Introduction to mathematical cryptography

Lecture 4: Elliptic Curve Cryptography

Sabrina Kunzweiler

Preliminary Arizona Winter School 2025

What we learned so far

Lecture 1 The Diffie-Hellman key exchange - first protocol in public key cryptography.

Lecture 2 DLP in a prime order group (order N) in finite fields.

- Baby-step giant-step and Pollard's rho: $O(\sqrt{N})$
- Index calculus: subexponential running time

Lecture 3 We discussed elliptic curves (mostly over finite fields).

- Addition law on the set of points
- Cryptographic one-way function given by scalar multiplication

---

This lecture We study Elliptic Curve Diffie-Hellman and the ECDLP

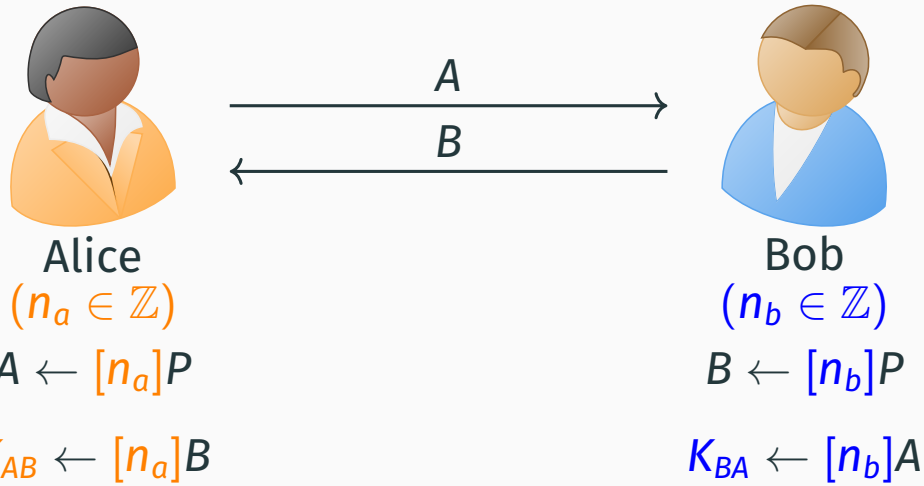
Introducing ECDH **and** ECDLP

Elliptic curve Diffie-Hellman (ECDH)

Setup

- Elliptic curve E over $\mathbb{F}_q$

- Point $P \in E(\mathbb{F}_q)$, $\#\langle P \rangle = N$



- **Secret keys** $(n_a \in \mathbb{Z})$ and $(n_b \in \mathbb{Z})$
- **Public keys** A and $B \in E(\mathbb{F}_q)$
- **Shared key** $K_{AB} = K_{BA} \in E(\mathbb{F}_q)$

ECDH **Example**

Setup $E : y^2 = x^3 - 3x + 1$ over $\mathbb{F}_{13}$, and $P = (0, 1) \in E(\mathbb{F}_{13})$, $ord(P) = 19$.

```
sage: Fp = GF(13)
sage: E = EllipticCurve(Fp, (-3,
   1))
sage: P = E([0,1])
sage: A = 5 * P; A
(1 : 8 : 1)
sage: K = 5 * E([4,1]); K
(10 : 3 : 1)
```

Alice

$(n_a = 5)$

$A = [5]P = (1, 8)$

$K_{AB} = [5](4, 1) = (10, 3)$

$(1, 8)$

$(4, 1)$

Bob

$(n_b = 12)$

$B = [12]P = (4, 1)$

$K_{BA} = [12](1, 8) = (10, 3)$

Real-world setup (Secp256k1 used in Bitcoin)

```
sage: p = 0xfffffffffffffffffffffffffffffffffffffffffffffffffffffffefffffc2f
sage: Fp = GF(p)
sage: a = Fp(0)
sage: b = Fp(7)
sage: E = EllipticCurve(Fp, (a, b))
sage: P = E(0x79be667ef9dcbbac55a06295ce870b07029bfcdb2dce28d959f2815b16f81798,  0
 x483ada7726a3c4655da4fbfc0e1108a8fd17b448a68554199c47d08ffb10d4b8)
```

Elliptic curve discrete logarithm problem

Elliptic Curve Discrete Logarithm Problem (ECDLP)

For $P \in E(K)$ and $Q \in \langle P \rangle$, the ECDLP asks to find $n_a \in \mathbb{Z}$ so that $[n_a]P = Q$.
Notation: $n_a = \mathrm{dlog}_P(Q)$.

- Natural analogue of DLP
  $\Rightarrow$ the word *logarithm* is used even though E is an additive group.
- If DLP is hard, then scalar multiplication is a cryptographic one-way function.

Analysis of ECDH and ECDLP

How hard is it to compute $\log_P(Q)$ for some $P \in E(K)$ of order N?

What we already know

(from the generic group setting)

- Pohlig-Hellman For $N = \prod p_i$ composite, the hardness only depends on the largest prime factor p_i
- Pollard's rho We can solve $\log_P(Q)$ in time $O(\sqrt{N})$ and constant memory

What we will study now

(specific to elliptic curves)

- MOV attack (Menezes-Okamoto-Vanstone): subexponential algorithm for some (weak) parameters.
- Invalid curve attack: implementation specific attack on ECDH

The MOV attack

Pairings

Pairing

Let G_1, G_2, H be groups. A **pairing** is a map
$e : G_1 \times G_2 \to H, \quad (g_1, g_2) \mapsto h = e(g_1, g_2)$
which is bilinear and non-degenerate.

In this lecture, we study pairings, where

- $G_1 = G_2 = E[N]$: The N-torsion group of an elliptic curve E
- H is an "easier group"

Main idea of MOV

Use a pairing to translate ECDLP in $E(\mathbb{F}_q)$ to DLP in $\mathbb{F}_{q^d}$.

Determinant pairing

Determinant pairing

$E/\mathbb{F}_q$ elliptic curve, $N \in \mathbb{N}$, $\gcd(q, N) = 1$. Write

$$E[N] = \langle T_1, T_2 \rangle \cong \mathbb{Z}/N\mathbb{Z} \times \mathbb{Z}/N\mathbb{Z}.$$

The **determinant pairing** w.r.t. (T_1, T_2) is

$$\det : E[N] \times E[N] \to \mathbb{Z}/N\mathbb{Z}, (aT_1 + bT_2, cT_1 + dT_2) \mapsto ad - bc$$

Properties: bilinear, alternating, non-degenerate

(a) bilinear:

$$\det(P_1 + P_2, Q) = \det(P_1, Q) + \det(P_2, Q),$$
$$\det(P, Q_1 + Q_2) = \det(P, Q_1) + \det(P, Q_2),$$

for all $P_1, P_2, Q_1, Q_2 \in E[N]$

Solving ECDLP with the determinant pairing (?)

ECDLP **challenge**: $Q \in \langle P \rangle$, find $n_a = \mathrm{dlog}_P(Q)$.
($E/\mathbb{F}_q$ elliptic curve, $N = \#\langle P \rangle$)

1. Find $T \in E[N]$ with $\alpha = \det(P, T)$ has order N.
2. Compute $\det(Q, T) = \beta$.
3. Note that $\det(\underbrace{P + \cdots + P}_{n_a}, T) = n_a \cdot \det(P, T)$ (linearity),

   $\Rightarrow n_a = \beta/\alpha \in \mathbb{Z}/N\mathbb{Z}$.

Two problems with this approach

- The full torsion group $E[N]$ is only defined over a field extension, i.e. $T \in E(\mathbb{F}_{q^d})$ for possibly large d.
- It is not known how to compute the determinant pairing efficiently.

Weil pairing

The **Weil pairing** (André Weil '1940) is a pairing

$$e_N : E[N] \times E[N] \to \boldsymbol{\mu_N} \subset \mathbb{F}_{q^d}$$

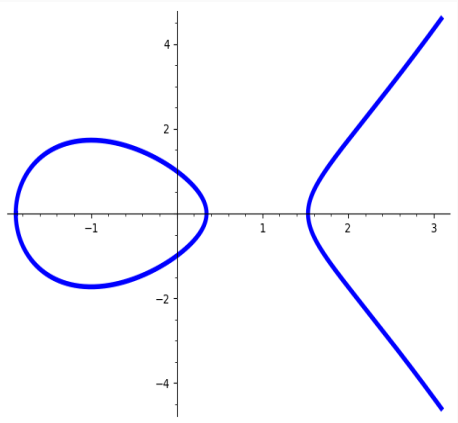
with $E/\mathbb{F}_q$ elliptic curve, $\gcd(N, q) = 1$ and $\boldsymbol{\mu_N}$ the group of N-th roots of unity.

- **Properties**: bilinear, alternating, nondegenerate (and more)
- Relation with the determinant pairing (exercise): $e_N(P, Q) = \mu^{\det(P,Q)}$, where $\mu = e_N(T_1, T_2)$
- Evaluation of e_N is efficient (in $\mathbb{F}_{q^d}$) using Miller's algorithm (Victor Miller, 1986)

We use the Weil pairing, and Miller's algorithm as a black box.

SAGE You can compute $e_N(P, Q)$ using `P.weil_pairing(Q,N)`
(more details on the evaluation of e_N in the lecture notes)

Example $N = 2$



$E : y^2 = x^3 + ax + b$ over $\mathbb{F}_q$

- Write $x^3 + ax + b = (x - \alpha_1)(x - \alpha_2)(x - \alpha_3)$
- $E[2] = \{(\alpha_1, 0), (\alpha_2, 0), (\alpha_3, 0), \infty\}$
  $\{P_1, P_2, P_3, P_\infty\}$

The 2-Weil pairing $e_2 : E[2] \times E[2] \to \{\pm 1\}$

$$(P_i, P_j) \mapsto \begin{cases} 1 & \text{if } i = j \text{ or } \infty \in \{i, j\} \\ -1 & \text{otherwise} \end{cases}$$

Embedding degree

Embedding degree

Let N be a positive integer, and $\mathbb{F}_q$ a finite field. The smallest value d with $N \mid q^d - 1$ is called the **embedding degree** of N in $\mathbb{F}_q$.

- $\mathbb{F}_{q^d}$ is the smallest field extension with $\boldsymbol{\mu_N \subset \mathbb{F}_{q^d}}$.
- If $\gcd(N, q-1) = 1$, then $E[N] \subset E(\mathbb{F}_{q^d})$ (Silverman, Lemma XI.6.2)

Examples (slide 3)

(a) $E(\mathbb{F}_{13}) = 19$, and the smallest integer d with $19 \mid 13^d - 1$ is $d = 18$.

(b) $E(\mathbb{F}_p) = N \approx 2^{256}$ (Secp256k1)
$d = 1929868153955269923726183083478131797547292737984581739$
7100860523586360249056, in particular $d \approx 2^{253}$.

The MOV algorithm

by Alfred Menezes, Scott Vanstone, Tatsuaki Okamoto (1991)

Algorithm 1 MOV algorithm

Input: $P \in E(\mathbb{F}_q)$ with $ord(P) = N$, $Q \in \langle P \rangle$, and $\gcd(q-1, N) = 1$.

Output: $g, A \in \mathbb{F}_{q^d}^*$ with $\mathrm{dlog}_g(A) = \mathrm{dlog}_P(Q)$.

1: $d \leftarrow$ embedding degree of N in $\mathbb{F}_q$
2: Determine $E(\mathbb{F}_{q^d}) \cong \mathbb{Z}/N_1\mathbb{Z} \times \mathbb{Z}/N_2\mathbb{Z}$ [a]
3: **repeat**
4: $\quad T \xleftarrow{\$} E(\mathbb{F}_{q^d})$
5: $\quad T \leftarrow [N_1/N] \cdot T$
6: $\quad g \leftarrow e_N(P, T) \in \boldsymbol{\mu_N} \subset \mathbb{F}_{q^d}^*$
7: **until** $ord(g) = N$
8: $A \leftarrow e_N(Q, T) \in \boldsymbol{\mu_N} \subset \mathbb{F}_{q^d}^*$
9: **return** (g, A)

[a] Can be done in heuristic polynomial time.

Correctness

✓ $e_N(Q, T) = e_N([n_a]P, T) = e_N(P, T)^{n_a}$ (linearity)
(note T is a point with non-trivial Weil-pairing)

Runtime

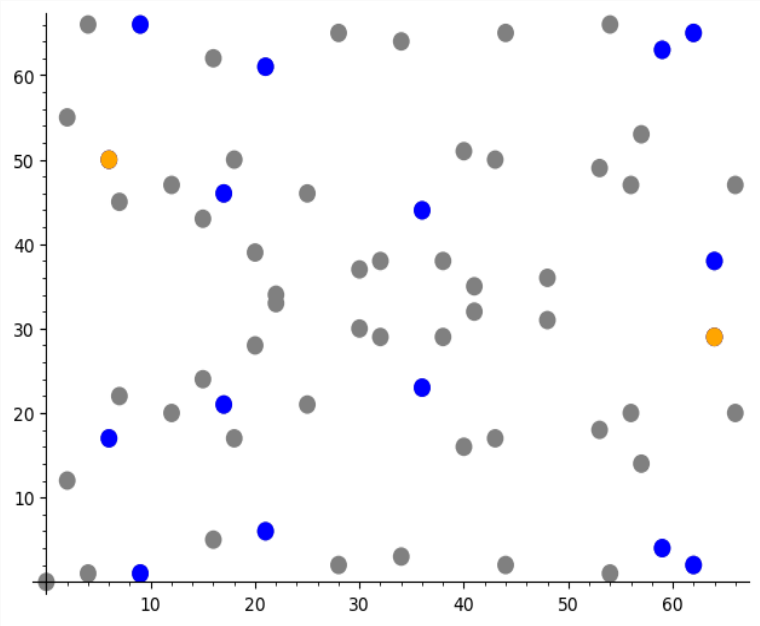
- Number of multiplication over $\mathbb{F}_{q^d}$ is polynomial in $\log(N)$

Memory

- $O(1)$ elements in $\mathbb{F}_{q^d}$

Example of the MOV algorithm

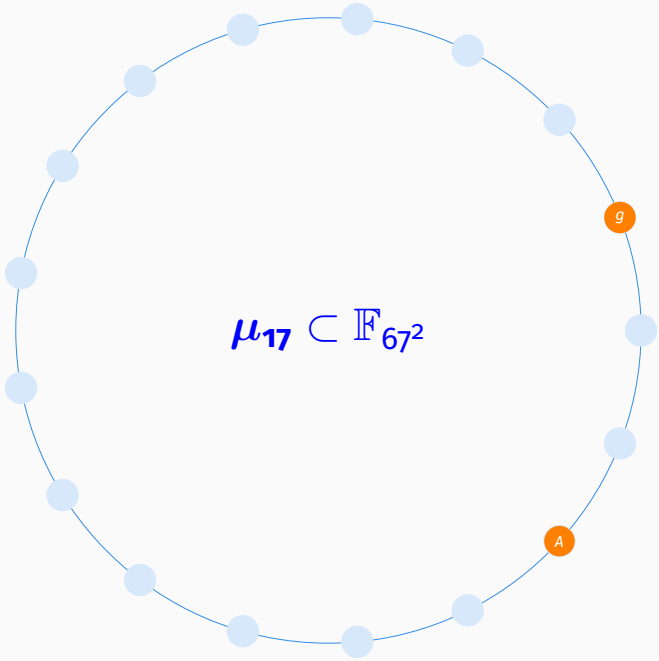
$E : y^2 = x^3 + x$ over $\mathbb{F}_{67}$



- $P = (64, 29)$, $Q = (6, 50)$
- $\#\langle P \rangle = 17$
- $17 \mid 67^2 - 1 \Rightarrow d = 2$
- $E[17] = \langle P, T \rangle$ with $T = (11i + 37, 22i + 42) \in E(\mathbb{F}_{67^2})$

MOV

$\Rightarrow$



- $g = e_{17}(P, T) = 39i + 50$
- $A = e_{17}(Q, T) = 46i + 30$

$\Rightarrow \operatorname{dlog}_P(Q) = \operatorname{dlog}_g(A) = 14.$

Consequences for Elliptic Curve Cryptography

Destructive

- ECDLP can be solved in subexponential time **if** the embedding degree is small
  (MOV + index calculus)

$\Rightarrow$ choose elliptic curves with large embedding degrees for Elliptic Curve Diffie-Hellman

Constructive

Pairings can also be used to construct more advanced cryptographic protocols

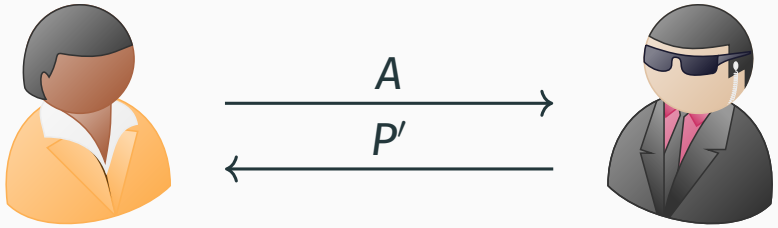
- Tripartite Diffie-Hellman key exchange (Antoine Joux, 2000)
- Identity-based encryption (Dan Boneh, Matthew Franklin, 2001)
- BLS Digital Signature (Dan Boneh, Ben Lynn, Hovav Shacham, 2004)

$\Rightarrow$ field of Pairing-based cryptography

Invalid curve attack

Invalid curve attack

by Ingrid Biehl, Bernd Meyer, Volker Müller (2000)



Setup $E : y^2 = x^3 + ax + b$ over $\mathbb{F}_q$, $P \in E(\mathbb{F}_q)$, $ord(P) = N$ is prime.

Mallory impersonates Bob, to find Alice's secret key.

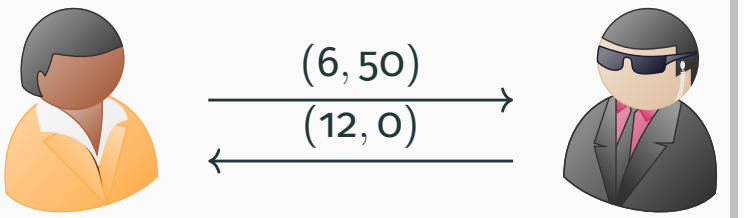
- Choose $E' : y^2 = x^3 + ax + b'$, and $P' \in E'(\mathbb{F}_q)$ with $ord(P') = m$ small
- Mallory computes all possible $K_i = [i]P'$ and check if $K_i = K_{AB}$.

$\Rightarrow$ recover $n_a \pmod m$.

Main observation The addition formulas only depend on a, not on b.

Example of an invalid curve attack

Setup $E : y^2 = x^3 + x$ over $\mathbb{F}_{67}$,
$P = (64, 29)$, $ord(P) = 17$.



$A = [14]P = (6, 50)$

$K_{AB} = [14]P' = \infty$

$K_0 = \infty$

$K_1 = P'$

Mallory chooses
$E' : y^2 = x^3 + x + 2$,
$\#E'(\mathbb{F}_q) = 2^3 \cdot 3^2$.

(1) $P' = (12, 0)$ with $ord(P') = 2$

- $K_0 = [0]P' = \infty$, $K_1 = 1P'$.

$\Rightarrow n_a \equiv 0 \pmod 2$.

(2) $P' = (5, 20)$ with $ord(P') = 3$

- $K_0 = \infty$, $K_1 = (5, 20)$, $K_2 = (5, 47)$.

$\Rightarrow n_a \equiv 2 \pmod 3$.

(3) $P' = (13, 1)$ with $ord(P') = 9$

- $K_2 = (36, 53)$, $K_5 = (18, 31)$,
  $K_8 = (13, 66)$.

$\Rightarrow n_a \equiv 5 \pmod 9$.

CRT: $n_a \equiv 14 \pmod{18}$

Countermeasures

1. **Public key validation**
   - Alice verifies that the point send by Bob lies on the curve. I.e. for $B = (x_B, y_B)$, she checks
   $$y_B^2 \stackrel{?}{=} x_B^3 + ax_B + b.$$

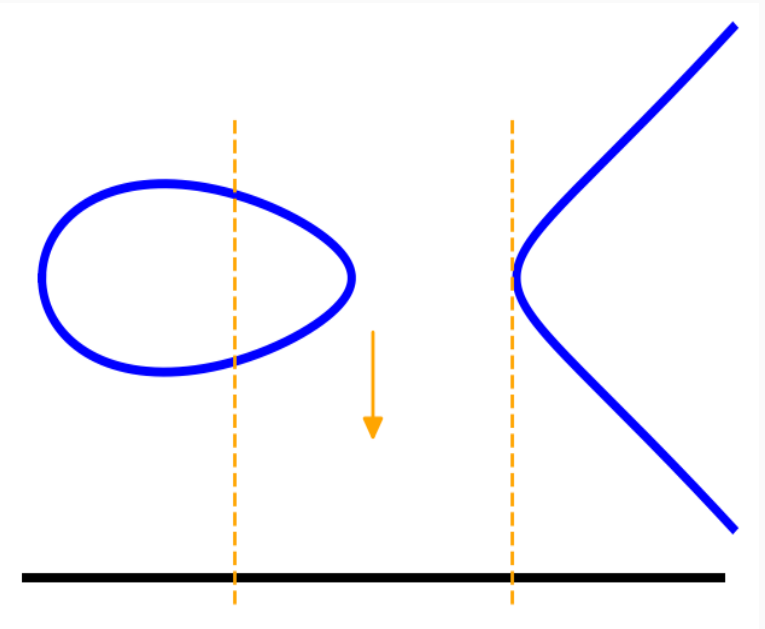
   $\rightarrow$ Cheap and easy modification
2. **x-only arithmetic**
   - Scalar multiplication is well defined for x-coordinates, i.e. there exist formulas that given $x(P)$ and N, output $x([N]P)$ (next slide)
   - Alice and Bob only publish the x-coordinates x_A of A and x_B of B. This is enough to find the x-coordinate x_{AB} of K_{AB}.
   - Formulas for x-only arithmetic implicitly check that a point is on the correct curve, and they are often faster than "normal" addition formulas.

   $\rightarrow$ Elegant solution

Digression: x-only arithmetic

Geometric description

- projection $\pi : E \to \mathbb{P}^1$
  $\pi(P) = x_P$, where $P = (x_P, y_P)$
- $\pi(-P) = \pi(P)$ for all $P \in E$.
- $\#\pi^{-1}(x_0) = 2$, unless
  $(x_0, 0) \in E[2]$.



Remnants of the group structure

- ✓ Scalar multiplication
  $[N]x(P) = x([N]P)$
- ✓ Translation by 2-torsion points
  $x(Q) + x(P) = x(Q + P)$ if one of
  P, Q is 2-torsion.
- x Addition of points $x(P) + x(Q) =?$

Example $x([2]P)$**:**

$$\frac{(3x(P)^2+a)^2}{4(x(P)^3+ax(P)+b)} - 2x(P)$$

Summary and outlook

Elliptic curves in cryptography

This lecture on Elliptic curve Diffie-Hellman (ECDH)

- The underlying hardness assumption is ECDLP, it can be solved in time $O(\sqrt{N})$ for any elliptic curve (generic algorithms).
- Weil pairing $e_N : E[N] \times E[N] \to \mu_N$ allows to translate ECDLP to DLP when the embedding degree is small (MOV attack)
- In the implementation: prevent invalid curve attacks

More attacks on special parameters using fancy math

- Over $\mathbb{F}_q$ with $q = p^k$ and $k > 2$, there are better attacks than generic attack using Weil descent to obtain a DLP on an abelian variety (Gerhard Frey, 1998); or a variant of index calculus (Pierrick Gaudry, 2009)
- For E with $E(\mathbb{F}_p) = p$, the ECDLP can be solved in polynomial time by computing in the formal group of an elliptic curve.