Elliptic Curve Cryptography

Luca De Feo

École Normale Supérieure & École Polytechnique

Università di Pisa, April 18, 2007

http://www.eleves.ens.fr/home/defeo

Plan

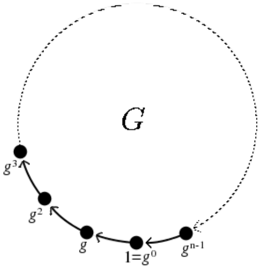
The Discrete Logarithm Problem

Cyclic groups

- A cyclic group $(G, *)$, a generator g of G of order n
- G is isomorphic to $\mathbb{Z}/n\mathbb{Z}$ via the bijection

$$\exp_g : x \mapsto g^x$$

- The function $\exp_g$ is easy to compute $(O(\log n))$



The Discrete Logarithm Problem

Cyclic groups

- A cyclic group $(G, *)$, a generator g of G of order n
- G is isomorphic to $\mathbb{Z}/n\mathbb{Z}$ via the bijection

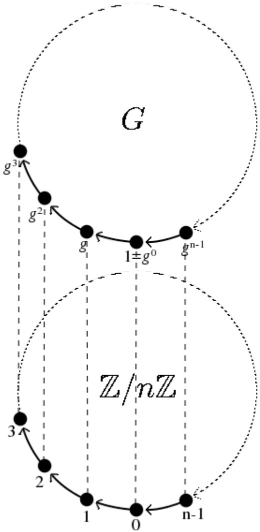
$$\exp_g \ : \ x \mapsto g^x$$

- The function $\exp_g$ is easy to compute ($O(\log n)$)

The discrete logarithm

- The inverse to the function $\exp_g$ is called discrete logarithm, noted $\log_g$:

$$\log_g \ : \ g^x \mapsto x$$



The Discrete Logarithm Problem

The Discrete Logarithm Problem (DLP)

- Computing the function $\log_g$ may be very easy... e.g.: $G = \mathbb{Z}/n\mathbb{Z}$

The Discrete Logarithm Problem

The Discrete Logarithm Problem (DLP)

- Computing the function $\log_g$ may be very easy... e.g.: $G = \mathbb{Z}/n\mathbb{Z}$
- ...or very hard e.g.: $G = \mathbb{Z}/n\mathbb{Z}^*$

The Discrete Logarithm Problem

The Discrete Logarithm Problem (DLP)

- Computing the function $\log_g$ may be very easy... e.g.: $G = \mathbb{Z}/n\mathbb{Z}$
- ...or very hard e.g.: $G = \mathbb{Z}/n\mathbb{Z}^*$
- an example : $G = \mathbb{Z}/23\mathbb{Z}^*$, $g = 5$. What's $\log_5 10$?

The Discrete Logarithm Problem

The Discrete Logarithm Problem (DLP)

- Computing the function $\log_g$ may be very easy... e.g.: $G = \mathbb{Z}/n\mathbb{Z}$
- ...or very hard e.g.: $G = \mathbb{Z}/n\mathbb{Z}^*$
- an example : $G = \mathbb{Z}/23\mathbb{Z}^*$, $g = 5$. What's $\log_5 10$?
  answer : 3, in fact $5^3 = 125 = 5 * 23 + 10$!

The Discrete Logarithm Problem

The Discrete Logarithm Problem (DLP)

- Computing the function $\log_g$ may be very easy... e.g.: $G = \mathbb{Z}/n\mathbb{Z}$
- ...or very hard e.g.: $G = \mathbb{Z}/n\mathbb{Z}^*$
- an example : $G = \mathbb{Z}/23\mathbb{Z}^*$, $g = 5$. What's $\log_5 10$?
  answer : 3, in fact $5^3 = 125 = 5 * 23 + 10$!

Algorithms

- The most efficient algorithms for a general group G are *BSGS* and *Pollard's Rho*. They both need $O(\sqrt{n})$ operations in the group
- *Pohlig and Hellman* improve this result by solving the DLP in the subgroups of G having prime order p s.t. $p|n$
- Thus we demand the order of G to be prime

The Discrete Logarithm Problem

The Discrete Logarithm Problem (DLP)

- Computing the function $\log_g$ may be very easy... e.g.: $G = \mathbb{Z}/n\mathbb{Z}$
- ...or very hard e.g.: $G = \mathbb{Z}/n\mathbb{Z}^*$
- an example : $G = \mathbb{Z}/23\mathbb{Z}^*$, $g = 5$. What's $\log_5 10$?
  answer : 3, in fact $5^3 = 125 = 5 * 23 + 10$!

Algorithms

- The most efficient algorithms for a general group G are *BSGS* and *Pollard's Rho*. They both need $O(\sqrt{n})$ operations in the group
- *Pohlig and Hellman* improve this result by solving the DLP in the subgroups of G having prime order p s.t. $p|n$
- Thus we demand the order of G to be prime
- The most efficient algorithm for the group $\mathbb{Z}/n\mathbb{Z}^*$ is the *Number Field Sieve*. It needs $L_n(1/3)$ operations in the group

Diffie-Hellman key exchange

Diffie-Hellman key exchange

Diffie-Hellman key exchange

Diffie-Hellman key exchange

A group G of prime order p. A generator g of G.

Diffie-Hellman key exchange

A group G of prime order p. A generator g of G.

- chooses $a \in \mathbb{Z}/p\mathbb{Z}$ at random

Diffie-Hellman key exchange

A group G of prime order p. A generator g of G.

- chooses $a \in \mathbb{Z}/p\mathbb{Z}$ at random
- computes g^a

Diffie-Hellman key exchange

A group G of prime order p. A generator g of G.

- chooses $a \in \mathbb{Z}/p\mathbb{Z}$ at random
- computes g^a

- chooses $b \in \mathbb{Z}/p\mathbb{Z}$ at random

Diffie-Hellman key exchange

A group G of prime order p. A generator g of G.

- chooses $a \in \mathbb{Z}/p\mathbb{Z}$ at random
- computes g^a

- chooses $b \in \mathbb{Z}/p\mathbb{Z}$ at random
- computes g^b

Diffie-Hellman key exchange

A group G of prime order p. A generator g of G.

- chooses $a \in \mathbb{Z}/p\mathbb{Z}$ at random
- computes g^a

- chooses $b \in \mathbb{Z}/p\mathbb{Z}$ at random
- computes g^b

$$\xrightarrow{\qquad g^a \qquad}$$

Diffie-Hellman key exchange

A group G of prime order p. A generator g of G.

- chooses $a \in \mathbb{Z}/p\mathbb{Z}$ at random
- computes g^a

- chooses $b \in \mathbb{Z}/p\mathbb{Z}$ at random
- computes g^b

$$\xrightarrow{\quad g^a \quad}$$

$$\xleftarrow{\quad g^b \quad}$$

Diffie-Hellman key exchange

A group G of prime order p. A generator g of G.

- chooses $a \in \mathbb{Z}/p\mathbb{Z}$ at random
- computes g^a

- chooses $b \in \mathbb{Z}/p\mathbb{Z}$ at random
- computes g^b

$$\xrightarrow{\quad g^a \quad}$$

$$\xleftarrow{\quad g^b \quad}$$

- computes $K_{ab} = \left(g^b\right)^a$

Diffie-Hellman key exchange

A group G of prime order p. A generator g of G.

- chooses $a \in \mathbb{Z}/p\mathbb{Z}$ at random
- computes g^a

- chooses $b \in \mathbb{Z}/p\mathbb{Z}$ at random
- computes g^b

$$\xrightarrow{\quad g^a \quad}$$
$$\xleftarrow{\quad g^b \quad}$$

- computes $K_{ab} = \left(g^b\right)^a$

- computes $K_{ab} = \left(g^a\right)^b$

The Diffie-Helman Problems

The security of the DH key exchange

- An eavesdropper sees the values g^a and g^b
- It has to compute the value $K_{ab} = g^{ab}$
- The hardness of the computation is expressed via two problems believed to be difficult

The Diffie-Helman Problems

Decisional Diffie-Hellman Problem (DDH)

Given a group G, a generator g for G, three random elements g^a, g^b and g^c, distinguish with a non-negligible probability the triples

$$(g^a, g^b, g^{ab}) \quad \text{and} \quad (g^a, g^b, g^c)\ .$$

Computational Diffie-Hellman Problem (CDH)

Given a group G, a generator g for G, two random elements g^a and g^b, compute g^{ab}.

The Diffie-Helman Problems

Decisional Diffie-Hellman Problem (DDH)

Given a group G, a generator g for G, three random elements g^a, g^b and g^c, distinguish with a non-negligible probability the triples

$$(g^a, g^b, g^{ab}) \quad \text{and} \quad (g^a, g^b, g^c) \ .$$

Computational Diffie-Hellman Problem (CDH)

Given a group G, a generator g for G, two random elements g^a and g^b, compute g^{ab}.

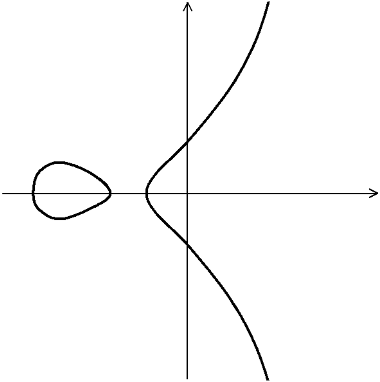
DLP and DH

- Clearly, if one can solve DLP, it can solve CDH and DDH as well
- The other direction is believed to be "almost true"

Plan

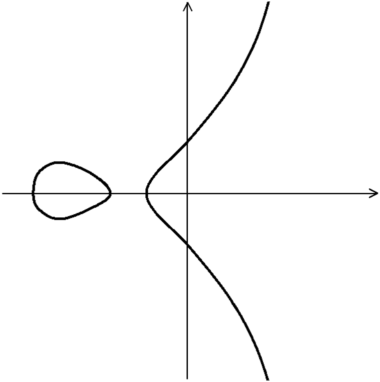
Elliptic curves

"An algebraic curve of genus 1"



Elliptic curves

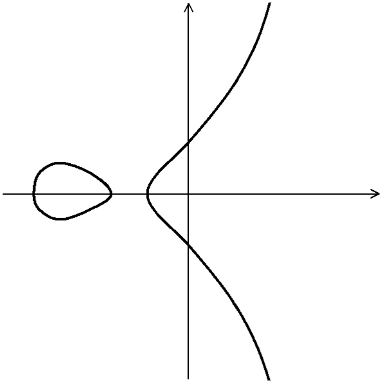
$$\mathbf{E} \;:\; Y^2 + a_1 XY + a_3 Y = X^3 + a_2 X^2 + a_4 X + a_6$$



X and Y taking values in a field K, $a_1, a_2, a_3, a_4, a_6 \in K$

Elliptic curves

$$\mathbf{E} : Y^2 = X^3 + a_4 X + a_6$$



assuming $\text{char}(K) \neq 2, 3$

Elliptic curves

$$\mathbf{E} \; : \; Y^2 = X^3 + a_4 X + a_6$$

We define

- The *discriminant* $\Delta = -64a_4^3 - 1728a_6^2$
- The j*-invariant* $j(E) = -\frac{-1728(4a_4)^3}{\Delta}$

Elliptic curves

$$\mathbf{E} \ : \ Y^2 = X^3 + a_4 X + a_6$$

We define

- The *discriminant* $\Delta = -64a_4^3 - 1728a_6^2$
- The *j-invariant* $j(E) = -\frac{-1728(4a_4)^3}{\Delta}$

Isomorphic curves have the same j-invariant

Elliptic curves

$$\mathbf{E} : Y^2 = X^3 + a_4 X + a_6$$

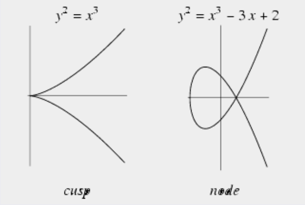
We define

- The *discriminant* $\Delta = -64a_4^3 - 1728a_6^2$
- The j-*invariant* $j(E) = -\frac{-1728(4a_4)^3}{\Delta}$

Isomorphic curves have the same j-invariant

We demand

- The curve to be *smooth* $\Leftrightarrow \Delta \neq 0$



The group law (the jacobian in one slide !)

Divisors

- We can define a formal group Div(E) over the points of the curve E
- We work in the projective space $\mathbb{P}^2(K)$: we add a point at infinity $\mathcal{O}$.
- The point at infinity acts as a zero for the group

The group law (the jacobian in one slide !)

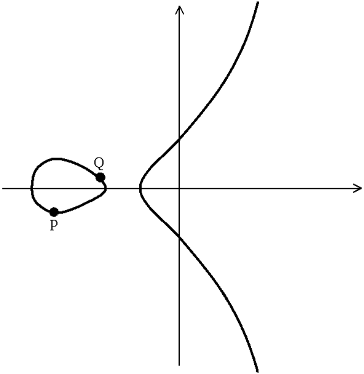
Divisors

- We can define a formal group Div(E) over the points of the curve E
- We work in the projective space $\mathbb{P}^2(K)$: we add a point at infinity $\mathcal{O}$.
- The point at infinity acts as a zero for the group

The jacobian

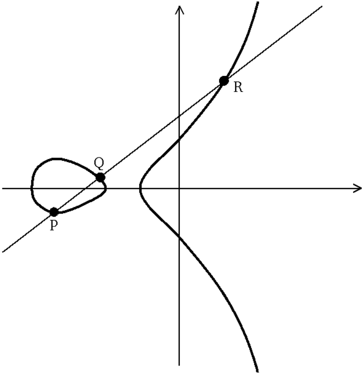
- With "some algebra", we define the group Jac(E) as a quotient of Div(E)
- Elements of Jac(E) are in one-to-one correspondence with the points of the curve, we note $E(K)$ the set of (rational) points of E.
- It turns out that the operation of the jacobian has a simple geometric interpretation...

Adding



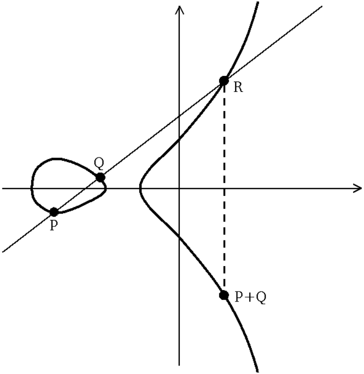
- $P = (x_0, y_0)$, $Q = (x_1, y_1)$
- $-P = (x_0, -y_0)$
- we assume $P \neq \pm Q$

Adding



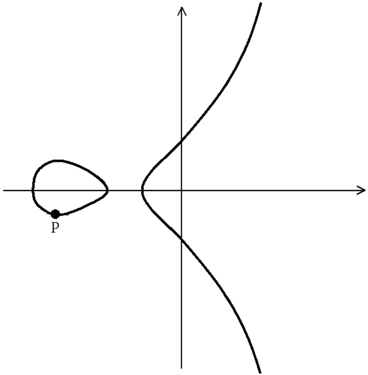
- $P = (x_0, y_0)$, $Q = (x_1, y_1)$
- $-P = (x_0, -y_0)$
- we assume $P \neq \pm Q$
- we set $\lambda = \frac{y_1 - y_0}{x_1 - x_0}$

Adding



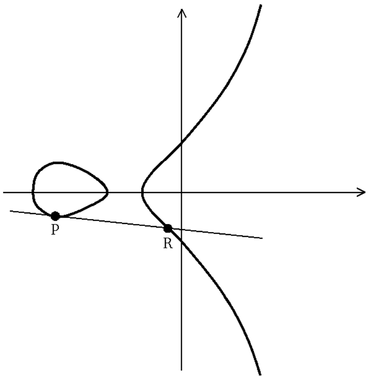
- $P = (x_0, y_0)$, $Q = (x_1, y_1)$
- $-P = (x_0, -y_0)$
- we assume $P \neq \pm Q$
- we set $\lambda = \frac{y_1 - y_0}{x_1 - x_0}$
- $x_2 = \lambda^2 - x_0 - x_1$
- $y_2 = (x_0 - x_2)\lambda - y_0$
- $P + Q = (x_2, y_2)$

Doubling



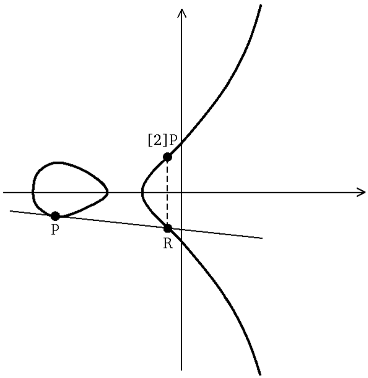
- $P = (x_0, y_0)$
- we assume $y_0 \neq 0$ (otherwise $[2]P = \mathcal{O}$)

Doubling



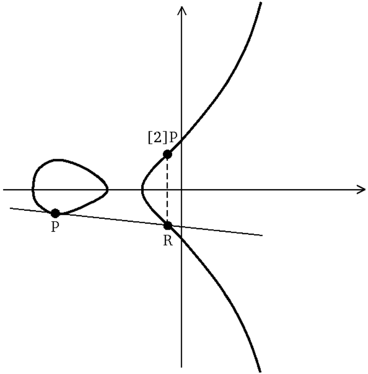
- $P = (x_0, y_0)$
- we assume $y_0 \neq 0$ (otherwise $[2]P = \mathcal{O}$)
- we set $\lambda = \frac{3x_0^2 + a_4}{2y_0}$

Doubling



- $P = (x_0, y_0)$
- we assume $y_0 \neq 0$ (otherwise $[2]P = \mathcal{O}$)
- we set $\lambda = \frac{3x_0^2 + a_4}{2y_0}$
- $x_2 = \lambda^2 - x_0 - x_1$
- $y_2 = (x_0 - x_2)\lambda - y_0$
- $[2]P = (x_2, y_2)$

Doubling



- $P = (x_0, y_0)$
- we assume $y_0 \neq 0$ (otherwise $[2]P = \mathcal{O}$)
- we set $\lambda = \frac{3x_0^2 + a_4}{2y_0}$
- $x_2 = \lambda^2 - x_0 - x_1$
- $y_2 = (x_0 - x_2)\lambda - y_0$
- $[2]P = (x_2, y_2)$
- generalizing, we note $[m]P = \underbrace{P + P + \ldots + P}_{m \text{ times}}$

Elliptic curves over finite fields

Elliptic Curve DLP

- We have a group...

Elliptic curves over finite fields

Elliptic Curve DLP

- We have a group... we want a hard DLP !

Elliptic curves over finite fields

Elliptic Curve DLP

- We have a group... we want a hard DLP !
- Infinite groups are not suitable for cryptography since the logarithm is closely related with the size of the elements
- Curves over finite fields are the good choice

Elliptic curves over finite fields

Elliptic Curve DLP

- We have a group... we want a hard DLP !
- Infinite groups are not suitable for cryptography since the logarithm is closely related with the size of the elements
- Curves over finite fields are the good choice

Theorem (Hasse's theorem)

Let E be an elliptic curve defined over a field $\mathbb{F}_q$, then we have

$$|\#E(\mathbb{F}_q) - q - 1| \leq 2\sqrt{q}.$$

Elliptic curves over finite fields

Elliptic Curve DLP

- We have a group... we want a hard DLP !
- Infinite groups are not suitable for cryptography since the logarithm is closely related with the size of the elements
- Curves over finite fields are the good choice

Theorem (Hasse's theorem)

Let E be an elliptic curve defined over a field $\mathbb{F}_q$, then we have

$$|\#E(\mathbb{F}_q) - q - 1| \leq 2\sqrt{q}.$$

Remarks

- There exist effective algorithms to calculate $\#E(F_q)$, see [BSS 1] and [BSS 2] for further readings.

Elliptic Curve Discrete Logarithm Problem

DLP

- A cyclic group $(G, *)$, a generator g of G of order n
- G is isomorphic to $\mathbb{Z}/n\mathbb{Z}$ via the bijection

$$\log_g \; : \; g^x \mapsto x$$

Elliptic Curve Discrete Logarithm Problem

DLP

- A cyclic group $(G,*)$, a generator g of G of order n
- G is isomorphic to $\mathbb{Z}/n\mathbb{Z}$ via the bijection

$$\log_g : g^x \mapsto x$$

Elliptic Curve Discrete Logarithm Problem

DLP

- A cyclic subgroup $\mathrm{Jac}(E)$, a generator g of $\mathrm{Jac}(E)$ of order n
- G is isomorphic to $\mathbb{Z}/n\mathbb{Z}$ via the bijection

$$\log_g : g^x \mapsto x$$

Elliptic Curve Discrete Logarithm Problem

DLP

- A cyclic subgroup Jac(E), a generator g of Jac(E) of order n
- G is isomorphic to $\mathbb{Z}/n\mathbb{Z}$ via the bijection

$$\log_g \;:\; g^x \mapsto x$$

Elliptic Curve Discrete Logarithm Problem

DLP

- A cyclic subgroup Jac(E), a generator P of Jac(E) of order n
- G is isomorphic to $\mathbb{Z}/n\mathbb{Z}$ via the bijection

$$\log_g : g^x \mapsto x$$

Elliptic Curve Discrete Logarithm Problem

DLP

- A cyclic subgroup Jac(E), a generator P of Jac(E) of order n
- G is isomorphic to $\mathbb{Z}/n\mathbb{Z}$ via the bijection

$$\log_g \; : \; g^x \mapsto x$$

Elliptic Curve Discrete Logarithm Problem

DLP

- A cyclic subgroup Jac(E), a generator P of Jac(E) of order n
- Jac(E) is isomorphic to $\mathbb{Z}/n\mathbb{Z}$ via the bijection

$$\log_g : g^x \mapsto x$$

Elliptic Curve Discrete Logarithm Problem

DLP

- A cyclic subgroup Jac(E), a generator P of Jac(E) of order n
- Jac(E) is isomorphic to $\mathbb{Z}/n\mathbb{Z}$ via the bijection

$$\log_g : g^x \mapsto x$$

Elliptic Curve Discrete Logarithm Problem

DLP

- A cyclic subgroup Jac(E), a generator P of Jac(E) of order n
- Jac(E) is isomorphic to $\mathbb{Z}/n\mathbb{Z}$ via the bijection

$$\log_g : [x]P \mapsto x$$

Elliptic Curve Discrete Logarithm Problem

DLP

- A cyclic subgroup Jac(E), a generator P of Jac(E) of order n
- Jac(E) is isomorphic to $\mathbb{Z}/n\mathbb{Z}$ via the bijection

$$\log_g : [x]P \mapsto x$$

Elliptic Curve Discrete Logarithm Problem

ECDLP

- A cyclic subgroup Jac(E), a generator P of Jac(E) of order n
- Jac(E) is isomorphic to $\mathbb{Z}/n\mathbb{Z}$ via the bijection

$$\log_g : [x]P \mapsto x$$

Elliptic Curve Discrete Logarithm Problem

ECDLP

- A cyclic subgroup Jac(E), a generator P of Jac(E) of order n
- Jac(E) is isomorphic to $\mathbb{Z}/n\mathbb{Z}$ via the bijection

$$\log_g : [x]P \mapsto x$$

Hardness of ECDLP

ECDLP is easy for various classes of elliptic curves :

- n is not prime $\rightarrow$ Pohlig-Hellman
- $n < 2^{160}$ $\rightarrow$ BSGS or Pollard's Rho
- $n = \text{char}(K)$ $\rightarrow$ *anomalous attack* (see [BSS 1])
- $(\#K)^t = 1 \mod n$ for a $t < 20$ $\rightarrow$ *MOV attack* (see [BSS 1])
- $\#K = p^l$ with l not prime $\rightarrow$ *Weil descent* (see [BSS 2])

Elliptic Curve Discrete Logarithm Problem

ECDLP

- A cyclic subgroup Jac(E), a generator P of Jac(E) of order n
- Jac(E) is isomorphic to $\mathbb{Z}/n\mathbb{Z}$ via the bijection

$$\log_g : [x]P \mapsto x$$

Hardness of ECDLP

- But for all the other cases no better algorithm is known than BSGS or Pollard's Rho !

Elliptic Curve Discrete Logarithm Problem

ECDLP

- A cyclic subgroup Jac(E), a generator P of Jac(E) of order n
- Jac(E) is isomorphic to $\mathbb{Z}/n\mathbb{Z}$ via the bijection

$$\log_g : [x]P \mapsto x$$

Hardness of ECDLP

- But for all the other cases no better algorithm is known than BSGS or Pollard's Rho !
- Thus, for crytpographic use, we select a random curve and verify that it's ECDLP is not easy

Plan

Elliptic Curve Diffie-Hellman (ECDH)

A group G of prime order p. A generator g of G.

- chooses $a \in \mathbb{Z}/p\mathbb{Z}$ at random
- computes g^a

- chooses $b \in \mathbb{Z}/p\mathbb{Z}$ at random
- computes g^b

$$\xrightarrow{\qquad g^a \qquad}$$

$$\xleftarrow{\qquad g^b \qquad}$$

- computes $K_{ab} = \left(g^b\right)^a$

- computes $K_{ab} = \left(g^a\right)^b$

Elliptic Curve Diffie-Hellman (ECDH)

A group G of prime order p. A generator g of G.

- chooses $a \in \mathbb{Z}/p\mathbb{Z}$ at random
- computes g^a

- chooses $b \in \mathbb{Z}/p\mathbb{Z}$ at random
- computes g^b

$$\xrightarrow{\quad g^a \quad}$$
$$\xleftarrow{\quad g^b \quad}$$

- computes $K_{ab} = \left(g^b\right)^a$

- computes $K_{ab} = \left(g^a\right)^b$

Elliptic Curve Diffie-Hellman (ECDH)

An elliptic curve E, a finite field K, a subgroup of prime order p of $E(K)$, a generator P.

- chooses $a \in \mathbb{Z}/p\mathbb{Z}$ at random
- computes g^a

- chooses $b \in \mathbb{Z}/p\mathbb{Z}$ at random
- computes g^b

$$\xrightarrow{\quad g^a \quad}$$
$$\xleftarrow{\quad g^b \quad}$$

- computes $K_{ab} = \left(g^b\right)^a$

- computes $K_{ab} = \left(g^a\right)^b$

Elliptic Curve Diffie-Hellman (ECDH)

An elliptic curve E, a finite field K, a subgroup of prime order p of $E(K)$, a generator P.

- chooses $a \in \mathbb{Z}/p\mathbb{Z}$ at random
- computes g^a

- chooses $b \in \mathbb{Z}/p\mathbb{Z}$ at random
- computes g^b

$$\xrightarrow{\quad g^a \quad}$$
$$\xleftarrow{\quad g^b \quad}$$

- computes $K_{ab} = \left(g^b\right)^a$

- computes $K_{ab} = \left(g^a\right)^b$

Elliptic Curve Diffie-Hellman (ECDH)

An elliptic curve E, a finite field K, a subgroup of prime order p of $E(K)$, a generator P.

- chooses $a \in \mathbb{Z}/p\mathbb{Z}$ at random
- computes $[a]P$

- chooses $b \in \mathbb{Z}/p\mathbb{Z}$ at random
- computes $[b]P$

$$\xrightarrow{\quad g^a \quad}$$
$$\xleftarrow{\quad g^b \quad}$$

- computes $K_{ab} = \left(g^b\right)^a$

- computes $K_{ab} = \left(g^a\right)^b$

Elliptic Curve Diffie-Hellman (ECDH)

An elliptic curve E, a finite field K, a subgroup of prime order p of $E(K)$, a generator P.

- chooses $a \in \mathbb{Z}/p\mathbb{Z}$ at random
- computes $[a]P$

- chooses $b \in \mathbb{Z}/p\mathbb{Z}$ at random
- computes $[b]P$

$$\xrightarrow{\quad g^a \quad}$$

$$\xleftarrow{\quad g^b \quad}$$

- computes $K_{ab} = \left(g^b\right)^a$

- computes $K_{ab} = \left(g^a\right)^b$

Elliptic Curve Diffie-Hellman (ECDH)

An elliptic curve E, a finite field K, a subgroup of prime order p of $E(K)$, a generator P.

- chooses $a \in \mathbb{Z}/p\mathbb{Z}$ at random
- computes $[a]P$

- chooses $b \in \mathbb{Z}/p\mathbb{Z}$ at random
- computes $[b]P$

$$\xrightarrow{[a]P}$$

$$\xleftarrow{[b]P}$$

- computes $K_{ab} = \left(g^b\right)^a$

- computes $K_{ab} = \left(g^a\right)^b$

Elliptic Curve Diffie-Hellman (ECDH)

An elliptic curve E, a finite field K, a subgroup of prime order p of $E(K)$, a generator P.

- chooses $a \in \mathbb{Z}/p\mathbb{Z}$ at random
- computes $[a]P$

- chooses $b \in \mathbb{Z}/p\mathbb{Z}$ at random
- computes $[b]P$

$$\xrightarrow{[a]P}$$
$$\xleftarrow{[b]P}$$

- computes $K_{ab} = \left(g^b\right)^a$

- computes $K_{ab} = \left(g^a\right)^b$

Elliptic Curve Diffie-Hellman (ECDH)

An elliptic curve E, a finite field K, a subgroup of prime order p of $E(K)$, a generator P.

- chooses $a \in \mathbb{Z}/p\mathbb{Z}$ at random
- computes $[a]P$

- chooses $b \in \mathbb{Z}/p\mathbb{Z}$ at random
- computes $[b]P$

$$\xrightarrow{[a]P}$$

$$\xleftarrow{[b]P}$$

- computes $K_{ab} = [a]([b]P)$

- computes $K_{ab} = [b]([a]P)$

ECDH Problems

ECCDH and ECDDH

We define the problems computational ECDH and decisional ECDH the same way we did for CDH and DDH

Elliptic Curve Digital Signature Algorithm (ECDSA)

Parameters

- A t-uple (E, K, n, P)
- A hash function $H : \{0,1\}^* \to \{0,1\}^l$
- A private key $x \in \mathbb{Z}/p\mathbb{Z}$ and a public key $Y = [x]P$

Elliptic Curve Digital Signature Algorithm (ECDSA)

Parameters

- A t-uple (E, K, n, P)
- A hash function $H : \{0,1\}^* \to \{0,1\}^l$
- A private key $x \in \mathbb{Z}/p\mathbb{Z}$ and a public key $Y = [x]P$

Signing a message m

1. Choose $k \in \mathbb{Z}/p\mathbb{Z}$ at random
2. $T \leftarrow [k]P$
3. $r \leftarrow x(T) \mod p$
4. $e \leftarrow H(m)$
5. $s \leftarrow \frac{e+xr}{k} \mod p$
6. Return (r, s)

Elliptic Curve Digital Signature Algorithm (ECDSA)

Parameters

- A t-uple (E, K, n, P)
- A hash function $H : \{0,1\}^* \longrightarrow \{0,1\}^l$
- A private key $x \in \mathbb{Z}/p\mathbb{Z}$ and a public key $Y = [x]P$

Signing a message m

1. Choose $k \in \mathbb{Z}/p\mathbb{Z}$ at random
2. $T \leftarrow [k]P$
3. $r \leftarrow x(T) \mod p$
4. $e \leftarrow H(m)$
5. $s \leftarrow \frac{e+xr}{k} \mod p$
6. Return (r, s)

Verifying a signature (r, s)

1. $e \leftarrow H(m)$
2. $u \leftarrow \frac{e}{s}$
3. $v \leftarrow \frac{r}{s}$
4. $T \leftarrow [u]P + [v]Y$
5. Accept if and only if $r = x(T) \mod p$

Summary

Other protocols

- ECMQV authentified key agreement
- ECIES integrated encryption system

Summary

Other protocols

- ECMQV authentified key agreement
- ECIES integrated encryption system

Security parameters

- DLP over finite fields requires nowadays 1024 bit keys to achieve a good security level (80 bits)
- For a comparable security level, ECDLP requires lesss than 200 bit keys
- The gain is given by the equation

$$n \approx N^{1/3}$$

  where n is the number of bits required for an EC cryptosystem and N is the number of bits required for a conventional one

Plan

1. Cryptography based on groups
   - Discrete Logarithm Problem
   - The Diffie-Hellman Problems
2. Elliptic curves
   - The arithmetic of elliptic curves
   - Elliptic Curve Discrete Logarithm Problem
3. Elliptic curve cryptography
   - ECDH
   - ECDSA
   - Summary
4. New perspectives in ECC
   - Pairings
   - Tripartite Diffie-Hellman
   - Identity Based Encryption

Pairings

Definition (Pairing)

Given two groups $(G_1, +_1)$ and $(G_2, +_2)$ with same exponent n, given a cyclic group $(G_3, *)$ of order n, a pairing is a function

$$e \; : \; G_1 \times G_2 \to G_3$$

satisfying the following properties :

Bilinearity :

- $e(P + P', Q) = e(P, Q)e(P', Q)$
- $e(P, Q + Q') = e(P, Q)e(P, Q')$

Non-degeneracy :

- for all P there is a Q such that $e(P, Q) \neq 1$
- for all Q there is a P such that $e(P, Q) \neq 1$

Pairings

Definition (Self-pairing)

With the same notation as above, taking $G_1 = G_2$, we define a self-pairing as function

$$e \ : \ G_1 \times G_1 \rightarrow G_3$$

satisfying the following properties :

Bilinearity :

- $e(P + P', Q) = e(P, Q)e(P', Q)$
- $e(P, Q + Q') = e(P, Q)e(P, Q')$

Symmetry : $e(P, Q) = e(Q, P)$ for all P and Q

Non-degeneracy : $e(P, P) \neq 1$ for all P

Pairings

Pairings over elliptic curves

- Suppose G_1 and G_2 are groups of points of elliptic curves

Pairings

Pairings over elliptic curves

- Suppose G_1 and G_2 are groups of points of elliptic curves
- Then pairings exist with G_3 a multiplicative subgroup of a finite field

Pairings

Pairings over elliptic curves

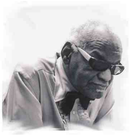
- Suppose G_1 and G_2 are groups of points of elliptic curves
- Then pairings exist with G_3 a multiplicative subgroup of a finite field
- If G_1 is a subgroup of Jac(E) for a $E(\mathbb{F}_q)$, then there exist a $k \in \mathbb{N}$ (called the *embedding degree*) and a self-pairing s.t. G_3 is a multilpicative subgroup of $\mathbb{F}_{q^k}$

Pairings

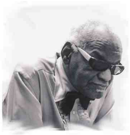
Pairings over elliptic curves

- Suppose G_1 and G_2 are groups of points of elliptic curves
- Then pairings exist with G_3 a multiplicative subgroup of a finite field
- If G_1 is a subgroup of Jac(E) for a $E(\mathbb{F}_q)$, then there exist a $k \in \mathbb{N}$ (called the *embedding degree*) and a self-pairing s.t. G_3 is a multilpicative subgroup of $\mathbb{F}_{q^k}$
- There exist classes of curves for which there is a pairing effectively computable, ECCDH is hard for the curve and DDH is hard for the finite field

Tripartite Diffie-Hellman (3DH)

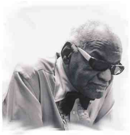


Tripartite Diffie-Hellman (3DH)



G_1 sugroup of $E(\mathbb{F}_q)$, G_3 subgroup of $\mathbb{F}_{q^k}$, a self-pairing e, a generator P of G_1

Tripartite Diffie-Hellman (3DH)

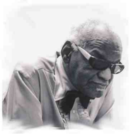


G_1 sugroup of $E(\mathbb{F}_q)$, G_3 subgroup of $\mathbb{F}_{q^k}$, a self-pairing e, a generator P of G_1

- select a random a
- select a random b
- select a random c

Tripartite Diffie-Hellman (3DH)





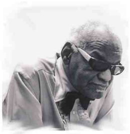
G_1 sugroup of $E(\mathbb{F}_q)$, G_3 subgroup of $\mathbb{F}_{q^k}$, a self-pairing e, a generator P of G_1

- select a random a
- broadcast $[a]P$

- select a random b
- broadcast $[b]P$

- select a random c
- broadcast $[b]P$

Tripartite Diffie-Hellman (3DH)

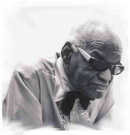
G_1 sugroup of $E(\mathbb{F}_q)$, G_3 subgroup of $\mathbb{F}_{q^k}$, a self-pairing e, a generator P of G_1

- select a random a
- broadcast $[a]P$
- $K_{abc} = e([b]P, [c]P)^a$

- select a random b
- broadcast $[b]P$
- $K_{abc} = e([a]P, [c]P)^b$

- select a random c
- broadcast $[b]P$
- $K_{abc} = e([a]P, [b]P)^c$

Tripartite Diffie-Hellman (3DH)



G_1 sugroup of $E(\mathbb{F}_q)$, G_3 subgroup of $\mathbb{F}_{q^k}$, a self-pairing e, a generator P of G_1

- select a random a
- broadcast $[a]P$
- $K_{abc} = e([b]P, [c]P)^a$

- select a random b
- broadcast $[b]P$
- $K_{abc} = e([a]P, [c]P)^b$

- select a random c
- broadcast $[b]P$
- $K_{abc} = e([a]P, [b]P)^c$

$$K_{abc} = e(P, P)^{abc}$$

Identity Based Non-Interactive Key Distribution

Trusted Authority

Identity Based Non-Interactive Key Distribution

Trusted Authority

A t-uple (G_1, G_3, e, P), a hash function $H\ :\ \Sigma^* \to G_1$

Identity Based Non-Interactive Key Distribution

Trusted Authority

A t-uple (G_1, G_3, e, P), a hash function $H : \Sigma^* \to G_1$

- has a *master secret* s

Identity Based Non-Interactive Key Distribution

Trusted Authority

A t-uple (G_1, G_3, e, P), a hash function $H : \Sigma^* \rightarrow G_1$

- has a public ID $Q_A = H(\text{Alice})$

- has a public ID $Q_B = H(\text{Bob})$

- has a *master secret* s

Identity Based Non-Interactive Key Distribution

Trusted Authority

A t-uple (G_1, G_3, e, P), a hash function $H : \Sigma^* \to G_1$

- has a public ID $Q_A = H(\text{Alice})$

- has a public ID $Q_B = H(\text{Bob})$

- has a *master secret* s
- gives $S_A = [s]Q_A$ to Alice over a private channel

Identity Based Non-Interactive Key Distribution

Trusted Authority

A t-uple (G_1, G_3, e, P), a hash function $H \; : \; \Sigma^* \to G_1$

- has a public ID $Q_A = H(\text{Alice})$

- has a public ID $Q_B = H(\text{Bob})$

- has a *master secret* s
- gives $S_A = [s]Q_A$ to Alice over a private channel
- gives $S_B = [s]Q_B$ to Bob over a private channel

Identity Based Non-Interactive Key Distribution

Trusted Authority

A t-uple (G_1, G_3, e, P), a hash function $H : \Sigma^* \to G_1$

- has a public ID $Q_A = H(\text{Alice})$
- $K_{AB} = e(S_A, Q_B)$

- has a public ID $Q_B = H(\text{Bob})$
- $K_{AB} = e(Q_A, S_B)$

- has a *master secret* s
- gives $S_A = [s]Q_A$ to Alice over a private channel
- gives $S_B = [s]Q_B$ to Bob over a private channel

Identity Based Non-Interactive Key Distribution

Trusted Authority

A t-uple (G_1, G_3, e, P), a hash function $H \; : \; \Sigma^* \rightarrow G_1$

- has a public ID $Q_A = H(\text{Alice})$
- $K_{AB} = e(S_A, Q_B)$

- has a public ID $Q_B = H(\text{Bob})$
- $K_{AB} = e(Q_A, S_B)$

- has a *master secret* s
- gives $S_A = [s]Q_A$ to Alice over a private channel
- gives $S_B = [s]Q_B$ to Bob over a private channel

$$K_{AB} = (Q_A, Q_B)^s$$

Identity Based Encryption

Trusted Authority

Identity Based Encryption

Trusted Authority

(G_1, G_3, e, P), hash functions $H_1 : \Sigma^* \to G_1$ and $H_2 : G_3 \to \{0,1\}^n$

Identity Based Encryption

Trusted Authority

(G_1, G_3, e, P), hash functions $H_1 : \Sigma^* \rightarrow G_1$ and $H_2 : G_3 \rightarrow \{0,1\}^n$

- has a public ID $Q_A = H(\text{Alice})$

- has a *master secret* s
- has a public key $Q_0 = [s]P$

Identity Based Encryption

Trusted Authority

(G_1, G_3, e, P), hash functions $H_1 : \Sigma^* \longrightarrow G_1$ and $H_2 : G_3 \longrightarrow \{0,1\}^n$

- has a public ID $Q_A = H(\text{Alice})$

- has a *master secret* s
- has a public key $Q_0 = [s]P$
- gives $S_A = [s]Q_A$ to Alice over a private channel

Identity Based Encryption

Trusted Authority

(G_1, G_3, e, P), hash functions $H_1 : \Sigma^* \longrightarrow G_1$ and $H_2 : G_3 \longrightarrow \{0,1\}^n$

- has a public ID $Q_A = H(\text{Alice})$

- a message M

- has a *master secret* s
- has a public key $Q_0 = [s]P$
- gives $S_A = [s]Q_A$ to Alice over a private channel

Identity Based Encryption

Trusted Authority

(G_1, G_3, e, P), hash functions $H_1 : \Sigma^* \longrightarrow G_1$ and $H_2 : G_3 \longrightarrow \{0,1\}^n$

- has a public ID $Q_A = H(\text{Alice})$

- a message M
- select a random t

- has a *master secret* s
- has a public key $Q_0 = [s]P$
- gives $S_A = [s]Q_A$ to Alice over a private channel

Identity Based Encryption

Trusted Authority

(G_1, G_3, e, P), hash functions $H_1 : \Sigma^* \to G_1$ and $H_2 : G_3 \to \{0,1\}^n$

- has a public ID $Q_A = H(\text{Alice})$

- a message M
- select a random t
- $U = [t]P$

- has a *master secret* s
- has a public key $Q_0 = [s]P$
- gives $S_A = [s]Q_A$ to Alice over a private channel

Identity Based Encryption

Trusted Authority

(G_1, G_3, e, P), hash functions $H_1 : \Sigma^* \to G_1$ and $H_2 : G_3 \to \{0,1\}^n$

- has a public ID $Q_A = H(\text{Alice})$

- a message M
- select a random t
- $U = [t]P$
- $V = M \oplus H_2(e(Q_A, Q_0)^t)$

- has a *master secret* s
- has a public key $Q_0 = [s]P$
- gives $S_A = [s]Q_A$ to Alice over a private channel

Identity Based Encryption

Trusted Authority

(G_1, G_3, e, P), hash functions $H_1 : \Sigma^* \to G_1$ and $H_2 : G_3 \to \{0,1\}^n$

- has a public ID $Q_A = H(\text{Alice})$

- a message M
- select a random t
- $U = [t]P$
- $V = M \oplus H_2(e(Q_A, Q_0)^t)$

$\xleftarrow{(U,V)}$

- has a *master secret* s
- has a public key $Q_0 = [s]P$
- gives $S_A = [s]Q_A$ to Alice over a private channel

Identity Based Encryption

Trusted Authority

(G_1, G_3, e, P), hash functions $H_1 : \Sigma^* \rightarrow G_1$ and $H_2 : G_3 \rightarrow \{0,1\}^n$

- has a public ID $Q_A = H(\text{Alice})$

- a message M
- select a random t
- $U = [t]P$
- $V = M \oplus H_2(e(Q_A, Q_0)^t)$

- has a *master secret* s
- has a public key $Q_0 = [s]P$
- gives $S_A = [s]Q_A$ to Alice over a private channel

$\xleftarrow{(U,V)}$

- $M = V \oplus H_2(e(S_A, U))$

Bibliography I

J.H. Silverman
The Arithmetic of Elliptic Curves
GTM 106, Springer-Verlag, 1986

I. Blake, G. Seroussi & N. Smart
Elliptic Curves in Cryptography
LMS 265, Cambridge University Press, 1999

(edited by) I. Blake, G. Seroussi & N. Smart
Advances in Elliptic Curve Cryptography
LMS 317, Cambridge University Press, 2005