

# M2 Cybersecurity (Univ. Grenoble Alpes/Grenoble INP)

## Cryptographic Mechanisms (Part 2) From Classical DLP...to Elliptic Curve Cryptography

v1.00, September 2016

Philippe Elbaz-Vincent



## Content of the lectures

- ❑ What is the DLP ?
- ❑ Classical cryptographic methods based on the DLP
- ❑ Solving the DLP (the generic attacks...)
- ❑ Elliptic Curve Cryptography (ECC) and the DLP
- ❑ Protocols based on ECC
- ❑ Computational costs and keysize comparison with RSA

## What is the DLP ?

Let  $G$  be a group and  $g$  an element of order  $p$  with  $p$  prime (it is not a necessary assumption but it is simpler). If  $h = g^r$  for a certain natural number  $r < p$ , we say that  $r$  *is the discrete logarithm of  $h$  with basis  $g$* .

**Experimental Fact :** Knowing  $h$  and  $g$ , it is very difficult in general to find  $r$  if  $p$  is big enough. It is what we call the *Discrete Logarithm Problem* or *DLP* for short.

**Idea :** Use this difficulty as basis for a cryptosystem.

## Difference between DLP and RSA

In contrast with the RSA setting, the (discrete exponential) function  $\exp : \mathbb{Z}/(p-1)\mathbb{Z} \rightarrow G, x \mapsto g^x$  is a one-way function without trapdoor. It does not have any additional information which makes the computation of the inverse function easy. As a consequence, the cryptographic schemes will have a different flavor than with RSA.

Nevertheless, we can still notice **a multiplicative property within the DLP** : Indeed, if  $a = g^\alpha$  and  $b = g^\beta$ , then  $ab = g^{(\alpha+\beta)}$ .

Such multiplicative structure could again be used by an attacker inside a “naive cryptographic scheme” !

## Mathematical motivations

In order to use groups  $G$  with DLP (and be cryptographically successful), we need to answer three main problems :

- Presentation of  $G$  should be simple and compact (small memory footprint, short key size)
- Arithmetic operations should be fast and/or cost efficient
- We should give evidence that the DLP is hard in theory and unfeasible in practice

☞ It is difficult to solve such problems, and consequence of the first three should be to provide efficient implementation (both hardware and software) with facilities for developing counter-measures.

And of course, it should be easy to generate “real life”  $G$ .

## First examples of $G$

The practical difficulty is to find a good  $G$ , where as a matter of fact, the DLP is hard.

- 1 Let  $G = \mathbb{Z}/N\mathbb{Z}$  (in this case the group law is  $+$ ). If  $g \in G$ ,  $g^r$  is just  $rg$ . Then solving the DLP is easy! (exercise)
- 2 Let  $G = (\mathbb{Z}/N\mathbb{Z})^\times$  (the invertibles of the ring  $\mathbb{Z}/N\mathbb{Z}$ ). Here, if  $N$  is large with  $g \in G$  of order  $p$  (also large) then solving the DLP is difficult.
- 3 Let  $G = \mathbb{F}_q^\times$ , the invertibles of the finite field  $\mathbb{F}_q$  (with  $q = \ell^\nu$  and  $\ell$  prime). Then again if  $q$  is large enough, solving the DLP is difficult.

## Experiments with PARI/GP

Do several experiments on the above examples with PARI/GP using `znorder` and `fforder` in order to convince yourself that it is not an easy computation. Compare the size of  $p$ , the difficulty to solve the DLP w.r.t.  $p$  and the difficulty to factorise an RSA modulus of the same size. In particular do several experiments with different type of  $p$ 's and see how the difficulty may vary with the “form of  $p$ ” (depending if  $p - 1$  has, for instance, lot of small factors or if  $p$  is of the form  $2p' + 1$  with  $p'$  also prime, and so on).

# The Diffie/Hellman “Key Exchange”

The Diffie/Hellman (DH) protocol works as follows. Suppose that  $A$  and  $B$  wish to agree on a common secret key while communicating over an insecure channel. They can perform the procedure below :

- 1 They agree on a group  $G$ , a large prime number  $p$  and an element  $g \in G$  of order  $p$ .
- 2  $A$  chooses an integer  $\alpha \in \{0, 1, \dots, p-2\}$  randomly, computes  $a = g^\alpha$  and sends the result  $a$  to  $B$  (but **keep  $\alpha$  secret**).
- 3  $B$  chooses an integer  $\beta \in \{0, 1, \dots, p-2\}$  randomly, computes  $b = g^\beta$  and sends the result  $b$  to  $A$  (but **keep  $\beta$  secret**).
- 4  $A$  computes  $b^\alpha$  and  $B$  computes  $a^\beta$ .
- 5 The common secret key is  $K = g^{\alpha\beta}$ .



Whitfield Diffie

(Chief Security Officer at Sun)

*The fathers of public key cryptography...*



Martin Hellman

(Professor Emeritus at Stanford)



## The data for the DH Key exchange

Notice that the parameters  $G$ ,  $g$  and  $p$  are public. The attacker has  $g^\alpha$  and  $g^\beta$ , if he/she is able to solve the DLP easily in  $G$  (with basis  $g$ ) then he/she will be able to retrieve  $K$ .

☞ The DHP or Diffie/Hellman Problem is to retrieve  $K$  from the public data. We do not know if solving the DHP is equivalent to solving the DLP.

**Remark :** the protocol does not provide authentication between  $A$  and  $B$ .

**Exercise :** Illustrate the DH protocol with PARI/GP with finite fields (primary and non-primary) of size 160 bits.

## Main caveat of the DH protocol

As pointed out, the DH protocol does not provide authentication between the parties. As a result, the protocol is weak against a “Man/Woman-in-the-middle” attack. In the previous scheme, let  $C$  be a malicious third party. Then, he/she does the following :

- $C$  intercepts both  $a$  and  $b$  and send instead  $c = g^\gamma$  to  $A$  and  $B$
- Then he/she can relay the messages from  $A$  to  $B$  (and see their contents).

## The ElGamal method (Part I)

This public key protocol was described by Taher ElGamal (an Egyptian American cryptographer who pioneered the SSL protocol when he was at Netscape) in 1984.



Taher ElGamal (appointed CTO of Tumbleweed in October 2006)

**Public Parameters :** We fix a commutative group  $G$  and an element  $g$  of order  $p$  with  $p$  a large prime (strictly speaking we do not need this, we only need the order large with a large prime divisor).

**Private key :** a random integer  $s \in \{1, \dots, p-1\}$ .

**Public key :**  $pub = g^s$ .

Suppose that we want to send a message  $m \in G$  to a recipient. We get the public key say  $pub$  and we generate a random integer  $r \in \{1, \dots, p-1\}$ . We compute

$$K_r = g^r, \quad C = m \times pub^r.$$

## The ElGamal method (Part II)

We send the couple  $(K_r, C)$



We then get  $m = C(K_r^{-s})$ . Indeed

$$pub^r = (g^s)^r = (g^r)^s = K_r^s.$$

**Remark :** Again, encryption is probabilistic. Two encipherings of the same message will (likely) get different ciphertexts.

Computation of the exponentiation is supposed to be fast (as in RSA). The *ElGamal method is used in the Digital Signature Algorithm (DSA)*.

**Exercise :** Illustrate the method with  $G = \mathbb{F}_{2^{160}}^\times$  and  $G = (\mathbb{Z}/p\mathbb{Z})^\times$  with  $p$  of length 512 bits and 1024 bits.

## Signature with the ElGamal method (Part I)

We suppose given a function  $f : G \rightarrow \mathbb{Z}/p\mathbb{Z}$  (again  $p$  is the order of  $g$ ) with our previous parameters. Now, we suppose that the “message” to be signed is  $m \in \mathbb{Z}/p\mathbb{Z}$ . We apply the following procedure.

- 1 We choose  $r \in \{1, \dots, p-1\}$  *randomly* and compute  $K_r = g^r$ .
- 2 We compute  $k_m$  such that

$$m = sf(K_r) + k_m r \mod p.$$

- 3 We send to the recipient  $m$  and the pair  $(K_r, k_m)$ .
- 4 The recipient accepts the signature if the following equation holds

$$pub^{f(K_r)} K_r^{k_m} = g^m.$$

**Exercise :** Check that the validity of the above signature does make sense !

## Signature with the ElGamal method (Part II)

From this signature scheme was derived one of the classical standard for signature known as the Digital Signature Algorithm (DSA). This was standardised by NIST (FIPS-186), IEEE1363 and others (also used in PGP for instance).

**Remark :** In practice  $f$  must preserve the randomness of  $K_r$ . It is of course possible, even if unnecessary, to choose as  $f$  a hash function but the choice of  $f$  will depend of the structure of  $G$ . If  $G$  is a classical cyclic group, we can take the “identity” or a modular reduction for  $f$ . In fact, the hash function will be used for hashing the original message, say  $M$ , and then get a shorter message  $m$  which will fit in  $\mathbb{Z}/p\mathbb{Z}$  (see the DSS for a “real-life” signature scheme).

## The DSA

The DSA is a slight variation on the previous scheme. Under the same hypothesis, we do the following steps.

- 1 We choose  $r \in \{1, \dots, p-1\}$  *randomly* and compute  $K_r = g^r$ .
- 2 We compute  $k_m$  such that

$$m = -sf(K_r) + k_m r \mod p.$$

- 3 We send to the recipient  $m$  and the pair  $(K_r, k_m)$ .
- 4 The recipient computes

$$u = mk_m^{-1} \mod p \quad v = f(K_r)k_m^{-1} \mod p,$$

and accepts the signature if the following equation holds

$$g^u \text{pub}^v = K_r.$$

**Exercise :** Check that the validity of the above signature does make sense! Is  $k_m$  necessarily invertible modulo  $p$ ? What should we do if not?

## The Claus Schnorr signature scheme (Part I)

This signature is in fact inspired from the Claus Schnorr variant of the ElGamal signature (1989). We will discuss in the cryptanalysis section the advantages of the different schemes. We give the algorithm below with precise parameters :

**Public parameters** : we choose a “large” prime number  $q$  and a large prime number  $p = aq + 1$ . We set  $G = (\mathbb{Z}/p\mathbb{Z})^\times$ . We choose  $g$  a generator of  $G$  and set  $g_a = g^a$ . We suppose given a (hash) function  $H : G \rightarrow \mathbb{Z}/q\mathbb{Z}$ .

**Secret key** : we choose a random  $s \in \mathbb{Z}/q\mathbb{Z}$

**Public key** :  $pub = g_a^s$



## The Claus Schnorr signature scheme (Part II)

Suppose that we want to sign a message  $m \in G$ .

- 1 We choose a random  $r \in (\mathbb{Z}/q\mathbb{Z})^\times$ .
- 2 We compute  $H_r = H(m || g_a^r)$  and  $h_m = H_r s + r$ . Then we send  $(H_r, h_m)$ .
- 3 The recipient accepts the signature if the following equation holds :

$$H_r = H(m || g_a^{h_m} \text{pub}^{-H_r}).$$

**Remark :** The key point is that, assuming a valid signature, we should have  $g^r = g_a^{h_m} \text{pub}^{-H_r}$ . The length of the signature is the size of the hash plus a number bounded by  $q$ .

**Exercise :** Check that the validity of the above signature does make sense !

## The Digital Signature Standard (DSS)

This is the concrete realisation for the DSA. It was published in 1994 by NIST. As function  $H$ , we will use SHA-1 (which gives a hash of 160 bits).

**Public parameters** : we choose a 160-bit prime number  $q$  and a large prime number  $p = aq + 1$  (say  $p$  of length at least 1024 bits). We set  $G = (\mathbb{Z}/p\mathbb{Z})^\times$ . We choose  $g$  a generator of  $G$  and set  $g_a = g^a$ . We have the SHA-1 function  $H$  which take value in  $\mathbb{Z}/q\mathbb{Z}$  and  $f : G \rightarrow \mathbb{Z}/q\mathbb{Z}$  the reduction mod  $q$  (it makes sense in the below scheme because  $q|(p-1)$ , but mathematically speaking  $f$  is not a function).

**Secret key** : we choose a random  $s \in \mathbb{Z}/q\mathbb{Z}$

**Public key** :  $pub = g_a^s$

## The Digital Signature Standard (DSS)

Suppose that we want to sign a message  $M$  (which could be unrelated to  $G$ ).

- 1 We choose a random  $r \in (\mathbb{Z}/q\mathbb{Z})^\times$  and compute  $k_r = f(g_a^r)$  (i.e.,  $k_r = g_a^r \bmod q$ ).

- 2 We compute

$$k_M = (sk_r + H(M))r^{-1}.$$

- 3 We send to the recipient  $M$  and the pair  $(k_r, k_M)$ .

- 4 The recipient computes

$$u = H(M)k_M^{-1}, \quad v = k_r k_M^{-1},$$

and accepts the signature if the following equation holds

$$f(g_a^u \text{pub}^v) = k_r.$$

## Remark on the DSS

Notice that it is almost exactly the DSA with  $k_r = f(K_r)$  (and  $K_r = g_a^r$ ), and  $m = H(M)$ , except that as we send  $k_r$  instead of  $K_r$ , we need to apply  $f$  to  $g_a^u \text{pub}^v$  (i.e., here to reduce modulo  $q$ ).

**Exercise :** Explain why in the DSS,  $k_M$  is likely an element of  $(\mathbb{Z}/q\mathbb{Z})^\times$  (i.e.,  $sk_r + H(M)$  is invertible modulo  $q$ ). Check the validity of the scheme. Implement the DSS. Test it with prime numbers of length 1024 and 2048 bits. Compare the speed for the different primes. Do a DSS2 version by using SHA-256 instead of SHA-1 (and then choosing  $q$  of size 256 bits). Compare the speed with DSS.

## Solving the DLP (the generic attacks...)

We have

$$x = g^d,$$

and knowing  $x$  and  $g$  we want to retrieve  $d$ .

The simplest (and most inefficient) method for solving the DLP is just to enumerate the different powers and stop when we get the good one.

# Complexity of DLP over prime finite fields

Algorithms	Complexity
Shanks (1971) / Rho method (Pollard/Brent, 1975-80)	$O(\sqrt{p})$
Index calculus (Adleman, 1972 for the initial idea)	$O(\exp(c + o(1)) \sqrt{\log(p) \log(\log(p))})$
FFS (NFS for DLP, 1993-2016)	$O(\exp((C + o(1)) \log(p)^{\frac{1}{3}} \log(\log(p))^{\frac{2}{3}}))$ , $C = (64/9)^{\frac{1}{3}}$

In **red** exponential complexity and in **blue** subexponential complexity.

**Conclusion :** The DLP over finite fields offer no specific advantages over RSA (both problems can be solved in subexponential time). It was hoped during the early days that the DLP (over finite fields) was harder to solve than RSA (at that time only exponential algorithms were known).

....Hence we have to look for new groups..... !

# Elliptic Curve Cryptography (ECC)

This new DLP based method was proposed, independently, in 1985 by Neal Koblitz (U. of Washington) and Victor S. Miller (from IBM at the time)



Neal Koblitz

Cryptography based on elliptic curves is mainly commercialised by Certicom, co-founded in 1985 by Scott Vanstone and Gordon Agnew, with lot of consulting from Alfred Menezes (all of them were at U. Waterloo at the time).



*The main company behind ECC...*

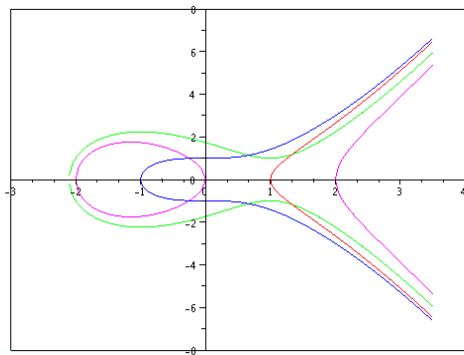


Scott Vanstone

👉 In 2003, NSA bought for \$25 millions of Certicom licensing rights for ECC.

In 2009, Certicom was bought by RIM (owner of BlackBerry cellphones).

# What on earth are Elliptic Curves?



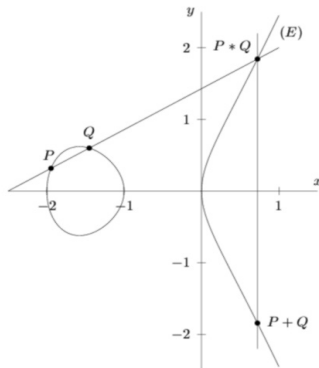
Examples of “elliptic curves” (as we can see them) in the real affine plane

They are particular curves “in the plane” enjoying some symmetry properties and endowed with an abelian group structure...



## Viewing the group law on Elliptic Curves

Over  $\mathbb{Q}$  or  $\mathbb{R}$ , you can geometrically visualize this group law.

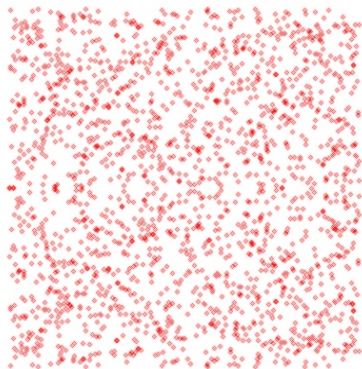


$$y^2 = x(x+1)(x+2).$$

If  $P$  and  $Q$  are two points of the “elliptic curve” and if  $P * Q$  designates the intersection of the straight line passing through  $P$  and  $Q$  with the curve  $E$ , then the point  $P + Q$  is obtained by taking the symmetrical opposite (relatively to the  $x$  axis) of the point  $P * Q$ .

## Elliptic curves over finite fields

But on a finite fields, the group law is less easy to see...



Some affine points of a curve over  $\mathbb{Z}/2003$ .

From the cryptographic viewpoint, one will look at curves over  $\mathbb{F}_{2^\nu}$  or over  $\mathbb{Z}/p$  with  $p$  of (binary) size  $\nu$  with  $\nu > 160$ .

👉 Now we need to go into the mathematical framework in order to see what exactly are “elliptic curves” and what are there

## Elliptic curves (EC) : generalities

If  $K$  is a field and  $\overline{K}$  is an algebraic closure of  $K$ , we will note  $K^\times$  its multiplicative group. We define an elliptic curve as being the set  $E$  of solutions in  $\mathbb{P}^2(\overline{K})$  (= set of lines in the three-dimensional vector space of  $K$ ) of the homogenous equation :

$$F(X, Y, Z) = Y^2Z + a_1XYZ + a_3YZ^2 - X^3 - a_2X^2Z - a_4XZ^2 - a_6Z^3 = 0,$$

where  $a_i \in K$  and the polynomial  $F$  being non-singular (aka regular), which signifies that the equations  $\frac{\partial F}{\partial X} = 0$ ,  $\frac{\partial F}{\partial Y} = 0$ ,  $\frac{\partial F}{\partial Z} = 0$  cannot cancel each other out simultaneously on the curve (reminder : the point  $(0, 0, 0)$  does not belong to the projective space!) This equation is called the extended form of Weierstrass. The coefficients  $a_i$  are often called “Tate coefficients” (or, rather, Tate notation).

This projective curve has a point at the infinity denoted  $O$  and which corresponds to the “projective point”  $(0, 1, 0)$ . If  $K \subset L \subset \overline{K}$  is an extension of  $K$ , we say that a point  $P \in E$  is  $L$ -rational if its coordinates are in  $L$ . We define  $E(L)$  as the set of  $L$ -rational points of  $E$ . If  $L = K$ , we generally simply speak of rational points of the curve. You can notice that  $O$  is  $L$ -rational for any extension of  $K$ . In practice, one often works with the affine form of the curve  $E$ , for which the affine Weierstrass equation is written :

$$F(X, Y) = Y^2 + a_1XY + a_3Y - X^3 - a_2X^2 - a_4X - a_6 = 0.$$

This is the projective equation with  $Z = 1$ . Again, we speak of rational points (these are essentially the same to which you have to add the point at infinity  $O$ ). We will change frequently from the projective representation to the affine representation, (and vice versa), or even to other practical representations for the calculations. If  $Z \neq 0$ , a projective point  $(X, Y, Z)$  corresponds to the affine point  $(X/Z, Y/Z)$ .

## Important point

If  $E$  is an EC over a field  $K$ , then  $E$  has an infinite number of points, even if  $K$  is finite!

But if  $K$  is finite and  $L \supset K$  also finite, then  $E(L)$  is a finite set (not easy to compute its order, but bounded by  $|L|^2$ ) and an abelian group. If  $L \subset L'$  then  $E(L) \subset E(L')$  and it is even a subgroup of  $E(L')$  and of course of  $E$ .

Writing that an EC over  $K$  is the set of point  $(x, y)$ , with  $x, y \in K$ , such that they are on the curve (plus the  $\infty$ ) is simply wrong!!

## Elliptic Curves in Mathematics?



Pierre de Fermat (1607/8(?) - 1665)

The famous Fermat's conjecture states that : if  $u, v, w$  are rational numbers,  $p > 2$  prime and  $u^p + v^p + w^p = 0$  holds, then  $uvw = 0$ . This conjecture was solved by A. Wiles (and R. Taylor) in 1995. One key ingredient in the prove was the study of the elliptic curve  $y^2 = x(x + u^p)(x - v^p)$  and showing that this curve *cannot be modular*. The Fermat's conjecture has been instrumental in the developpment of the arithmetic of elliptic curves in Mathematics.

## Main tools for classification of EC

One usually define the following constants (still using Tate notation) :

$$\begin{aligned}b_2 &= a_1^2 + 4a_2, & b_4 &= a_1a_3 + 2a_4, & b_6 &= a_3^2 + 4a_6, \\b_8 &= a_1^2a_6 + 4a_2a_6 - a_1a_3a_4 + a_2a_3^2 - a_4^2, \\c_4 &= b_2^2 - 24b_4, & c_6 &= -b_2^2 + 36b_2b_4 - 216b_6.\end{aligned}$$

One then define the discriminant of the curve, noted  $\Delta(E)$  (or simply  $\Delta$  if there is no ambiguity), as the quantity :

$$\Delta(E) = -b_2^2b_8 - 8b_4^3 - 27b_6^2 + 9b_2b_4b_6.$$

We notice that  $\Delta(E)$  is an element of  $K$  (possibly zero). When the characteristic of  $K$  is different from 2 and 3, then we simply have :

$$\Delta = \frac{c_4^3 - c_6^2}{1728}.$$

With  $1728 = 2^63^3$ , this explains that the rational fraction above has a pole in 2 and 3.

## Main tools for classification of EC (continued)

We then have the following result :

**Proposition** : An algebraic curve in Weierstrass form  $E/K$ , with  $\text{char}(K) \neq 2, 3$  is non-singular (or regular) if and only if  $\Delta(E) \neq 0$ .

We introduce the  $j$ -invariant of an EC by :

$$j(E) = \frac{c_4^3}{\Delta(E)}.$$

The  $j$ -invariant plays an important role, because it makes it possible to detect the  $K$ -isomorphisms. Let us introduce this concept.



## EC isomorphisms : the main concept for EC classification

**Definition :** If there are two EC given by their Weierstrass forms,  $E/K$  (in variables  $X, Y$ ) and  $E'/K$  (in variables  $X', Y'$ ). They are said to be isomorphic (over  $K$ ) if and only if there are  $r, s, t \in K$  and  $u \in K^\times$ , such that the change of variables :

$$X = u^2 X' + r, \quad Y = u^3 Y' + su^2 X' + t,$$

converts  $E$  into  $E'$ . Such a change of variables is called *admissible*.

Such an isomorphism also puts the rational points of  $E$  and  $E'$  into bijection. The isomorphism is defined over  $K$ , but you can also consider an isomorphism on an extension of  $K$ . Two non-isomorphic curves on  $K$  can become isomorphic on an extension of  $K$ .

We can write this isomorphism in a matrix form :

$$\begin{pmatrix} X \\ Y \end{pmatrix} \mapsto \begin{pmatrix} u^2 & 0 \\ u^2 s & u^3 \end{pmatrix} \begin{pmatrix} X' \\ Y' \end{pmatrix} + \begin{pmatrix} r \\ t \end{pmatrix} .$$

Since  $u$  is invertible, the matrix 2 defined in the transformation is invertible. The reverse transformation is given by :

$$\begin{pmatrix} X' \\ Y' \end{pmatrix} \mapsto \begin{pmatrix} u^{-2} & 0 \\ -u^{-2}s & u^{-3} \end{pmatrix} \begin{pmatrix} X \\ Y \end{pmatrix} - \begin{pmatrix} u^{-2}r \\ u^{-3}(t - rs) \end{pmatrix} .$$

Now we can speak of invariant of an EC. An invariant of an EC is a quantity invariant by isomorphisms of EC.

**Proposition :** the  $j$ -invariant is a  $K$ -invariant of the EC. In addition, if two EC  $E/K$  and  $E'/K$  have the same  $j$ -invariant then they are isomorphic over  $\overline{K}$ .

## Classification of EC

Using change of variables, we can classify the various types of EC by expressing them in their normal form.

Normal form of Weierstrass	$\Delta$	$j$
$\text{char}(K) \neq 2, 3, \quad Y^2 = X^3 + a_4X + a_6$	$-16(4a_4^3 + 27a_6^2)$	$1728 \frac{4a_4^3}{4a_4^3 + 27a_6^2}$
$\text{char}(K) = 3, j \neq 0, \quad Y^2 = X^3 + a_2X^2 + a_6$	$-a_2^3a_6$	$-\frac{a_2^3}{a_6}$
$\text{char}(K) = 3, j = 0, \quad Y^2 = X^3 + a_4X + a_6$	$-a_4^3$	0
$\text{char}(K) = 2, j \neq 0, \quad Y^2 + XY = X^3 + a_2X^2 + a_6$	$a_6$	$\frac{1}{a_6}$
$\text{char}(K) = 2, j = 0, \quad Y^2 + a_3Y = X^3 + a_4X + a_6$	$a_3^4$	0

## Group law on EC

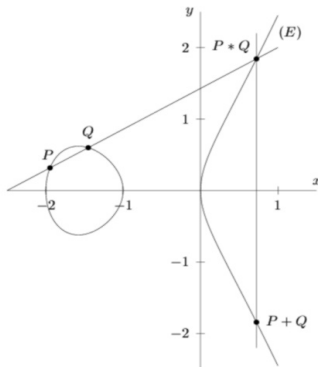
The advantage of EC is not in the explicit description of their normal forms, it is rather in the fact that you can put a group structure (commutative) on these curves which are particularly rich from the arithmetic viewpoint.

Let  $K \subset L \subset \overline{K}$  and  $E/K$  a general (affine) equation :

$$Y^2 + a_1XY + a_3Y = X^3 + a_2X^2 + a_4X + a_6.$$

Let  $P$  and  $Q$  two points of  $E(L)$ . set  $P = (x_1, y_1)$  and  $Q = (x_2, y_2)$ . We define a law of composition on  $E(L)$ , denoted  $+$ , in the following manner (as seen in the beginning) :

## Group law on EC (continued)



$$y^2 = x(x+1)(x+2).$$

$P$  and  $Q$  being two points of the elliptic curve and if  $P * Q$  designates the intersection of the straight line passing through  $P$  and  $Q$  with the curve  $E$ , then the point  $P + Q$  is obtained by taking the symmetrical opposite (relatively to the  $x$  axis) of the point  $P * Q$ .

## Group law on EC (continued)

The previous picture translates into the following formulae. Set  $R = P + Q$  with  $R = (x_3, y_3)$ .

If  $\text{char}(K) \neq 2$ , we set ;

$$x_3 = \begin{cases} \left( \frac{y_2 - y_1}{x_2 - x_1} \right)^2 + a_1 \left( \frac{y_2 - y_1}{x_2 - x_1} \right) - a_2 - x_1 - x_2 & \text{if } P \neq Q, \\ \left( \frac{3x^2 + 2a_2x + a_4 - a_1y}{2y + a_1x + a_3} \right)^2 + a_1 \left( \frac{3x^2 + 2a_2x + a_4 - a_1y}{2y + a_1x + a_3} \right) - a_2 - 2x & \text{if } P = Q, \end{cases}$$

and

$$y_3 = \begin{cases} \left( \frac{y_2 - y_1}{x_2 - x_1} \right) (x_1 - x_3) - y_1 - (a_1x_3 + a_3) & \text{if } P \neq Q, \\ \left( \frac{3x^2 + 2a_2x + a_4 - a_1y}{2y + a_1x + a_3} \right) (x - x_3) - y - (a_1x_3 + a_3) & \text{if } P = Q. \end{cases}$$

If  $\text{char}(K) = 2$  and  $j \neq 0$ , we set ;

$$x_3 = \begin{cases} \left( \frac{y_2 + y_1}{x_2 + x_1} \right)^2 + \left( \frac{y_2 + y_1}{x_2 + x_1} \right) + a_2 + x_1 + x_2 & \text{if } P \neq Q, \\ x^2 + \frac{a_6}{x^2} & \text{if } P = Q, \end{cases}$$

and

$$y_3 = \begin{cases} \left( \frac{y_2 + y_1}{x_2 + x_1} \right) (x_1 + x_3) + y_1 + x_3 & \text{if } P \neq Q, \\ \frac{x^2 + y}{x} x_3 + x^2 + x_3 & \text{if } P = Q. \end{cases}$$

If  $\text{char}(K) = 2$  and  $j = 0$  (**case very particular**), then the equation is of the form  $Y^2 + a_3Y = X^3 + a_4X + a_6$  and we set ;

$$x_3 = \begin{cases} \left( \frac{y_2 + y_1}{x_2 + x_1} \right)^2 + x_1 + x_2 & \text{if } P \neq Q, \\ \left( \frac{x^2 + a_4}{a_3} \right)^2 & \text{if } P = Q, \end{cases}$$

and

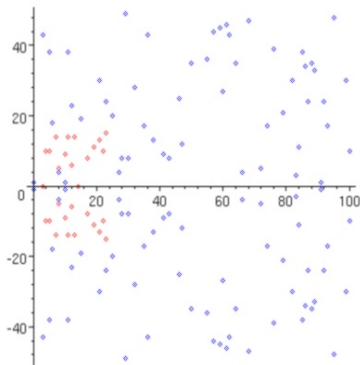
$$y_3 = \begin{cases} \left( \frac{y_2 + y_1}{x_2 + x_1} \right) (x_1 + x_3) + y_1 + a_3 & \text{if } P \neq Q, \\ \left( \frac{x^2 + a_4}{a_3} \right)^2 (x + x_3) + y + a_3 & \text{if } P = Q. \end{cases}$$

**Proposition :** The law  $+$  endows  $E(L) \cup \{\infty\}$  of a structure of abelian group with neutral element  $O = \infty$ .

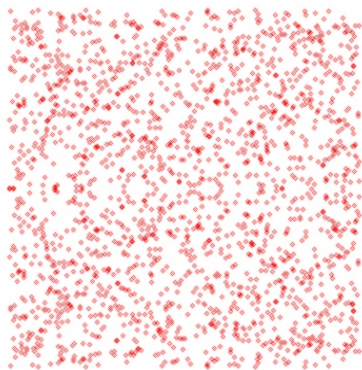


## EC over finite fields

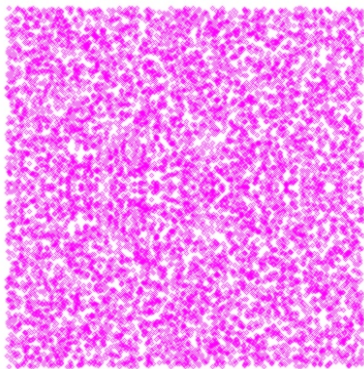
In the case of finite fields, an EC is a quite different object, even if one can still perceive its multiple symmetries. Here are a few examples with the equation  $y^2 = x^3 + x + 1$  seen in various finite fields :



in blue (rational points) over  $\mathbb{Z}/101$  and in red (rational points) over  $\mathbb{Z}/31$ .



over  $\mathbb{Z}/2003$ , but this picture is already wrong !.



over  $\mathbb{Z}/10007$  and it is even more wrong!.

From the cryptographic viewpoint, one will look at the curves on  $\mathbb{F}_{2^\nu}$  or over  $\mathbb{Z}/p$  with  $p$  of binary size  $\nu$  with  $\nu > 160$ .

## EC over finite fields with PARI/GP (Part 1)

The main functions for manipulating elliptic curves with PARI/GP are : `ellinit`, `elladd`, `ellsub`, `ellpow`, `ellorder`, `ellordinate`, `ellisoncurve`.

Let investigate these functions :

`E=ellinit(x)`; where  $x$  is the vector of Tate coefficients for the curve  $E$ , namely  $x = [a_1, a_2, a_3, a_4, a_6]$  defining the curve

$$Y^2 + a_1XY + a_3Y = X^3 + a_2X^2 + a_4X + a_6.$$

The function `ellinit` will initialise the curve  $E$  and return a large vector. The field where the coefficients  $a_i$  are defined will set the base field for the curve. If  $a_i \in K$  then  $E$  will be a curve over  $K$ . Furthermore the point `[0]` represents the infinity. We can access the data of  $E$  as member functions. For instance `E.j` will return the  $j$ -invariant, `E.disc` its discriminant, `E.b4` the Tate  $b_4$  coefficient, etc.

## EC over finite fields with PARI/GP (Part 1)

If  $P$  and  $Q$  are two points on the curve  $E$ , then

`R=elladd(E,P,Q)`; will compute  $R = P + Q$  on the curve  $E$ . In the same way `ellsub(E,P,Q)`; will compute  $P - Q$ .

The function `ellpow(E,P,n)`; with  $P$  on the curve  $E$  and  $n$  a positive integer will compute  $nP$ . For instance computing  $2P$  (i.e., doubling) can be done more efficiently than computing  $P + P$ .

The function `ellorder(E,P)` will compute the order of  $P$ . If  $P$  is non-torsion (*never the case when the curve is defined over a finite field*), the function return 0.

The function `ellordinate(E,x)` will return the  $y$ -coordinates of  $E$  corresponding to the ordinate  $x$ . If  $x$  is not the  $x$ -coordinate of an affine point of  $E$ , the function will return the infinity (i.e., `[0]`) or an empty set.

The function `ellisoncurve(E,P)` returns 1 (i.e., true) if  $P$  is on the curve  $E$  and 0 if not.

## EC for cryptography

From the cryptographic viewpoint, one will look at the curves on  $\mathbb{F}_{2^\nu}$  or over  $\mathbb{Z}/p$  with  $p$  of binary size  $\nu$  with  $\nu > 160$ .

**Main fact :** DLP is hard to solve on “random elliptic curves over  $\mathbb{F}_q$ ” and the best generic algorithm ( $\rho$ -Pollard) is in  $O(\sqrt{s})$  where  $s$  is the largest prime in the order of  $E(\mathbb{F}_q)$ , as we will see later.

## ECC from Microsoft viewpoint

In their WMA (i.e., the DRM in it), Microsoft is using an elliptic curve over  $\mathbb{F}_p$ , where  $p$  is a 160 bit prime number (given below). The equation of the curve is given in standard Weierstrass form (as required by most of the standards)  $y^2 = x^3 + ax + b$ ,  $a$  and  $b$  are coefficients (in  $\mathbb{F}_p$ ) given below.

All values are represented as packed binary values : in other words, a single value over  $\mathbb{F}_p$  is encoded simply as 20 bytes (stored in little endian order). A point on the elliptic curve is therefore a 40 bytes block, which consists of two 20 byte words representing the coordinates  $(x, y)$ .

Here are the parameters for the elliptic curve used in MS-DRM :

- $p = 89abcdef012345672718281831415926141424f7$
- $a = 37a5abccd277bce87632ff3d4780c009ebe41497$
- $b = 0dd8dabf725e2f3228e85f1ad78fdedf9328239e$
- $P_x = 8723947fd6a3a1e53510c07dba38daf0109fa120$
- $P_y = 445744911075522d8c3c5856d4ed7acda379936f$
- Order of the curve :  
 $89abcdef012345672716b26eec14904428c2a675$

But does it bring interesting performance ?



## ECC from Apple viewpoint

Their focus is on mobile device (i.e., small memory footprint) and use the Montgomery normal form

$$y^2 = x^3 + cx^2 + x, \quad c \neq \pm 2,$$

over  $\mathbb{F}_p$  with  $p = w^s - k$ ,  $w > 1$  and  $1 \leq k \leq w - 1$ , with  $k \equiv 1 \pmod{4}$ . The size of  $w$  is related to the CPU architecture and  $s$  can be seen as the “security parameter”. For instance, on a 32bit CPU with fast 16x16 bits operations, we can take  $w = 2^{16}$  (and  $s = 10$  as a minimal security level).

## ECC from Apple viewpoint (continued)

The curves are usually chosen such that their orders over  $\mathbb{F}_p$  are equal to  $w^s - j$  (for some  $j \leq w^{\frac{1+s}{2}}$ ), with  $c = 4$ . The goal is to have “fast encryption” (it is what they call “Small Memory Fast Elliptic Encryption”). In practice, the order is divided by 4 and most of the curves could be put into either Edwards or Jacobi quartic forms. The base point will be chosen of the type  $(x, 1)$ .

Example : as above, with  $k = 57$ ,  $x = 30$ , and  $j = 134739906578290596453580$ .

## General results on EC over a finite field

In what follows, we suppose given an EC  $E/K$  with  $K$  a field (which will be finite most of the time).

**Points of torsions** : If  $m \in \mathbb{N}$ , we note by  $[m]$  the “multiplication by  $m$ ” group morphism (this is an abuse of language!). This morphism is defined by :

$$\begin{aligned} [m] : E &\rightarrow E \\ P &\mapsto mP = \underbrace{P + \dots + P}_{m \text{ times}}. \end{aligned}$$

We set  $[-m]P = -(mP)$ . We say that *a point is of finite order (or torsion) if  $mP = O$  for some  $m$* . The kernel of the morphism  $[m]$  characterises the points of  $m$ -torsion of  $E$  (or of  $E(L)$  if you look at  $E$  on an extension of  $K$ ). We note this subgroup of  $E$  (or  $E(L)$ ) by  $E[m]$  (or  $E(L)[m]$ ). When  $\text{char}(K) = p$  and  $\gcd(m, p) = 1$ , we can show that there are  $m^2$  points of  $m$ -torsions.

## Structure theorem for EC over finite fields

The group of rational points of an EC over a finite field has a simple structure :

**Theorem** : “structure of  $E(\mathbb{F}_q)$ ”.

Set  $q = p^\nu$  with  $p$  prime. The group  $E(\mathbb{F}_q)$  is then finite and moreover :

$$E(\mathbb{F}_q) \cong \mathbb{Z}/d_1 \times \mathbb{Z}/d_2 ,$$

with  $d_1 | d_2$  et  $d_1 | (q - 1)$ .

## The Frobenius trace

Denote by  $a_q$  the integer  $a_q = q + 1 - \#E(\mathbb{F}_q)$ . The number  $a_q$  is called “trace of Frobenius of  $E$  at  $q$ ”. We also have a Frobenius morphism related to the trace.

$$\begin{aligned} \text{Frob}_q : E &\rightarrow E \\ (x, y, ) &\mapsto (x^q, y^q) \\ O &\mapsto O \end{aligned}$$

Then, we have

$$\text{Frob}_q^2 - [a_q]\text{Frob}_q + [q] = [0].$$

Written for a point  $P = (x, y)$ , the above equation means

$$(x^{q^2}, y^{q^2}) - [a_q](x^q, y^q) + [q](x, y) = O.$$

We have the following estimate for  $a_q$ .

**Theorem (Hasse, 1935) :** The Frobenius trace satisfies

$$|a_q| \leq 2\sqrt{q}.$$

## The Frobenius trace (continued)

If  $q$  is prime (and  $q \neq 2$ ), then there is an EC for each of the possible orders between  $q + 1 - 2\sqrt{q}$  and  $q + 1 + 2\sqrt{q}$ , we have a distribution that is practically uniform (consequence of results of analytical numbers theory). These results have major consequences with regard to the possible choices of “good curves” for cryptography. Note that if  $q = 2$ , the result is false and has little interest. What should be remembered is that :

**$\#E(\mathbb{F}_q)$  is of the same size than  $q$**

We can also deduce from the knowledge of the Frobenius trace the computation of  $\#E(\mathbb{F}_{q^n})$  (i.e., rational points over an extension of  $\mathbb{F}_q$ ).

Let  $\#E(F_q) = q + 1 - a_q$ . Write  $X^2 - a_qX + q = (X - \alpha)(X - \beta)$ .  
Then

$$\#E(\mathbb{F}_{q^n}) = q^n + 1 - (\alpha^n + \beta^n),$$

for all  $n \geq 1$ .

## A Family of curves

Let  $p$  be an odd prime and  $\lambda \in \mathbb{F}_p$  with  $\lambda \neq 0$ . Let  $a_p$  be the Frobenius trace at  $p$  of  $E$  (defined over  $\mathbb{F}_p$ ), where  $E$  is the elliptic curve

$$y^2 = x^3 - \lambda x.$$

- ① If  $p \equiv 3 \pmod{4}$ , then  $a_p = 0$  (*Notice that this does not depend of  $\lambda$* ).
- ② If  $p \equiv 1 \pmod{4}$ , write  $p = u^2 + v^2$  with  $u, v \in \mathbb{Z}$ ,  $v$  even and  $u + v \equiv 1 \pmod{4}$ . Then

$$a_p = \begin{cases} 2u & \text{if } \lambda \text{ is a fourth power,} \\ -2u & \text{if } \lambda \text{ is a square but not a fourth power,} \\ \pm 2v & \text{if } \lambda \text{ is not a square.} \end{cases}$$

Notice that  $u$  is uniquely determined and  $v$  is uniquely determined up to sign. This family is often interesting for testing “point counting algorithms”.

## More on the $m$ -torsion points

We can refine the structure theorem for the  $m$ -torsion points.

**Structure of  $m$ -torsion** : Let  $q = p^\nu$ ,  $E/\mathbb{F}_q$  an EC and  $m \in \mathbb{N}$ ,  $m \neq 0$ . Then

- If  $p$  does not divide  $m$ ,

$$E[m] \cong \mathbb{Z}/m \times \mathbb{Z}/m.$$

- If  $m = p^r$  then

$$E[p^r] \cong \mathbb{Z}/p^r \quad \text{or} \quad 0.$$



## EC over finite fields with PARI/GP (Part 2)

The function `ellap(E,p)` computes the Frobenius trace (at  $p$ ) of  $E$  (defined over  $\mathbb{F}_p$ ) Notice that  $p$  **must be a prime**. Not all the PARI/GP functions for elliptic curves work with non-prime finite fields (and in particular `ellap`).

Notice that the function uses Shanks/Mestre algorithm for small  $p$  (less than 30 digits) and Schoof/Elkies/Atkin for larger primes (up to 200 digits).

## Anomalous and supersingular curves

If  $E$  is an EC defined over  $\mathbb{F}_q$  with  $q = p^r$ . We say that  $E$  is **anomalous** (or “abnormal”) if  $t = 1$  (i.e.,  $\#E(\mathbb{F}_q) = q$ ). We say that  $E$  is **supersingular** if  $p \nmid t$  (e.g. When  $t = 0$ ). We have the following characterisation :

**Theorem :** An EC  $E/\mathbb{F}_q$  is supersingular if

- $p = 2$  or  $3$  and  $j(E) = 0$ .
- $p \geq 5$  and  $t = 0$ .

As we will see later, the anomalous and supersingular EC have properties that make them unusable (from the security viewpoint) for cryptography.

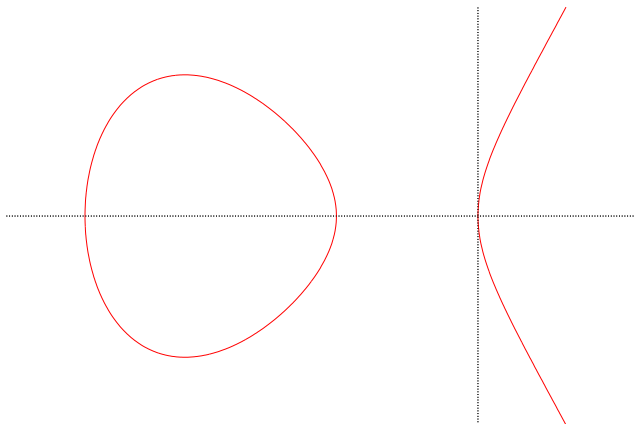
## Interest of EC for Cryptography ? (again !)

The DLP is difficult to solve over EC (if the base field is big enough) and only exponential algorithms are known at the present time.

## Solving the presentation and computational problems

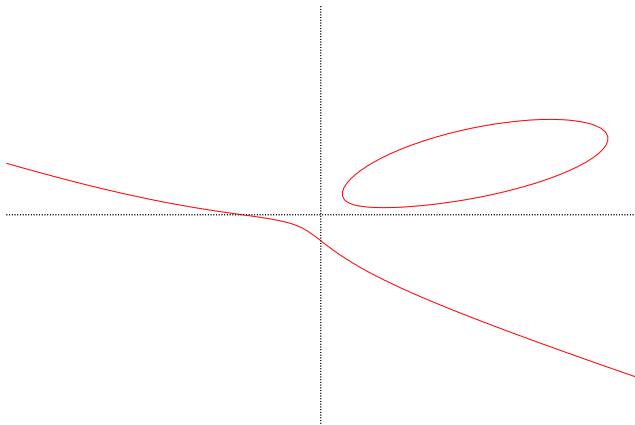
We have several ways to represent an elliptic curve ; either from implicit equations or parametric forms (or eventually “analytic forms” using theta functions), as cubic or as intersections of quadrics (or even fancier algebraic surfaces). Let us give a brief overview of the equations with some examples and then we will give more details on the Jacobi’s quartic representation.

# Weierstrass



$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$$

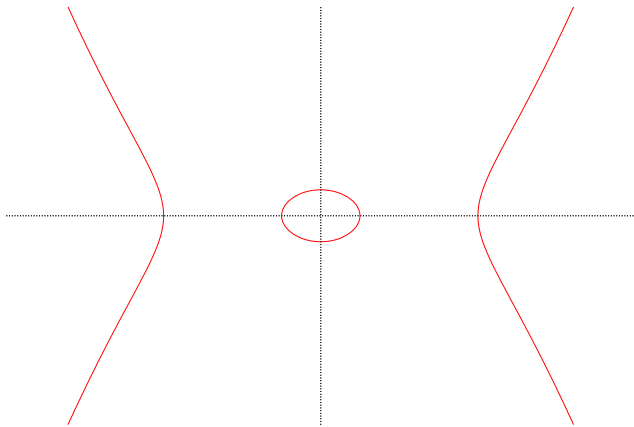
## Hesse's cubic



$$x^3 + y^3 + 1 = 3cxy$$

Chudnovsky & Chudnovsky, 1986 ; Joye & Quisquater, 2001 ; Smart, 2001

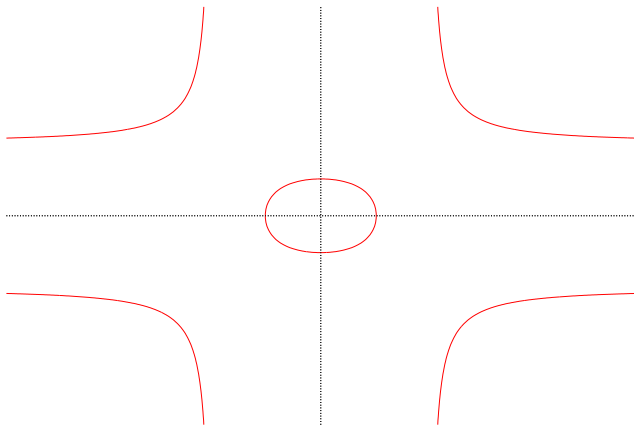
## Jacobi's quartic



$$y^2 = x^4 + 2ax^2 + 1$$

Chudnovsky & Chudnovsky, 1986 ; Liardet & Smart, 2001

# Edwards

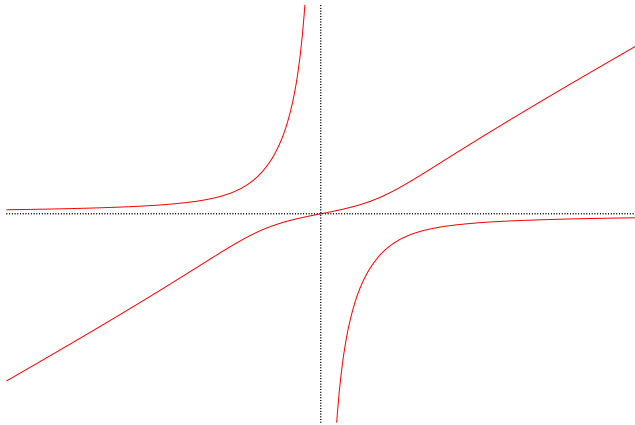


$$x^2 + y^2 = a + b(xy)^2$$

Bernstein & Lange, 2007



## Huff's cubic



$$ax(y^2 - 1) = by(x^2 - 1)$$

Joye, Tibouchi & Vergnaud, 2010

## Influence of coordinate systems on the performances

The use of primes of specific shapes for NIST curves is for efficiency purpose (fast reduction on computers with word of 32bits or 64bits, fast multiplication, fast squaring). The coordinates systems of the curves have also a huge impact on the computational cost. We can summarise some comparison in the following table (for non-binary curves) :

Coordinates system	Doubling	General addition
Affine	1I, 2M, 2S	1I, 2M, 1S
Projective	7M, 3S	12M, 2S
Jacobian	4M, 4S	12M, 4S
Chudnovsky	5M, 4S	11M, 3S
Edwards (2008)	3M, 4S	9M, 1S, 1D
Jacobi quartic (2009)	2M, 5S, 1D	7M, 3S, 1D
Huff model (Joye/Tibouchi/Vergnaud 2010)		12M
Mixed coordinates		
$J + A \rightarrow J$		8M, 3S
$J + C \rightarrow J$		11M, 3S
$C + A \rightarrow C$		8M, 3S

## Influence of coordinate systems on the performances

It should be pointed out that some models need constraints on the shape of the curve or equivalently on the group structure of  $E(K)$ . In particular, for some coordinates systems, we need to have either 2-torsion, or 4-torsion or even 3-torsion. It is for instance the case for Edwards, Jacobi and Huff.

## Jacobi's quartic in details

## Unifying addition formula for Jacobi's quartic

Let  $K$  be a field of characteristic different from 2. The generic equation of a Jacobi's quartic is

$$y^2 = z^2 + 2ax^2 + t^2, \quad x^2 = zt$$

Neutral :  $\varepsilon = (0 : 1 : 0 : 1)$  (the equations are homogeneous, we can write projectively if we want).

We have the following unified addition formula :

$$x_3 = (x_1y_2 + y_1x_2)(t_1t_2 - z_1z_2)$$

$$y_3 = (y_1y_2 + 2ax_1x_2)(z_1z_2 + t_1t_2) + 2x_1x_2(z_1t_2 + t_1z_2)$$

$$t_3 = (t_1t_2 - z_1z_2)^2 = (z_1z_2 + t_1t_2)^2 - (2x_1x_2)^2$$

$$z_3 = (x_1y_2 + y_1x_2)^2$$

$$= (z_1z_2 + t_1t_1)(z_1t_2 + t_1z_2) + 2x_1x_2(2ax_1x_2 + y_1y_1)$$

$$y_3 + z_3 = (z_1z_2 + 2x_1x_2 + t_1t_2)(y_1y_2 + 2ax_1x_2 + z_1t_2 + t_1z_2)$$

**Cost :  $7M + 3S + D_a$**  (Hisil/Wong/Carter/Dawson 2009).

## Changing coordinates from Weierstrass to Jacobi

Assuming we have an affine curve of equation :

$2\eta^2 = \xi(\xi - a - 1)(\xi - a + 1)$ . The change of coordinates is then given by

$$\begin{pmatrix} x \\ y \\ z \\ t \end{pmatrix} = \begin{pmatrix} 0 & 2 & 0 & 0 \\ 0 & 0 & 1 & 1 - a^2 \\ -2a & 0 & 1 & a^2 - 1 \\ 2 & 0 & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} \xi \\ \eta \\ \xi^2 \\ 1 \end{pmatrix}.$$

It is possible to apply such coordinate change to a general form  $y^2 = (x - r_1)(x - r_2)(x - r_3)$  as long as one of the  $r_i - r_j$  is a square.

## Summarizing the Jacobi's types over $\mathbb{F}_p$ ( $p \neq 2$ )

	$p = +1 \pmod{4}$	$p = -1 \pmod{4}$
$(\mathbb{Z}/2\mathbb{Z})^2$	$7\mathbf{M} + 3\mathbf{S} + \mathbf{D}_a + 2\mathbf{D}_b$	$7\mathbf{M} + 3\mathbf{S} + \mathbf{D}_a$
$\mathbb{Z}/2\mathbb{Z}$	$8\mathbf{M} + 3\mathbf{S} + \mathbf{D}_a + 2\mathbf{D}_b$	$8\mathbf{M} + 3\mathbf{S} + \mathbf{D}_a$

For  $p = +1 \pmod{4}$ ,  $b$  is in one of the four classes of  $\mathbb{F}_p^\times / (\mathbb{F}_p^\times)^4$ .  
We can also summarize the addition cost with respect to the other types of coordinates systems :

Coordinate system	Torsion constraints	Addition cost
Weierstrass		$12\mathbf{M} + 2\mathbf{S}$
Hesse	$\mathbb{Z}/3\mathbb{Z}$	$6\mathbf{M} + 6\mathbf{S}$
<b>Jacobi</b>	$\mathbb{Z}/2\mathbb{Z}$	$8\mathbf{M} + 3\mathbf{S} + \mathbf{D}_a + 2\mathbf{D}_b$
<b>Jacobi</b>	$(\mathbb{Z}/2\mathbb{Z})^2$	$7\mathbf{M} + 3\mathbf{S} + \mathbf{D}_a + 2\mathbf{D}_b$
Edwards	$\mathbb{Z}/4\mathbb{Z}$	$9\mathbf{M} + 1\mathbf{S}$
Huff	$\mathbb{Z}/2\mathbb{Z} \times \mathbb{Z}/4\mathbb{Z}$	$12\mathbf{M}$

Plut, 2009-2011 for the Jacobi's quartic cases

## ECC versions of the classical DLP protocols

As usual we will need to precise the domain parameters. In the case of ECC, there will be the base field  $F$  (which could be  $\mathbb{F}_{2^\nu}$ , or  $\mathbb{F}_p$  for a large prime  $p$  or eventually  $\mathbb{F}_{p^\nu}$ ), an elliptic curve  $E$  (usually given in affine Weierstrass form) over  $F$  and a point  $P \in E$  of large order.

The DLP for ECC (that we will call ECDLP), will be to recover the integer  $d$  from the public data  $Q = dP$  and  $P$ .

As group  $G$ , we will consider, supposing that  $P$  is of order  $n$ , the cyclic group

$$\langle P \rangle = \{\infty, P, 2P, \dots, (n-1)P\}.$$

The private key is an integer  $d$  that is selected uniformly at random from the interval  $[1, n-1]$  and the corresponding public key is  $Q = dP$ .



## Domain parameters for ECC

Domain parameters for an elliptic curve scheme

**Definition :** Domain parameters  $D = (q, FR, S, a, b, P, n, h)$  are comprised of :

- 1 The field order  $q$ .
- 2 An indication  $FR$  (field representation) of the representation used for the elements of  $\mathbb{F}_q$ .
- 3 A seed  $S$  if the curve was randomly generated.
- 4 Two coefficients  $a, b \in \mathbb{F}_q$  that define the equations of the elliptic curve  $E$  over  $\mathbb{F}_q$  (i.e.,  $y^2 = x^3 + ax + b$  for a prime field of  $\text{char} \neq 2, 3$  and  $y^2 + xy = x^3 + ax^2 + b$  for a binary field).
- 5 Two field elements  $x_P$  and  $y_P$  in  $\mathbb{F}_q$  that define a finite point  $P = (x_P, y_P) \in E(\mathbb{F}_q)$  in affine coordinates.  $P$  has prime order and is called the *base point*.
- 6 The order  $n$  of  $P$ .
- 7 The cofactor  $h = \#E(\mathbb{F}_q)/n$ .

## Choices for the domain parameters for ECC

The domain parameters are public and **should be chosen so that the ECDLP is resistant to all known attacks**. For realistic experimentation (close to real-life parameters), at minimum, one should have  $n > 2^{160}$  (recall that we assume  $n$  prime) and  $h$  should be small (usually  $h = 1, 2, 3, 4$ ). Furthermore we need to avoid anomalous curves (i.e.,  $\#E(\mathbb{F}_q) \neq q$ ) and supersingular curves ( $\#E(\mathbb{F}_q) \neq q + 1$ ). To avoid some attacks, one should ensure that  $n$  does not divide  $q^k - 1$  for all  $1 \leq k \leq C$  where  $C$  is large enough so that the classical DLP in  $\mathbb{F}_{q^C}^\times$  is intractable (for real-life parameters,  $C = 20$  is enough).

Furthermore **if  $q = p^\nu$  for some prime  $p$ , we should ensure that  $\nu > 11$ . In the binary case (and even in general), we should only use  $\mathbb{F}_{2^\nu}$  with  $\nu$  prime.**

**Remark :** the mathematics (including the mathematical concepts behind a scheme) should not depend of a specific representation  $FR$ , which is only here for efficiency reason (improvement of the arithmetics).

# Choices for domain parameters : NIST curves.

## NIST-recommended elliptic curves over prime fields

P-192 :

$$p = 2^{192} - 2^{64} - 1, a = -3, h = 1,$$

$b = 0x\ 64210519\ E59C80E7\ 0FA7E9AB\ 72243049\ FEB8DEEC\ C146B9B1$

$n = 0x\ FFFFFFFF\ FFFFFFFF\ FFFFFFFF\ 99DEF836\ 146BC9B1\ B4D22831$

$P_x = 0x\ 188da80eb03090f67cbf20eb43a18800f4ff0afd82ff1012$

$P_y = 0x\ 07192b95ffc8da78631011ed6b24cdd573f977a11e794811$

P-224 :

$$p = 2^{224} - 2^{96} + 1, a = -3, h = 1,$$

$b = 0x\ B4050A85\ 0C04B3AB\ F5413256\ 5044B0B7\ D7BFD8BA\ 270B3943\ 2355FFB4$

$n = 0x\ FFFFFFFF\ FFFFFFFF\ FFFFFFFF\ FFFF16A2\ E0B8F03E\ 13DD2945\ 5C5C2A3D$

$P_x = 0x\ b70e0cbd6bb4bf7f321390b94a03c1d356c21122343280d6115c1d21$

$P_y = 0x\ bd376388b5f723fb4c22dfe6cd4375a05a07476444d5819985007e34$

P-256 :

$$p = 2^{256} - 2^{224} + 2^{192} + 2^{96} - 1, a = -3, h = 1,$$

$b = 0x\ 5AC635D8\ AA3A93E7\ B3EBBD55\ 769886BC\ 651D06B0$

$CC53B0F6\ 3BCE3C3E\ 27D2604B$

$n = 0x\ FFFFFFFF\ 00000000\ FFFFFFFF\ FFFFFFFF\ BCE6FAAD$

$A7179E84\ F3B9CAC2\ FC632551$

$P_x = 0x\ 6b17d1f2e12c4247f8bce6e563a440f277037d812deb33a0f4a13945d898c296$

$P_y = 0x\ 4fe342e2fe1a7f9b8ee7eb4a7c0f9e162bce33576b315ececbb6406837bf51f5$

P-384 :

$$p = 2^{384} - 2^{128} - 2^{96} + 2^{32} - 1, a = -3, h = 1,$$

$b = 0x$  B3312FA7 E23EE7E4 988E056B E3F82D19 181D9C6E

FE814112 0314088F 5013875A C656398D 8A2ED19D 2A85C8ED D3EC2AEF

$n = 0x$  FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF

FFFFFFFF C7634D81 F4372DDF 581A0DB2 48B0A77A ECEC196A CCC52973

$P_x = 0x$  aa87ca22 be8b0537 8eb1c71e f320ad74 6e1d3b62 8ba79b98 59f741e0 82542a38

5502f25d bf55296c 3a545e38 72760ab7

$P_y = 0x$  3617de4a 96262c6f 5d9e98bf 9292dc29 f8f41dbd 289a147c e9da3113 b5f0b8c0

0a60b1ce 1d7e819d 7a431d7c 90ea0e5f

P-521 :

$$p = 2^{521} - 1, a = -3, h = 1,$$

$b = 0x$  00000051 953EB961 8E1C9A1F 929A21A0 B68540EE A2DA725B 99B315F3

B8B48991 8EF109E1 56193951 EC7E937B 1652C0BD 3BB1BF07 3573DF88

3D2C34F1 EF451FD4 6B503F00

$n = 0x$  000001FF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF

FFFFFFFF FFFFFFFF 51868783 BF2F966B 7FCC0148 F709A5D0 3BB5C9B8

899C47AE BB6FB71E 91386409

$P_x = 0x$  c6858e06 b70404e9 cd9e3ecb 662395b4 429c6481 39053fb5 21f828af 606b4d3d

baa14b5e 77efe759 28fe1dc1 27a2ffa8 de3348b3 c1856a42 9bf97e7e 31c2e5bd66

$P_y = 0x$  11839296 a789a3bc 0045c8a5 fb42c7d1 bd998f54 449579b4 46817afb d17273e6

62c97ee7 2995ef42 640c550 b9013fa d0761353 c7086a27 2c24088 be94769 fd16650

## Generating particular curves over $\mathbb{F}_q$

**Problem :** How to generate curves which are cryptographically suitable ?

One difficulty is the computation of the order of a curve  $E$ .

Our friends are :

- The characteristic equation of the Frobenius endomorphism

$$\Phi^2 - t\Phi + q/d = 0,$$

where  $t$  is the trace of the Frobenius endomorphism...

- ...and its relation with the order of the curve

$$\#E(\mathbb{F}_q) = q + 1 - t.$$

In fact, both aspects of the Frobenius morphism are connected :

Let  $\alpha$  and  $\beta$  the distinct (complex) solutions of the equation  $X^2 - tX + q$ , then

$$\#E(\mathbb{F}_{q^k}) = q^k + 1 - (\alpha^k + \beta^k).$$

We can then use this simple tool to construct curves of high order.

## Koblitz curves

The idea is the following :

- 1 Choose a random curve  $E$  over  $\mathbb{F}_p$  with  $p$  prime *and small*.
- 2 Count the order of  $E$  over  $\mathbb{F}_p$  (using the naive method is welcome here...)
- 3 Compute the roots  $\alpha$  and  $\beta$  of the characteristic equation of the Frobenius map of  $E$ .
- 4 Choose  $k$  such that  $p^k$  correspond to the desired security size.
- 5 Compute  $\alpha^k$  and  $\beta^k$  and deduce the order of the curve  $E(\mathbb{F}_{p^k})$

👉 **Problem of the method :** there is no certainty that the order of the curve over  $\mathbb{F}_{p^k}$  will have a subgroup of the desired security.

## Koblitz curves : first example

Consider the elliptic curve  $y^2 + xy = x^3 + 1$  over  $\mathbb{F}_2$  (is there other elliptic curves over  $\mathbb{F}_2$ ?). Its Frobenius trace is  $-1$  and  $\#E = 4$ .

The roots of  $X^2 + X + 2$  are  $\alpha = \frac{-1+\sqrt{7}}{2}$  and  $\beta = \frac{-1-\sqrt{7}}{2}$ . Suppose we want a more or less 160 bits, so take  $k = 163$  (because 163 is prime). Then

$$\alpha^k + \beta^k = 4845466632539410776804317,$$

and

$$\#E(\mathbb{F}_{2^{163}}) = 2^{163} + 1 - 4845466632539410776804317.$$

But this last number has too much small factors. However, for  $k = 223$  we have a subgroup of order of bitsize 169 bits, while for  $k = 233, 239$  we can achieve size of 232 and 238 bits respectively.

## Koblitz curves : second example

Consider the curve  $E : y^3 = x^2 - x + 1$  over  $\mathbb{F}_3$ . This curve has  $t = -3$  and  $\#E = 7$ . Its characteristic equation is  $X^2 + 3X + 3$  which has roots  $\alpha = \frac{-3+\sqrt{3}}{2}$  and  $\beta = \frac{-3-\sqrt{3}}{2}$ . Take  $k = 101$  as  $3^k$  is close to 160 bits. Then

$$\alpha^k + \beta^k =$$

again the order of  $E(\mathbb{F}_{3^k})$  has too much small factors. Looking for interesting values of  $k$  we can see  $k = 149$  for which we can find a subgroup of bitsize 220 bits while the field is of size 237 bits.

**Exercises :** find interesting Koblitz curves for  $p = 5, 7, 11$  and good finite prime extensions for which we can achieve 160 and 224 bits security without having a too large ground field.



## About the DLP for elliptic curves

While solving the DLP for general elliptic curves is hard, it could be easier for some particular curves and for high genus hyperelliptic curves. In the case of elliptic curves we have four attacks :

- The MOV reduction (dangerous mainly for supersingular curves)
- Reduction for anomalous curves
- Reduction for curves with automorphism

## The MOV reduction

Let  $E$  be an elliptic curve over  $\mathbb{F}_q$  and  $G$  a subgroup of  $E$  of prime order  $\ell$ . We define the **MOV degree** as the smallest natural number  $k$  such that  $\ell \mid (q^k - 1)$ . We then have the following result :

**Theorem (Menezes/Okamoto/Vanstone, 1993 and Frey/Rück, 1994) :** The DLP in  $G$  is reduced to a DLP in  $\mathbb{F}_{q^k}^\times$ .

It means that, in theory we can reduce to a sub-exponential algorithm, however, for a general curve,  $k$  is large. The main tool for the reduction is the Weil pairing (see NTC lectures).

☞ **But**, for supersingular curves  $k$  is small and then the reduction leads to an attack faster than the generic attack on the curve (i.e., Pollard Rho).

☞ **Counter-measure** : do not use supersingular curves !

## Reduction for anomalous curves

Recall that it is a curve with such that the order is equal to the order of the base field (assuming a prime base field).

☞ Then we have an isomorphism of  $E$  with  $\mathbb{F}_p$  (seen as additive group)

**Theorem (Smart, Satoh/Araki, Semaev) :** This isomorphism can be made explicit using the “elliptic  $p$ -adic logarithm”.


As a result we can solve “easily” the DLP on such type of curves.

☞ **Counter-measure :** do not use anomalous curves !

## Reduction for curves with automorphism

Let  $E$  be an elliptic curve over  $\mathbb{F}_q$ . Suppose that this curve admit an automorphism of order  $m$  supposed *effective* (i.e., fast to compute). Then, using the action of this automorphism on the curve we can adapt the Pollard algorithm to take advantage of this action :  
Indeed, *instead on working on elements of the curve, we work on the orbits w.r.t. the action of this automorphism. As a result we divide the space of elements by  $m$  and the running time is then divided by  $\sqrt{m}$ .* While this is not very compromising, it could nevertheless decrease a little the bitsize security of the scheme. The effectiveness of this attacks, depends of the order  $m$ . If  $m$  is very large, then it could be dangerous, but for small  $m$ , this provide no real advantage for cryptographic size curves.

**Example :** for Koblitz curves, the Frobenius map is an automorphism of degree equals to the one of the extension.

 **Counter-measure :** check the non-existence of automorphisms of high order. Random curves have rarely automorphism of high order.

## Arithmetics of ECC

The use of primes of specific shapes for NIST curves is for efficiency purpose (fast reduction on computers with word of 32bits or 64bits, fast multiplication, fast squaring). The coordinates systems of the curves have also a huge impact on the computational cost. We can summarise some comparison in the following table (for non-binary curves) :

Coordinates system	Doubling	General addition
Affine	1I, 2M, 2S	1I, 2M, 1S
Projective	7M, 3S	12M, 2S
Jacobian	4M, 4S	12M, 4S
Chudnovsky	5M, 4S	11M, 3S
Edwards (2008)	3M, 4S	9M, 1S, 1D
Jacobi quartic (2009)	2M, 5S, 1D	7M, 3S, 1D
Mixed coordinates		
$J + A \rightarrow J$		8M, 3S
$J + C \rightarrow J$		11M, 3S
$C + A \rightarrow C$		8M, 3S

As reference for ECC arithmetics, you can look at the book of Hankerson, Menezes and Vanstone ; *Guide to Elliptic Curve Cryptography*, Springer 2003..

## Key size comparisons between ECC and RSA

bits of security	Passwd size	RSA key size	ECC key size
80	13	1024	160
112	18	2048	224
128	21	3072	256
256	41	15360	512

Recall that for ECDLP, the best known general attack (assuming well chosen parameters) is in  $O(\sqrt{n})$  where  $n$  is the order of the base point.

The above table should be understood as follows : given a line with  $k$  bits of security, an RSA key size of  $m$  bits and an ECC key size of  $n$  bits, it means that we need  $2^k$  operations in order to recover the RSA key of  $m$  bits or the ECC key of  $n$  bits. Passwd sizes (in characters) assume that cost of passwd cipher is neglectible.

## About point compression for ECC

*point compression* is a standard trick which reduces the storage requirement for points on elliptic curves.

Recall that a non-infinite point on an elliptic curve  $E$  is given by a couple  $(x, y)$ . For simplicity, assume we work over  $\mathbb{Z}/p\mathbb{Z}$  (and  $p > 3$ ) and  $E$  has for equation (in Weierstrass form) :

$y^2 = x^3 + ax + b$ . Thus, given a value  $x$ , and unless  $x^3 + ax + b = 0$ , there are two possible values for  $y$ , which are “negatives” of each other modulo  $p$ .

Since  $p$  is odd, one of the possible values  $y$  is even and the other is odd. Thus we just need to know the parity bit of  $y$ . This reduces the storage by (almost) 50% at the expense of additional computations (a square root in a field). The operation of point compression is often expressed as a function

$$\text{PointCompress} : E \setminus \{\infty\} \rightarrow \mathbb{F}_p \times \mathbb{F}_2$$

defined by  $\text{PointCompress}(P) = (x, \tilde{y} \bmod 2)$ , with  $P \in E \setminus \{\infty\}$  and  $\tilde{y}$  denotes the unique lift in  $\mathbb{Z}$  of  $y$  such that  $0 < \tilde{y} < p$ .

## Decompression algorithm

The inverse operation, point decompression, reconstructs the elliptic curve point and can be implemented as follows :

- Given  $x$  and  $i$  (the parity bit), compute  $z = x^3 + ax + b \bmod p$
- **If**  $z$  is a quadratic non-residue modulo  $p$  **then** return “Failure” (i.e.,  $z$  is not a square mod  $p$ )
- **Else**  $y = \sqrt{z} \bmod p$ ; **If**  $\tilde{y} = i \bmod 2$  **then** return  $(x, y)$  **else** return  $(x, p - y)$ .



## ElGamal for ECC

We can adapt the basic ElGamal scheme for EC. With the above parameters, we have the following encryption scheme.

- 1 Represent the message  $m$  as a point  $M$  in  $E(\mathbb{F}_q)$ .
- 2 Choose *randomly*  $r \in [1, n - 1]$ .
- 3 Compute  $k_r = rP$ .
- 4 Compute  $C = M + rQ$ .
- 5 Return  $(k_r, C)$

and its decryption scheme, assuming we get the pair  $(k_r, C)$ .

- 1 Compute  $M = C - dk_r$  and extract  $m$  from  $M$ .
- 2 Return  $m$ .

## ECDSA : the new standard for signature

This is analogous of DSA for ECC standardised in PKCS #11, ANSI X9.62, FIPS 186-2, IEEE 1363-2000, ISO/IEC 15946-2, TLS 1.1 and others. Let see the ECDSA signature generation for a message  $m$  (assuming a cryptographic hash function  $H$  with bitlength outputs smaller than  $n$  and also a conversion function from coordinates of the curves to integers).

- 1 Choose *randomly*  $r \in [1, n - 1]$ .
- 2 Compute  $rP = (x, y)$  and convert  $x$  to an integer  $\tilde{x}$ .
- 3 Compute  $k_r = \tilde{x} \bmod n$ . If  $k_r == 0$  then go to step (1).
- 4 Compute  $h_m = H(m)$ .
- 5 Compute  $k_m = r^{-1}(h_m + dk_r) \bmod n$ . If  $k_m == 0$  then go to step (1).
- 6 Return  $(k_r, k_m)$ .

## ECDSA : acceptance of signature

Given a ECDSA signature generation we proceed as follows.

- 1 Verify that  $k_r$  and  $k_m$  are integers in the interval  $[1, n - 1]$ . If it fails, we reject the signature.
- 2 Compute  $h_m = H(m)$ .
- 3 Compute  $w = k_m^{-1} \bmod n$ .
- 4 Compute  $u = h_m w \bmod n$  and  $v = k_r w \bmod n$ .
- 5 Compute  $X = uP + vQ$ .
- 6 If  $X = \infty$  then reject the signature.
- 7 Convert the  $x$ -coordinate of  $X$  to an integer  $\tilde{x}$  and compute  $k' = \tilde{x} \bmod n$ .
- 8 If  $k' = k_r$  then we accept the signature else we reject the signature.

## ECIES : An encryption scheme

Designed by Michel Abdalla, Mihir Bellare and Phillip Rogaway in 1998. The scheme is as efficient as ElGamal encryption, but has stronger security properties.

It is a Diffie/Hellman based scheme that combines a symmetric encryption method, a message authentication code, and a hash function, in addition to number-theoretic operations, in a way which is intended to provide security against chosen-ciphertext attacks. The proofs of security are based on the assumption that the underlying symmetric primitives are secure and on appropriate assumptions about the Diffie/Hellman problem (originally, we can use it for any group for which the DH problem is hard to solve).

## ECIES (continued)

It has been standardised in ANSI X9.63, ISO/IEC 15946-3 and IEEE P1363.

In ECIES, a Diffie/Hellman shared secret is used to derive two symmetric keys  $k_1$  and  $k_2$ . Key  $k_1$  is used to encrypt the plaintext using a symmetric-key cipher, while key  $k_2$  is used to authenticate the resulting ciphertext.

The authentication guards against chosen-ciphertext attacks since the adversary cannot generate valid ciphertexts.

In ECIES, the following cryptographic primitives are used :

- 1 A key derivation function  $KDF$  that is constructed from a hash function  $H$ .
- 2  $Enc$  is the encryption function for a symmetric-key encryption scheme such as AES, and  $Dec$  is the decryption function.
- 3  $MAC$  is a message authentication code algorithm (for instance HMAC).

## ECIES : Encryption

Assuming domain parameters  $D = (q, FR, S, a, b, P, n, h)$ , a public key  $Q$  and a plaintext  $m$ .

- 1 Choose randomly  $k \in [1, n - 1]$
- 2 Compute  $R = kP$  and  $Z = hkQ$ . If  $Z == \infty$  then go to step (1).
- 3 Compute  $(k_1, k_2) = KDF(x_Z, R)$ , where  $x_Z$  is the  $x$ -coordinate of  $Z$ .
- 4 Compute  $C = Enc_{k_1}(m)$  and  $tag = MAC_{k_2}(C)$ .
- 5 Return  $(R, C, tag)$

## ECIES : Decryption

Assuming domain parameters

$D = (q, FR, S, a, b, P, n, h)$ , a private key  $d$  and a ciphertext (output of ECIES Encryption)  $(R, C, tag)$ .

- 1 Verify that  $R$  is valid public key (i.e., check that  $R \neq \infty$ ,  $R$  is properly represented and  $R$  is on the curve!). If it fails reject the ciphertext.
- 2 Compute  $Z = hdR$ . If  $Z = \infty$  then reject the ciphertext.
- 3 Compute  $(k_1, k_2) = KDF(x_Z, R)$ , where  $x_Z$  is the x-coordinate of  $Z$ .
- 4 Compute  $t = MAC_{k_2}(C)$ . If  $t \neq tag$  then reject the ciphertext.
- 5 Compute  $m = Dec_{k_1}(C)$ .
- 6 Return  $m$ .

## Validity of the ECIES scheme

If the ciphertext  $(R, C, tag)$  is valid then

$$hdR = hd(kP) = hk(dP) = hkQ.$$

Hence we will get the same keys  $(k_1, k_2)$ .

**Remark :** Due to the fact that we use the  $x$ -coordinate, it is important to include  $R$  in the input of KDF. If not,  $(R, C, tag)$  and  $(-R, C, tag)$  are two valid outputs with the same plaintext and ciphertext.



## ECMQV : A key establishment protocol

The purpose of a key establishment protocol is to provide two or more entities communication over an open network with a shared secret. A *key transport* protocol is a key establishment protocol where one entity creates the secret key and securely transfers it to others. ECIES can be considered a two-party key transport protocol when the plaintext message consists of the secret key.

A *key agreement* protocol is a key establishment protocol where all participating entities contribute to the information which is used to derive the shared secret key.

ECMQV is a three-pass key agreement protocol that has been standardised in ANSI X9.63, ISO/IEC 15946-3 and IEEE P1363. It is based on Diffie/Hellman and provides authentication (it is resilient against man-in-the-middle attack). It was designed by Alfred Menezes, Minghua Qu, and Scott Vanstone in 1995 and modified by Laurie Law and Jerry Solinas in 1998.

## ECMQV : A key establishment protocol

At the difference of most of the protocols that we have seen, ECMQV is patented (by Certicom). It is also the fact of several others key establishment protocol robust against man-in-the-middle attack. The protocol is in fact designed to work with arbitrary groups. There are several variants of the protocol, but the main variant is the three-pass version.

If  $R$  is a point of a curve, we will denote by  $\tilde{R}$  the integer  $(\tilde{x} \bmod 2^{\lceil f/2 \rceil}) + 2^{\lceil f/2 \rceil}$  where  $\tilde{x}$  is the integer representation of the x-coordinate of  $R$  and  $f = \lfloor \log_2(n) \rfloor + 1$  is the bitlength of  $n$

**Remark :** this is a tool in order to speed up the scheme. It has not yet been shown that this weaker the scheme.

## ECMQV key agreement

Assuming domain parameters  $D = (q, FR, S, a, b, P, n, h)$ , key pair  $(Q_A, d_A)$  of entity  $A$ , key pair  $(Q_B, d_B)$  for  $B$ , a key derivation KDF and a MAC function. If any verification in the protocol fails, then the protocol run is terminated with failure.

**Step 1 :**  $A$  randomly chooses  $k_A \in [1, n - 1]$ , computes  $R_A = k_A P$  and sends  $R_A$  to  $B$  (and eventually an extra data  $A$ ).

**Step 2 :**  $B$  does the following :

- 1 Performs a validation of  $R_A$ .
- 2 Randomly chooses  $k_B \in [1, n - 1]$  and computes  $R_B = k_B P$
- 3 Computes  $s_B = (k_B + \tilde{R}_B d_B) \bmod n$  and  $Z = hs_B(R_A + \tilde{R}_A Q_A)$ , and verifies that  $Z \neq \infty$ .
- 4 Computes  $(k_1, k_2) = KDF(x_Z)$ , where  $x_Z$  is the  $x$ -coordinate of  $Z$ .
- 5 Computes  $t_B = MAC_{k_1}(2, B, A, R_B, R_A)$ .
- 6 Sends  $R_B, t_B$  (and eventually an extra data  $B$ ) to  $A$ .

## ECMQV (continued)

**Step 3 :** A does the following :

- 1 Perform validity of  $R_B$ .
- 2 Computes  $s_A = (k_A + \tilde{R}_A d_A) \bmod n$ ,  $Z = hs_A(R_B + \tilde{R}_B Q_B)$  and verify that  $Z \neq \infty$ .
- 3 Computes  $(k_1, k_2) = KDF(x_Z)$ , where  $x_Z$  is the x-coordinate of  $Z$ .
- 4 Computes  $t = MAC_{k_1}(2, B, A, R_B, R_A)$  and verifies that  $t == t_B$ .
- 5 Computes  $t_A = MAC_{k_1}(3, A, B, R_A, R_B)$  and sends  $t_A$  to  $B$ .

**Step 4 :** B computes  $t = MAC_{k_1}(3, A, B, R_A, R_B)$  and verifies that  $t == t_A$ .

**Step 5 :** The session key is  $K = k_2$ .

## Some comments on ECMQV

The extra data for  $A$  and  $B$  could be certificates associated to the key pairs. The numbering 2 and 3 during the passes are here to provide specific tag in order to distinguish the steps and the authentication between  $A$  and  $B$ .

The quantity

$$s_A = (k_A + \tilde{R}_A d_A) \mod n$$

serves as an *implicit signature* for  $A$ 's ephemeral public key  $R_A$  (idem for  $B$ ).

It is a signature in the sense that the “only” person who can compute it is  $A$  and is implicit because  $B$  (idem for  $A$ ) indirectly verifies its validity by using

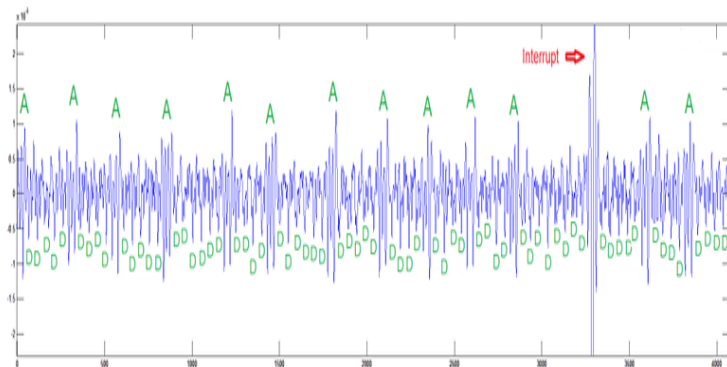
$$s_A P = R_A + \tilde{R}_A Q_A.$$

**Exercise :** Check the validity of the ECMQV scheme.

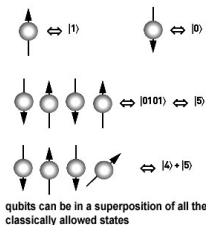
👉 We will see later, other variant of MQV schemes.

## Physical attack on ECC software implementation

Genkin, Pachmanov, Pipman, Tromer and Yarom have shown (2016) that modern cryptographic software on mobile phones, implementing the ECDSA digital signature algorithm, may inadvertently expose its secret keys through physical side channels (with even a low cost attack) :



# RSA, Classical DLP, ECC and Quantum computers



In 1994, Peter Shor gave algorithms for the factorization of integers and for solving the DLP running in quantum polynomial time and requiring a “big enough” quantum computer. Those algorithms have been improved during the past years in both space requirement and running time.

In 2008, it has been shown by Proos and Zacka that we can solve ECDLP in Quantum polynomial time. Furthermore, they have shown that we can solve it with less qubits and faster than the (quantum) factorization of an equivalent RSA modulus. **Conclusion : A good public key cryptosystem for the postquantum era is still to be discovered...**