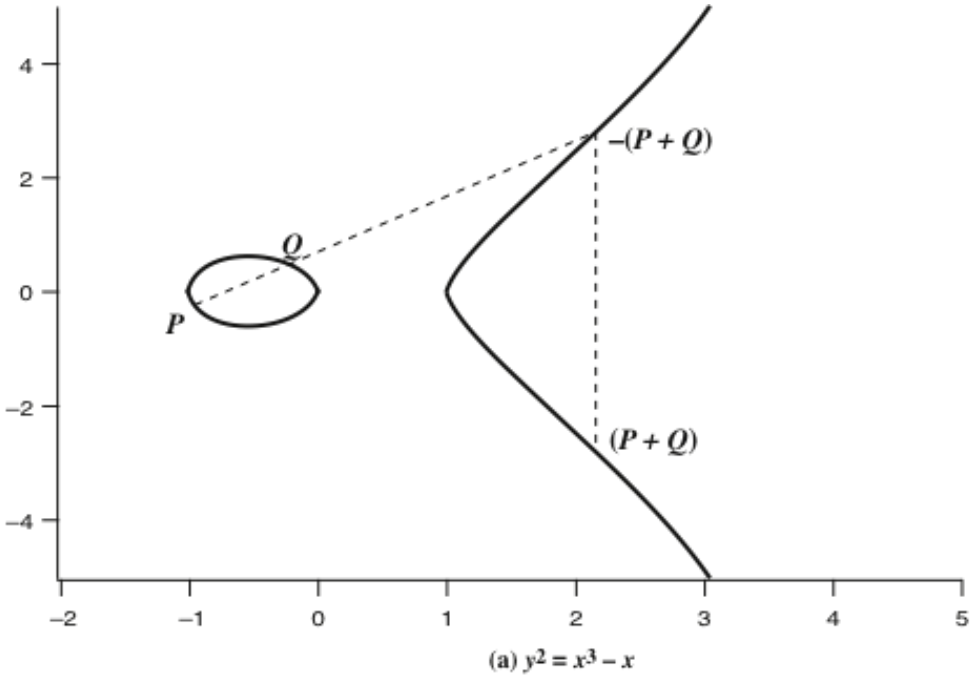
a●e = e●a = a for all a in G

(A4) Inverse element: For each *a* in *G* there is an element *a'* in *G* such that

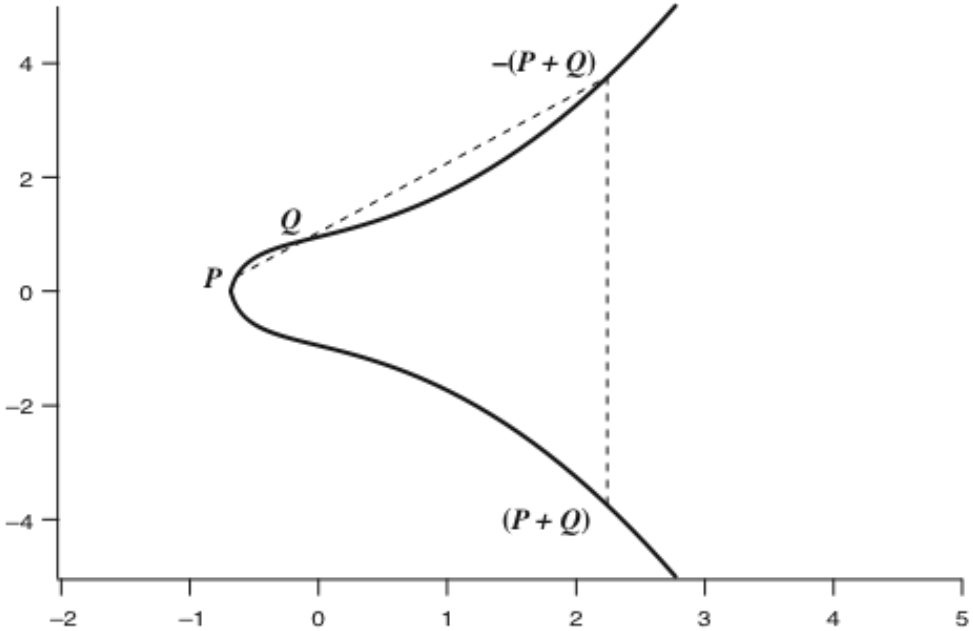
a●a' = a' ● a = e

(A5) Commutative: a●b = b●a for all a, b in G

Example of Elliptic Curves



(a) $y^2 = x^3 - x$



(b) $y^2 = x^3 + x + 1$

Elliptic Curves Over Z to the p power

- Elliptic curve cryptography uses curves whose variables and coefficients are finite
- Two families of elliptic curves are used in cryptographic applications:
  - **Prime curves over** Z_p
    - Use a cubic equation in which the variables and coefficients all take on values in the set of integers from 0 through p-1 and in which calculations are performed modulo p
    - Best for software applications
  - Binary curves over $GF(2^m)$
    - Variables and coefficients all take on values in $GF(2^m)$ and in calculations are performed over $GF(2^m)$
    - Best for hardware applications

Points (other than O) on the Ellipitic Curve E sub 2 3 left parenthesis 1, 1 right parenthesis

(0,1)	(6,4)	(12,19)
(0,22)	(6,19)	(13,7)
(1,7)	(7,11)	(13,16)
(1,16)	(7,12)	(17,3)
(3,10)	(9,7)	(17,20)
(3,13)	(9,16)	(18,3)
(4,0)	(11,3)	(18,20)
(5,4)	(11,20)	(19,5)
(5,19)	(12,4)	(19,18)

Security of Elliptic Curve Cryptography

- Depends on the difficulty of the elliptic curve logarithm problem
- Fastest known technique is “Pollard rho method”
- Compared to factoring, can use much smaller key sizes than with RSA
- For equivalent key lengths computations are roughly equivalent
- Hence, for similar security ECC offers significant computational advantages

Comparable Key Sizes in Terms of Computational Effort for Cryptanalysis (NIST SP-800-57)

Symmetric key algorithms	Diflie-Hellman, Digital Signature Algorithm	RSA (size of n in bits)	ECC (modulus size in bits)
80	*L* = 1024 *N* = 160	1024	160-223
112	*L* = 2048 *N* = 224	2048	224-255
128	*L* = 3072 *N* = 256	3072	256-383
192	*L* = 7680 *N* = 384	7680	384-511
256	*L* = 15,360 *N* = 512	15,360	521+

Note: L = size of public key, *N* = size of private key