

ALMA MATER STUDIORUM · UNIVERSITÀ DI BOLOGNA

---

SCUOLA DI SCIENZE  
Corso di Laurea Magistrale in Matematica

## **Il Sistema IBE di Boneh e Franklin**

Tesi di Laurea in Crittografia

**Relatore:**  
**Prof.**  
**DAVIDE ALIFFI**

**Candidato:**  
**FEDERICO COSLOVICH**

**Sessione Unica**  
**Anno Accademico 2020-2021**



# Introduzione

Nel 1976 Whitfield Diffie e Martin Hellman pubblicarono il famoso articolo *New Directions in Cryptography* con il quale rivoluzionarono il mondo della crittografia, dando un concreto inizio ai sistemi a chiave pubblica. Otto anni dopo, nel 1984, Adi Shamir propose lo studio di un nuovo tipo di sistema crittografico asimmetrico, un sistema basato sull'*identità*.

Un sistema di cifratura IBE (*Identity-Based Encryption Scheme*) si basa su un sistema crittografico a chiave pubblica, costituita però in questo caso da una stringa arbitraria. Invece di generare una coppia casuale di chiavi pubbliche e private e pubblicare la prima, l'utente utilizza come chiave pubblica la sua "identità", ovvero una combinazione di informazioni opportune (nome, indirizzo, numero dell'ufficio, numero telefonico...) che lo identifichino in maniera univoca, così che l'utente non possa ripudiarla in seguito e questa sia facilmente reperibile alle altre parti. In questo modo ad ogni coppia di utenti risulta possibile comunicare in sicurezza e verificare le reciproche firme digitali senza lo scambio di chiavi private o pubbliche, senza la necessità di mantenere una *key directory* e senza dover ricorrere ogni volta ai servizi di un ente esterno.

La funzione solitamente svolta da un'Autorità Centrale viene esplicitata da quello che si chiama un *Centro di Generazione di Chiavi* (KGC, *Key Generation Center*), il cui scopo principale è quello di verificare la corretta identità dei nuovi utenti e di fornire, quando questi si uniscono alla rete, una *smart card* personalizzata, che sostanzialmente permette la cifratura e decifratura di messaggi, e la generazione e la verifica di firme digitali, a prescindere dall'identità dell'altro utente con cui si sta interagendo.

Quando Alice vuole mandare un messaggio a Bob, firma il messaggio con la chiave privata presente nella sua scheda personale ( $sk_A$ ), lo cifra utilizzando la chiave pubblica di Bob ( $pk_B$ ) ottenuta dalla sua identità (per esempio nome e indirizzo della rete), aggiunge la propria identità al messaggio e lo invia. Quando Bob lo riceve, lo decifra utilizzando la chiave privata presente nella sua scheda e poi verifica la firma utilizzando l'identità del mittente come chiave di verifica.

Lo schema crittografico collega il messaggio con l'informazione necessaria per l'identificazione *id* mentre il possesso della carta lega effettivamente *id* con l'utente fisico.

Nella sua ideazione iniziale lo schema è ideale per un gruppo ristretto di utenti, come po-

---

trebbe essere quello composto dai dipendenti di una società, il cui quartier generale può fungere da centro affidabile per la generazione di chiavi. Rimane comunque funzionale anche su scala maggiore, dove un numero elevato di KGC può permetterne l'utilizzo a una grande quantità di utenti. Estendendone poi le potenzialità potrebbe porre le basi per una nuova metodologia di identificazione personale, grazie alla quale firmare elettronicamente assegni o documenti legali.

Uno schema basato sull'identità assomiglia a un sistema di posta ideale: tramite la conoscenza del nome e dell'indirizzo di una persona le si può mandare messaggi che solo lei può leggere, e si può verificare la firma che solo lei può aver fatto. Tutto ciò rende quindi gli aspetti crittografici della comunicazione pressoché trasparenti.

La sicurezza generale dello schema dipende da vari aspetti, tra cui ovviamente la sicurezza delle funzioni crittografiche utilizzate. A differenza di quanto avviene in un normale schema asimmetrico, ad un avversario viene data maggior potenza in un attacco al sistema, poiché gli è permesso ottenere le chiavi private di arbitrarie chiavi pubbliche, creando ad hoc delle "identità" fittizie collegate.

Al momento di aprire la questione nel 1984 Shamir propose già uno schema per la firma e l'autenticazione digitale, ma non uno per un sistema crittografico basato sull'identità. In seguito furono proposti vari sistemi Identity-Based ma non del tutto efficienti. Nel 2001 Boneh e Franklin proposero uno schema completamente funzionante con sicurezza *chosen ciphertext security in the random oracle model*, basato su un analogo del problema computazionale di Diffie-Hellman e che da un punto di vista tecnico-matematico utilizza la crittografia su curve ellittiche e la mappa bilineare *Weil Pairing*.

La grande diffusione in crittografia delle curve ellittiche su campi finiti è dovuta al fatto che è possibile implementare i principali crittosistemi sulle curve invece che su strutture numeriche, ottenendo una maggior sicurezza a parità di lunghezza di chiavi e di costo computazionale. Inoltre, grazie alla struttura di gruppo che riescono ad avere, le curve ellittiche sono anche utilizzate per escogitare strategie di fattorizzazione. Quest'ultimo aspetto fu introdotto nel 1987 da Hendrik Lenstra e rappresentò sicuramente una novità, in quanto era la prima volta che si utilizzavano tecniche geometriche per fattorizzare interi.

Il *Weil Pairing* è una mappa bilineare definita sul prodotto dei gruppi di torsione  $E[n] \times E[n]$  di una curva ellittica e ha immagini nel gruppo delle radici  $n$ -esime dell'unità. Questo è uno strumento molto utile ed efficace nello studio delle curve ellittiche e delle questioni crittografiche annesse. Per esempio, è utilizzato nella dimostrazione del Teorema di Hasse, si ritrova nella costruzione di schemi crittografici e può essere impiegato per attaccare il problema del logaritmo discreto su una curva ellittica, spostandolo dalla curva al gruppo moltiplicativo di un'estensione del campo su cui è definita.

Nel Capitolo 1 andremo a dare un rapido sguardo alle curve ellittiche e a come l'insieme dei

loro punti costituisca un gruppo. Dopo le prime definizioni su campi qualsiasi e su campi finiti (sui quali si basa l'utilizzo delle curve nella crittografia) e l'esposizione delle proprietà essenziali, ci soffermeremo su una minima descrizione dei gruppi di torsione e delle curve supersingolari, dato che sulle loro proprietà si basano sia la costruzione del *Weil Pairing* che varie proprietà del sistema IBE. Ovviamente, per mancanza di spazio e in quanto non strettamente necessaria, è stata omessa una buona parte di altri risultati preliminari necessari, tra cui, per esempio, svariate proprietà dell'endomorfismo di Frobenius.

L'obiettivo del Capitolo 2 è quello di dare una sintetica trattazione dei *divisori* su curve ellittiche e in particolare dei *divisori di funzioni*, grazie ai quali sarà poi possibile costruire la mappa bilineare *Weil Pairing* e dimostrarne le proprietà.

Nel Capitolo 3 viene chiarita l'idea sottostante a un sistema crittografico basato sull'identità, evidenziando le differenze sostanziali con un sistema a chiave pubblica e uno a chiave privata. Si mostra in parte anche l'idea proposta da Shamir per uno schema di firma digitale basato sull'identità. Vengono infine presentate le prime definizioni di un sistema IBE, che saranno poi adattate nel dettaglio al sistema di Boneh e Franklin.

Nel Capitolo 4 vengono presentate le definizioni dei vari livelli di sicurezza in un sistema a chiave pubblica, per poi estenderle a un sistema basato sull'identità. In questo modo si vanno a sottolineare le nozioni (minime) di sicurezza che deve soddisfare il sistema IBE, la cui dimostrazione viene trattata nel Capitolo 6.

Il fine del Capitolo 5 è descrivere l'*ipotesi bilineare di Diffie-Hellman*, una modifica dell'*ipotesi Computazionale* su cui si basa il sistema IBE di Boneh e Franklin. Inizialmente viene descritto il collegamento tra il logaritmo discreto sulle curve ellittiche e il logaritmo discreto su un'estensione del campo di base e viene approfondita la *separazione* tra l'ipotesi computazionale di Diffie-Hellman e l'ipotesi decisionale, ovvero si evidenzia che quest'ultima risulta risolubile in gruppi particolari (come le curve ellittiche supersingolari utilizzate nel sistema IBE). Da queste considerazioni sorge la necessità di introdurre questa nuova ipotesi per il sistema crittografico.

Infine nel Capitolo 6 viene descritto nel dettaglio il sistema di Boneh e Franklin. Ne viene dimostrata la sicurezza IND-ID-CCA e ne vengono delineati i parametri e le mappe specifici per l'attuazione.



# Indice

<b>Introduzione</b>	<b>i</b>
<b>1 Curve Ellittiche</b>	<b>3</b>
1.1 Punti di Torsione . . . . .	8
1.1.1 Weil Pairing . . . . .	10
1.2 Curve ellittiche su campi finiti . . . . .	12
1.3 Curve supersingolari . . . . .	13
<b>2 Divisori</b>	<b>17</b>
2.1 Definizioni ed esempi . . . . .	17
2.2 Weil Pairing . . . . .	26
2.3 Aspetto computazionale . . . . .	32
<b>3 Sistema basato sull'identità</b>	<b>35</b>
3.0.1 Firma digitale <i>Identity-Based</i> . . . . .	38
3.1 Prime definizioni . . . . .	42
<b>4 Sicurezza</b>	<b>45</b>
4.1 Definizioni di sicurezza . . . . .	49
4.2 Sicurezza in un sistema basato sull'identità . . . . .	52
4.2.1 IND-ID-CPA . . . . .	52
4.2.2 IND-ID-CCA . . . . .	53
<b>5 Bilinear Diffie Hellman Problem</b>	<b>55</b>
5.1 Applicazione alle Curve Ellittiche . . . . .	57
5.2 Ipotesi bilineare di Diffie-Hellman . . . . .	59
<b>6 Sistema IBE di Boneh e Franklin</b>	<b>61</b>
6.1 BasicIdent e sicurezza IND-ID-CPA . . . . .	61

## INDICE

---

6.2	FullIdent e sicurezza IND-ID-CCA . . . . .	70
6.3	Sistema IBE . . . . .	78
6.3.1	FullIdent' . . . . .	78
6.3.2	Weil Pairing Modificato . . . . .	80
6.3.3	Parametri BDH . . . . .	81
6.3.4	MapToPoint . . . . .	81

# Notazioni

In questo elaborato utilizziamo notazioni standard per descrivere algoritmi probabilistici ed esperimenti.

Se  $A$  è un algoritmo probabilistico, allora  $A(x_1, x_2, \dots; r)$  è il risultato dell'esecuzione di  $A$  con input  $x_1, x_2, \dots$  e bit casuale  $r$ . Denotiamo con  $y \leftarrow A(x_1, x_2, \dots)$  l'esperimento di prendere  $r$  casuale e di ottenere  $y$  come output di  $A(x_1, x_2, \dots; r)$ . Se  $S$  è un insieme finito allora  $x \leftarrow S$  indica l'operazione di prendere uniformemente un elemento in  $S$ . Se  $\alpha$  non è né un algoritmo né un insieme, allora  $x \leftarrow \alpha$  rappresenta semplicemente un'assegnazione. Si dice che  $y$  può essere un output di  $A(x_1, x_2, \dots)$  se esiste un qualche bit  $r$  tale che  $A(x_1, x_2, \dots; r) = y$ .

Ricordiamo che una funzione  $\epsilon : \mathbb{N} \rightarrow \mathbb{R}$  è *trascurabile* se per ogni costante  $c \geq 0$  esiste un intero  $k_c$  tale che  $\epsilon(k) \leq k^{-c}$  per ogni  $k \geq k_c$ .

## INDICE

---

# Capitolo 1

## Curve Ellittiche

Nel presente capitolo andremo a dare un rapido sguardo alle curve ellittiche e a come l'insieme dei loro punti costituisca un gruppo. Dopo le prime definizioni su campi qualsiasi e su campi finiti (sui quali si basa l'utilizzo delle curve nella crittografia) e l'esposizione delle proprietà essenziali, ci soffermeremo su una minima descrizione dei gruppi di torsione e delle curve supersingolari, dato che sulle loro proprietà si basano sia la costruzione del *Weil Pairing* che varie proprietà del sistema IBE. Ovviamente, per mancanza di spazio e in quanto non strettamente necessaria, è stata omessa una buona parte di altri risultati preliminari necessari, tra cui, per esempio, svariate proprietà dell'endomorfismo di Frobenius.

Consideriamo un campo  $K$  e  $a, b \in K$ . Una curva ellittica  $E$  definita su  $K$  è il luogo geometrico del piano affine  $K^2$  descritto dall'equazione

$$y^2 = x^3 + ax + b, \quad (1.1)$$

detta **equazione di Weierstrass** per la curva ellittica, a cui si aggiunge un punto speciale, il punto all'infinito  $\mathcal{O}$ :

$$E(K) := \{(x, y) \in K^2 \mid y^2 = x^3 + ax + b\} \cup \mathcal{O}. \quad (1.2)$$

Il campo  $K$  può essere di qualsiasi natura, gli esempi più frequenti considerano  $\mathbb{R}$ ,  $\mathbb{C}$ ,  $\mathbb{Q}$  oppure, come nel caso dell'applicazione delle curve ellittiche alla crittografia, un campo finito  $\mathbb{F}_p$  con  $p$  primo. Per una rappresentazione grafica generale è conveniente guardare la curva come sottoinsieme di  $\mathbb{R}^2$ . Nel piano reale, le curve ellittiche presentano sostanzialmente due tipi di curve, come mostrato nella Figura 1.1. In un caso il polinomio in  $x$  associato,  $x^3 + ax + b$ , ha tre radici reali distinte, nell'altro caso una sola.

In generale, i coefficienti  $a$  e  $b$  non possono essere presi casualmente ma devono essere tali

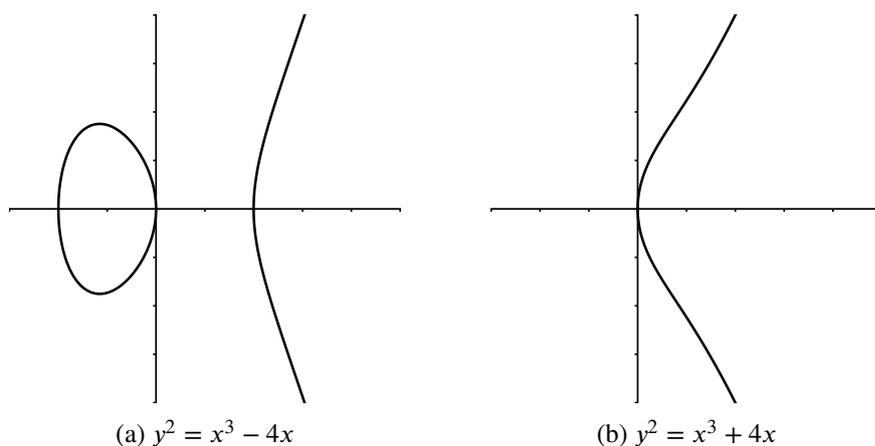


Figura 1.1: Curve ellittiche nel piano reale.

da soddisfare la condizione sul discriminante

$$4a^3 + 27b^2 \neq 0 \tag{1.3}$$

in maniera che la curva sia *non singolare*.

Riguardo all'equazione che descrive una curva ellittica, se ne può considerare una in forma più generale, data da

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6, \tag{1.4}$$

con  $a_1, \dots, a_6$  costanti. L'equazione sotto questa forma è detta **equazione di Weierstrass generalizzata** ed è utile quando si lavora con campi di caratteristica 2 o 3. Quando la caratteristica del campo è diversa da questi due valori allora, tramite opportune trasformazioni, è possibile ricondursi all'equazione 1.1.

Il punto all'infinito  $\mathcal{O}$  viene aggiunto alla curva per poterla poi caratterizzare con una struttura di gruppo. Può essere denotato anche con il simbolo  $\infty$ . In coordinate omogenee proiettive corrisponde al punto  $[0, 0, 1]$  e lo si può immaginare come posizionato all'"estremità" superiore (e inferiore) dell'asse  $y$ , dove "convergono" tutte le rette verticali.

L'aspetto interessante di una curva ellittica  $E$  è che possiede una struttura algebrica di gruppo. È quindi possibile definire un'operazione binaria  $+$  su  $E$ , per la quale l'insieme 1.2 diventa un gruppo abeliano, di cui  $\mathcal{O}$  rappresenta l'elemento neutro.

Presi due punti  $P = (x_1, y_1)$ ,  $Q = (x_2, y_2) \neq \mathcal{O}$  su  $E$ , per definire la somma  $P + Q$  distinguiamo tre casi:

- 1)  $x_1 \neq x_2$ : consideriamo  $L$ , retta per  $P$ ,  $Q$ . Per la condizione sul discriminante  $L$  interseca

$E$  in un terzo punto distinto  $R'$ . Sia  $R$  il suo simmetrico rispetto all'asse  $x$ . Si definisce

$$P + Q := R.$$

Da un punto di vista di coordinate, detto  $m$  il coefficiente angolare della retta  $L$ ,  $m = \frac{y_2 - y_1}{x_2 - x_1}$ , si ottiene

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

con  $R = (x_3, y_3)$ .

- 2)  $x_1 = x_2 \wedge y_1 \neq y_2$ : in questo caso la retta per i due punti è una retta verticale e quindi interseca  $E$  nel punto all'infinito  $\mathcal{O}$ :

$$P + Q = \mathcal{O}.$$

Quindi si ottiene  $Q = -P$ , ovvero  $Q$  è l'inverso di  $P$  in  $(E, +)$ .

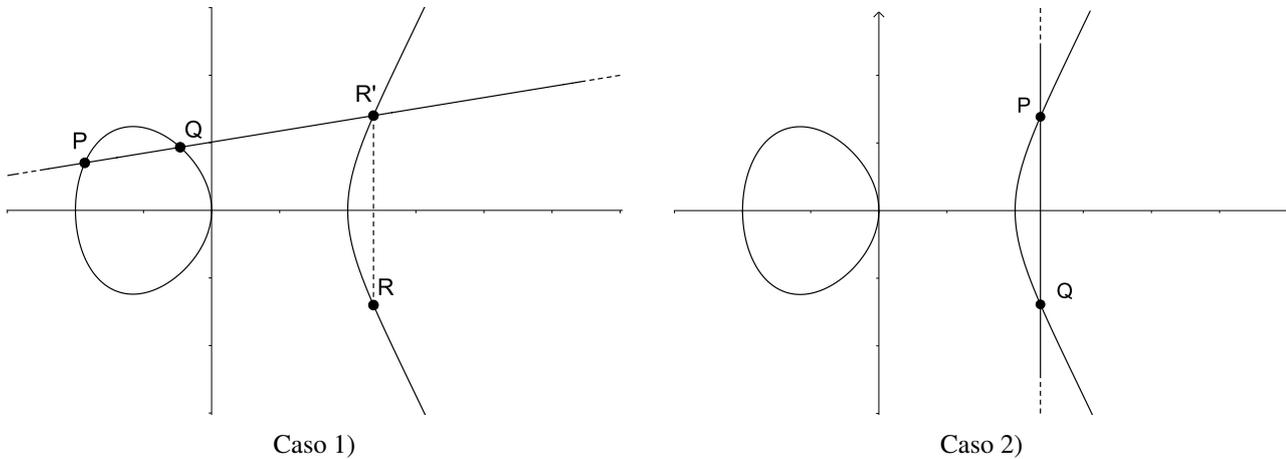


Figura 1.2: Somma in  $E$

- 3)  $x_1 = x_2 \wedge y_1 = y_2$ : il caso in cui  $P \equiv Q$  rappresenta il caso più importante nelle applicazioni crittografiche, da qui si ottiene il punto  $2P = P + P$ , elemento di  $\langle P \rangle \subseteq E$ . Sia  $L$  la tangente alla curva in  $P$  e sia  $R'$  il suo punto d'intersezione con  $E$ . Si definisce

$$2P = P + P = -R' = R.$$

Nel caso in cui  $y_1 \neq 0$ , detto  $m$  il coefficiente della tangente si ottiene  $m = \frac{3x_1^2+a}{2y_1}$  e

$$x_3 = m^2 - x_1 - x_2$$

$$y_3 = m(x_1 - x_3) - y_1$$

con  $R = (x_3, y_3)$ .

Nel caso in cui  $y_1 = 0$  la tangente è una retta verticale e si ottiene  $2P = \mathcal{O}$ , ovvero  $P$  è un punto (di torsione) di periodo 2.

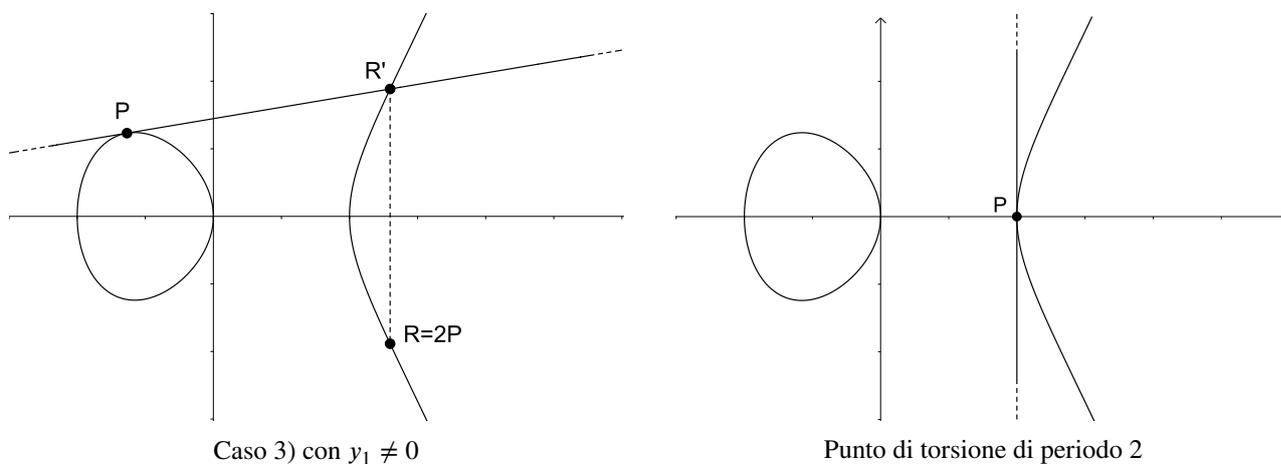


Figura 1.3: Somma in  $E$

Nel caso in cui  $Q = \mathcal{O}$ , la retta per  $P$  e  $\mathcal{O}$  è una retta verticale che interseca  $E$  in  $P'$ , punto simmetrico di  $P$  rispetto all'asse  $x$ . Applicando la riflessione a  $P'$  per ottenere  $P + \mathcal{O}$  si torna nuovamente sul punto  $P$ . Quindi

$$P + \mathcal{O} = \mathcal{O} + P = P, \quad \forall P \in E.$$

Quando  $P$  e  $Q$  hanno coordinate in un campo  $K$  che contiene  $a$  e  $b$ , allora anche  $P + Q$  ha coordinate in  $K$ . Di conseguenza  $E(K)$  è chiuso rispetto alla definizione di somma appena vista. Si dimostra che vale il seguente teorema.

**Teorema 1.1.** *Data una curva ellittica  $E$ , la somma di punti soddisfa le seguenti proprietà:*

1. *Commutatività:  $P_1 + P_2 = P_2 + P_1$  per ogni  $P_1, P_2 \in E$ .*
2. *Esistenza dell'elemento neutro:  $P + \mathcal{O} = P$  per ogni punto  $P$  su  $E$ .*
3. *Esistenza dell'inverso: dato  $P \in E$ , esiste un punto  $P' \in E$  tale che  $P + P' = \mathcal{O}$ . Solitamente  $P'$  si denota con  $-P$ .*

4. *Associatività*:  $(P_1 + P_2) + P_3 = P_1 + (P_2 + P_3)$  per ogni  $P_1, P_2, P_3$  in  $E$ .

Quindi, l'insieme dei punti di  $E$  forma un gruppo additivo abeliano con  $\mathcal{O}$  come elemento neutro.

Consideriamo ora  $p$  primo e come campo  $K = \mathbb{Z}_p$ .

**Definizione.** Una curva ellittica su  $\mathbb{Z}_p$  è l'insieme dei punti  $(x, y) \in \mathbb{Z}_p \times \mathbb{Z}_p$  tali che

$$y^2 = x^3 + ax + b \pmod{p} \tag{1.5}$$

con  $a, b \in \mathbb{Z}_p, 4a^3 + 27b^2 \not\equiv 0 \pmod{p}$ , più il punto all'infinito  $\mathcal{O}$ .

L'operazione di somma  $+$  è definita algebricamente, allo stesso modo del caso generale, con tutte le operazioni in  $\mathbb{Z}_p$ . Con questa operazione  $(E, +)$  è ancora un gruppo abeliano.

**Osservazione.** Nel caso di un campo finito si ha che anche l'insieme dei punti della curva ellittica  $E$  è finito. Nello specifico, se  $K = \mathbb{Z}_p$  vale  $|E| \leq p^2$ . In questo caso l'interpretazione geometrica precedente non vale più.

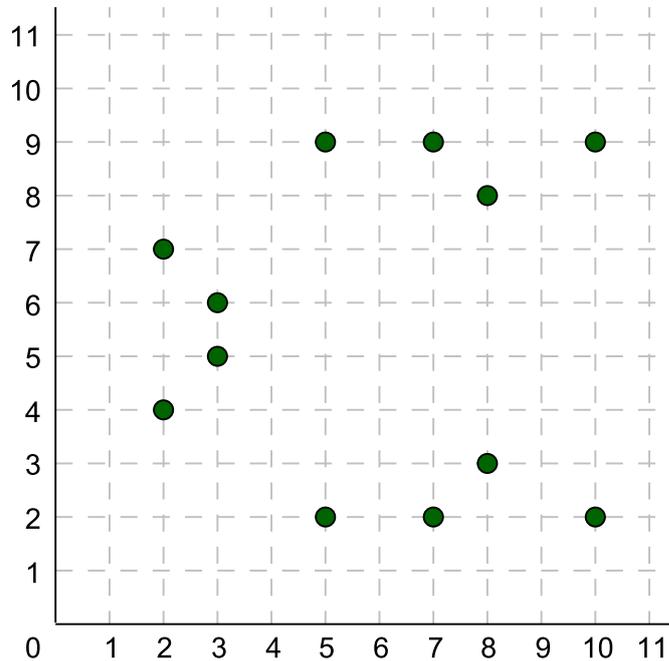


Figura 1.4: Curva ellittica  $y^2 = x^3 + x + 6 \pmod{11}$ .

**Esempio 1.0.1.** Consideriamo  $p = 11$  e la curva ellittica

$$E : y^2 = x^3 + x + 6 \text{ su } \mathbb{Z}_{11}.$$

Per trovare  $|E|$  e i suoi punti si può risolvere l'equazione modulare per ogni valore di  $x \in \mathbb{Z}_{11}$ , in maniera da ottenere un'equazione del tipo  $y^2 = x_P \pmod{11}$ . In questo modo ci si è ricondotti allo studio dei residui quadratici mod 11.

Studiando tutti i casi si ottiene  $|E| = 13$ , 12 punti "al finito" più  $\mathcal{O}$ :

$$E = \{(2, 4), (2, 7), (3, 5), (3, 6), (5, 2), (5, 9), (7, 2), (7, 9), (8, 3), (8, 8), (10, 2), (10, 9), \mathcal{O}\}.$$

In questo caso, poiché la cardinalità di  $E$  è 13, numero primo, si ha che  $E$  è ciclico e quindi vale l'isomorfismo  $E \simeq (\mathbb{Z}_{13}, +)$  e tutti i punti  $P \neq \mathcal{O}$  sono generatori.

### Multiplo di un punto e logaritmo discreto

Grazie alla definizione di somma data precedentemente è possibile definire il multiplo intero di un punto  $P$  di  $E$ . Preso  $k$  intero positivo, allora  $kP$  denota semplicemente la somma ripetuta (con  $k$  addendi)  $P + P + \dots + P$ . Se  $k < 0$ , allora  $kP = (-P) + \dots + (-P)$ . Per calcolare  $kP$  per un intero  $k$  grande, è però poco efficiente operare ripetutamente la somma con  $P$  ed è molto più veloce utilizzare il metodo delle *duplicazioni successive*. L'intero  $k$  viene quindi visto in base 2 e il multiplo  $kP$  viene computato calcolando prima i multipli  $2^i P$  e sommandone quelli opportuni. Per esempio, per calcolare  $19P$  si calcola:

$$2P, \quad 4P = 2P + 2P, \quad 8P = 4P + 4P, \quad 16P = 8P + 8P, \quad 19P = 16P + 2P + P.$$

L'unica difficoltà di questo calcolo risiede nel fatto che i moduli delle coordinate possono crescere molto rapidamente, cosa che però non avviene quando si lavora in un campo finito  $\mathbb{F}_p$ , grazie alla riduzione mod  $p$  ad ogni passaggio che mantiene le coordinate relativamente "piccole".

Dal punto di vista inverso, se si lavora su un campo finito, dati i punti  $P$  e  $kP$  è estremamente difficile risalire al valore di  $k$ . Questo quesito è detto **problema del logaritmo discreto per curve ellittiche** ed è alla base delle applicazioni crittografiche delle curve ellittiche.

## 1.1 Punti di Torsione

**Definizione.** Un punto  $P$  si dice *punto di torsione* se ha ordine finito, ovvero se esiste un intero  $k$  tale che  $kP = \mathcal{O}$ .

Sia  $E$  una curva ellittica definita su un campo  $K$  e sia  $n$  un intero positivo. Consideriamo

$$E[n] := \{P \in E(\overline{K}) \mid nP = \mathcal{O}\}, \tag{1.6}$$

dove  $\overline{K}$  indica la chiusura algebrica di  $K$ . Osserviamo che rappresenta un sottogruppo, il sottogruppo dei punti di torsione di ordine  $n$ , e non contiene solo punti a coordinate in  $K$  ma anche in  $\overline{K}$ .

**Esempio 1.1.1.** Consideriamo un campo tale che  $\text{char}K \neq 2, 3$  e cerchiamo di determinare  $E[2]$ .

L'equazione di una curva ellittica si può riscrivere nella forma

$$y^2 = (x - e_1)(x - e_2)(x - e_3),$$

con  $e_1, e_2, e_3 \in \overline{K}$ . Un punto  $P$  soddisfa  $2P = \infty$  se e solo se la tangente in  $P$  è una retta verticale, ovvero se  $y = 0$ . Quindi

$$E[2] = \{\infty, (e_1, 0), (e_2, 0), (e_3, 0)\}.$$

Si osserva che come gruppo è isomorfo a  $\mathbb{Z}_2 \oplus \mathbb{Z}_2$ .

Considerando ora  $E[3]$ . Si ha che  $3P = \infty$  se e solo se  $2P = -P$ . Sostituendo nelle equazioni di una generica curva ellittica si ottiene un'equazione in  $x$  di quarto grado che presenta 4 radici distinte in  $\overline{K}$  ed ogni valore di  $x$  porta a due valori distinti di  $y$ . Quindi si ottengono 8 punti "finiti" di ordine 3 a cui va aggiunto il punto all'infinito. Di conseguenza  $E[3]$  è un gruppo di ordine 9 in cui ogni elemento ha ordine 3 e perciò vale l'isomorfismo

$$E[3] \simeq \mathbb{Z}_3 \oplus \mathbb{Z}_3.$$

Si dimostra che in generale vale il seguente teorema:

**Teorema 1.2.** *Sia  $E$  una curva ellittica su un campo  $K$  e sia  $n$  un intero positivo. Se la caratteristica di  $K$  non divide  $n$ , oppure è 0, allora*

$$E[n] \simeq \mathbb{Z}_n \oplus \mathbb{Z}_n.$$

*Se la caratteristica di  $K$  è  $p > 0$  e  $p|n$ , sia  $n = p^r n'$  con  $p \nmid n'$ . Allora*

$$E[n] \simeq \mathbb{Z}_{n'} \oplus \mathbb{Z}_{n'} \quad \text{oppure} \quad \mathbb{Z}_n \oplus \mathbb{Z}_{n'}.$$

**Definizione.** Una curva ellittica  $E$  su un campo di caratteristica  $p$  è detta **ordinaria** se  $E[p] \simeq \mathbb{Z}_p$ . É detta **supersingolare** se  $E[p] \simeq 0$ .

**Osservazione.** Sia  $n$  un intero positivo non divisibile per la caratteristica di  $K$ . Per i risultati del Teorema 1.2 si può prendere una base  $\{\beta_1, \beta_2\}$  per  $E[n] \simeq \mathbb{Z}_n \oplus \mathbb{Z}_n$ . Di conseguenza ogni

elemento di  $E[n]$  si può esprimere nella forma  $m_1\beta_1 + m_2\beta_2$  con interi  $m_1, m_2$  determinati in maniera univoca mod  $n$ .

### 1.1.1 Weil Pairing

Il *Weil Pairing*, una mappa basata sui gruppi di torsione, è uno strumento molto utile nello studio delle curve ellittiche. Per esempio, è utile per dimostrare il Teorema di Hasse riguardo al numero di punti di una curva ellittica su un campo finito, può essere utilizzato per attaccare il problema del logaritmo discreto su una curva ellittica ed è anche utilizzato per costruire schemi crittografici (come quello IBE di Boneh e Franklin).

Sia  $E$  una curva ellittica su un campo  $K$  e sia  $n$  un intero non divisibile per la caratteristica. Allora  $E[n] \simeq \mathbb{Z}_n \oplus \mathbb{Z}_n$ . Sia

$$\mu_n = \{x \in \overline{K} \mid x^n = 1\}$$

il gruppo delle radici  $n$ -esime dell'unità in  $\overline{K}$ . Dato che la caratteristica di  $K$  non divide  $n$ , l'equazione  $x^n = 1$  non ha radici multiple e quindi ha  $n$  radici distinte in  $\overline{K}$ . Perciò  $\mu_n$  è un gruppo ciclico di ordine  $n$  e un suo qualsiasi generatore  $\zeta$  è detto radice *primitiva*. Equivalentemente  $\zeta^k = 1$  se e solo se  $n$  divide  $k$ .

Il *Weil Pairing* è definito sul prodotto dei gruppi di torsione  $E[n] \times E[n]$  a immagini nel gruppo delle radici  $n$ -esime dell'unità:

$$e_n : E[n] \times E[n] \longrightarrow \mu_n$$

La sua costruzione nel dettaglio e la dimostrazione delle sue proprietà saranno affrontati nel Capitolo 2.

#### **Teorema 1.3.**

*Sia  $E$  una curva ellittica definita su un campo  $K$  e sia  $n$  un intero positivo. Supponiamo che la caratteristica di  $K$  non divida  $n$ . Allora il Weil pairing  $e_n$  soddisfa le seguenti proprietà:*

(1)  $e_n$  è bilineare su entrambe le variabili, ovvero

$$e_n(aS, bT) = e_n(S, T)^{ab}$$

per ogni  $S, T \in E[n]$ ,  $a, b \in \mathbb{Z}^+$ .

(2)  $e_n$  è non degenera su ogni variabile. Questo significa che, se  $e_n(S, T) = 1$  per ogni  $T \in E[n]$  allora  $S = \infty$  e che, se  $e_n(S, T) = 1$  per ogni  $S \in E[n]$  allora  $T = \infty$ .

(3)  $e_n(T, T) = 1$  per ogni  $T \in E[n]$ .

- (4)  $e_n(T, S) = e_n(S, T)^{-1}$  per ogni  $S, T \in E[n]$ .
- (5)  $e_n(\sigma S, \sigma T) = \sigma(e_n(S, T))$  per ogni  $\sigma$  automorfismo di  $\overline{K}$  tale che  $\sigma$  sia l'identità sui coefficienti di  $E$  (se  $E$  è espressa tramite l'equazione di Weierstrass, si ha  $\sigma(A) = A$  e  $\sigma(B) = B$ ).
- (6)  $e_n(\alpha(S), \alpha(T)) = e_n(S, T)^{\deg(\alpha)}$  per ogni  $\alpha$  endomorfismo separabile di  $E$ . Se i coefficienti di  $E$  giacciono in un campo finito  $F_q$ , allora la proprietà vale anche quando  $\alpha$  è l'endomorfismo di Frobenius  $\phi_q$ .

Considerata una curva ellittica  $E$  su un campo finito  $\mathbb{F}_q$ , l'endomorfismo di Frobenius  $\phi_q$  è definito da

$$\begin{aligned}\phi_q : E(\overline{F}_q) &\longrightarrow E(\overline{F}_q) \\ \phi_q(x, y) &= (x^q, y^q).\end{aligned}$$

Questo esempio di endomorfismo ricopre un ruolo molto importante nella teoria delle curve ellittiche su campi finiti. In generale, considerata  $\mathbb{F}$  un'estensione qualsiasi di  $\mathbb{F}_q$ , l'endomorfismo di Frobenius permuta gli elementi di  $E(\mathbb{F})$  ed è l'identità sul sottogruppo  $E(\mathbb{F}_q)$ .

**Corollario 1.4.** Sia  $\{T_1, T_2\}$  una base di  $E[n]$ . Allora  $e_n(T_1, T_2)$  è una radice primitiva  $n$ -esima dell'unità.

**Dimostrazione:** Supponiamo  $e_n(T_1, T_2) = \zeta$  con  $\zeta^d = 1$ . Allora  $e_n(T_1, dT_2) = 1$  e, per le proprietà del Teorema 1.3, vale anche  $e_n(T_2, dT_2) = e_n(T_2, T_2)^d = 1$ . Sia  $S \in E[n]$ . Allora  $S = aT_1 + bT_2$  per opportuni interi  $a, b$ . Quindi

$$e_n(S, dT_2) = e_n(T_1, dT_2)^a e_n(T_2, dT_2)^b = 1.$$

Dato che questo vale per ogni  $S$ , per la proprietà (2), segue che  $dT_2 = \infty$ . Dato che  $dT_2 = \infty$  se e solo se  $n|d$ , segue immediatamente che  $\zeta$  è una radice primitiva  $n$ -esima dell'unità.  $\square$

**Corollario 1.5.** Se  $E[n] \subseteq E(K)$ , allora  $\mu_n \subset K$ .

**Dimostrazione:** Sia  $\sigma$  un automorfismo di  $\overline{K}$  che è l'identità su  $K$ . Siano  $T_1, T_2$  elementi di  $E[n]$  che ne formano una base. Per ipotesi devono avere coordinate in  $K$  e quindi  $\sigma T_1 = T_1$  e  $\sigma T_2 = T_2$ . Per le proprietà del Weil pairing vale

$$\zeta = e_n(T_1, T_2) = e_n(\sigma T_1, \sigma T_2) = \sigma(e_n(T_1, T_2)) = \sigma(\zeta).$$

Per il teorema fondamentale della Teoria di Galois, se un elemento  $x \in \overline{K}$  è fissato da tutti gli automorfismi  $\sigma$  di questo tipo, allora  $x \in K$ . Di conseguenza  $\zeta \in K$ . Per il Corollario 1.4 si ha che  $\zeta$  è una radice primitiva e quindi segue che  $\mu_n \subset K$ .  $\square$

Osserviamo che in generale i punti di  $E[n]$  possono avere coordinate in  $\overline{K}$ . L'ipotesi del Corollario 1.5 richiede che questi punti abbiano coordinate in  $K$ .

## 1.2 Curve ellittiche su campi finiti

Come già osservato precedentemente, una curva ellittica  $E$  definita su un campo finito  $\mathbb{F}$  è composta da un numero finito di punti.

Enunciamo ora, senza dimostrarli, i due principale risultati riguardo i gruppi  $E(\mathbb{F}_q)$ .

### **Teorema 1.6. (Teorema di struttura)**

*Sia  $E$  una curva ellittica definita su  $\mathbb{F}_q$ . Allora*

$$E(\mathbb{F}_q) \simeq \mathbb{Z}_n \quad \text{oppure} \quad \mathbb{Z}_{n_1} \oplus \mathbb{Z}_{n_2}$$

*per qualche intero  $n \geq 1$ , o per qualche intero  $n_1, n_2 \geq 1$  con  $n_2$  divisibile per  $n_1$ .*

### **Teorema 1.7. (Teorema di Hasse)**

*Sia  $E$  una curva ellittica definita su  $\mathbb{F}_q$ . Allora l'ordine di  $E(\mathbb{F}_q)$  soddisfa*

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

### **Determinare l'ordine del gruppo**

Il Teorema di Hasse fornisce delle limitazioni all'ordine del gruppo dei punti di una curva ellittica su un campo finito  $\mathbb{F}_q$ . Generalmente può essere utile determinare l'ordine del gruppo  $E(\mathbb{F}_{q^n})$  per un qualche  $n$ . Quando  $n = 1$  si può determinare l'ordine facendo un elenco dei punti o tramite qualche altro metodo elementare. L'aspetto interessante è che questo poi permette di determinare l'ordine per ogni  $n$ .

**Teorema 1.8.** *Sia  $|E(\mathbb{F}_q)| = q + 1 - a$ . Consideriamo il polinomio  $X^2 - aX + q = (X - \alpha)(X - \beta)$ .*

*Allora*

$$|E(\mathbb{F}_{q^n})| = q^n + 1 - (\alpha^n + \beta^n) \tag{1.7}$$

*per ogni  $n \geq 1$ .*

Il vantaggio del Teorema 1.8 sta nel fatto che permette di determinare l'ordine del gruppo molto velocemente però presenta lo svantaggio di richiedere che la curva sia definita su un campo finito "piccolo".

**Osservazione. (Ordine dei punti)**

Sia  $P \in E(\mathbb{F}_q)$  un punto di ordine  $k$ ; allora  $nP = \infty$  se e solo se l'ordine di  $P$  divide  $n$  e inoltre l'ordine di  $P$  divide sicuramente l'ordine del gruppo  $E(\mathbb{F}_q)$ .

Per il Teorema di Hasse,  $|E(\mathbb{F}_q)|$  giace in un intervallo di ampiezza  $4\sqrt{q}$ . Quindi, se si riesce a trovare un punto di ordine maggiore di  $4\sqrt{q}$ , allora ci può essere solamente un multiplo di questo ordine nell'intervallo corretto, e deve necessariamente essere proprio  $|E(\mathbb{F}_q)|$ . Nel caso in cui l'ordine del punto sia minore  $4\sqrt{q}$  si ottiene comunque una ristretta lista di possibilità per  $|E(\mathbb{F}_q)|$ . Utilizzando qualche altro punto si può restringere ulteriormente la lista in modo da riuscire a trovare  $|E(\mathbb{F}_q)|$ . Per trovare l'ordine di un punto uno dei metodi possibili è utilizzare il cosiddetto l'algoritmo *Baby Step, Giant Step*<sup>1</sup>.

### 1.3 Curve supersingolari

Una curva ellittica su un campo di caratteristica  $p$  è detta **supersingolare** se  $E[p] = \{\infty\}$ , ovvero se non ha punti di ordine  $p$ , nemmeno nella chiusura algebrica del campo. Le curve supersingolari presentano interessanti proprietà e saranno usate poi per costruire il sistema crittografico IBE di Boneh e Franklin. Nella presente sezione andiamo a presentare solo alcuni aspetti che le caratterizzano.

Il seguente risultato descrive un metodo utile per determinare se una curva ellittica su un campo finito sia supersingolare.

**Proposizione 1.9.** *Sia  $E$  una curva ellittica definita su  $\mathbb{F}_q$ , dove  $q$  è una potenza di un primo  $p$ . Sia  $a = q + 1 - |E(\mathbb{F}_q)|$ . Allora  $E$  è supersingolare se e solo se  $a \equiv 0 \pmod p$ , ovvero se e solo se  $|E(\mathbb{F}_q)| \equiv 1 \pmod p$ .*

**Dimostrazione:** Consideriamo  $X^2 - aX + q = (X - \alpha)(X - \beta)$ . Per il Teorema 1.8 si ha che

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

Considerata la successione  $s_n = \alpha^n + \beta^n$ , questa soddisfa la ricorrenza

$$s_0 = 2, \quad s_1 = a, \quad s_{n+1} = as_n - qs_{n-1}.$$

<sup>1</sup>È un algoritmo che richiede  $4q^{1/4}$  passaggi ed è utilizzato per calcolare l'ordine di un punto e per altri scopi, tra cui anche effettuare un attacco al logaritmo discreto su curve ellittiche. Per i dettagli si veda [WL].

Supponiamo  $a \equiv 0 \pmod p$ . Allora  $s_1 = a \equiv 0 \pmod p$ , e  $s_{n+1} \equiv 0 \pmod p$  per ogni  $n \geq 1$ . Di conseguenza vale

$$|E(\mathbb{F}_{q^n})| = q^n + 1 - s_n \equiv 1 \pmod p$$

e quindi per ogni  $n \geq 1$  non ci sono punti di ordine  $p$  in  $E(\mathbb{F}_{q^n})$ . Dato che  $\overline{\mathbb{F}_q} = \bigcup_{n \geq 1} \mathbb{F}_{q^n}$ , non ci sono punti di ordine  $p$  nemmeno in  $E(\overline{\mathbb{F}_q})$ . Quindi  $E$  è supersingolare.

Supponiamo ora  $a \not\equiv 0 \pmod p$ . Per la ricorrenza su  $s_n$  si ha che  $s_{n+1} \equiv as_n \pmod p$  per ogni  $n \geq 1$ . Poiché  $s_1 = a$ , si ottiene  $s_n \equiv a^n \pmod p$  per ogni  $n \geq 1$ . Quindi

$$|E(\mathbb{F}_{q^n})| = q^n + 1 - s_n \equiv 1 - a^n \pmod p.$$

Per il piccolo Teorema di Fermat vale  $a^{p-1} \equiv 1 \pmod p$ . Perciò  $E(\mathbb{F}_{q^{p-1}})$  ha ordine divisibile per  $p$  e quindi contiene un punto di ordine  $p$ . Questo implica che  $E$  non è supersingolare.

Per l'ultima parte del teorema osserviamo che

$$|E(\mathbb{F}_q)| \equiv q + 1 - a \equiv 1 - a \pmod p,$$

e allora  $|E(\mathbb{F}_q)| \equiv 1 \pmod p$  se e solo se  $a \equiv 0 \pmod p$ . □

**Corollario 1.10.** *Sia  $p \geq 5$  un primo e sia  $E$  definita su  $\mathbb{F}_p$ . Allora  $E$  è supersingolare se e solo se  $a = 0$ , ovvero se e solo se  $|E(\mathbb{F}_p)| = p + 1$ .*

**Dimostrazione:** Se  $a = 0$ , allora  $E$  è supersingolare per la proposizione. Inversamente, supponiamo che  $E$  sia singolare ma  $a \neq 0$ . Allora  $a \equiv 0 \pmod p$  implica che  $|a| \geq p$ . Per il Teorema di Hasse  $|a| \leq 2\sqrt{p}$ , quindi  $p \leq 2\sqrt{p}$ . Questo implica  $p \leq 4$ . □

Un modo per costruire curve supersingolari è dato dal seguente risultato.

**Proposizione 1.11.** *Sia  $q$  dispari e tale che  $q \equiv 2 \pmod 3$  e sia  $B \in \mathbb{F}_q^{\times}$ .<sup>2</sup> Allora la curva ellittica definita da  $y^2 = x^3 + B$  è supersingolare.*

**Dimostrazione:** Sia  $\psi : \mathbb{F}_q^{\times} \rightarrow \mathbb{F}_q^{\times}$  l'omomorfismo definito da  $\psi(x) = x^3$ . Dato che  $q - 1$  non è un multiplo di 3, non ci sono elementi di ordine 3 in  $\mathbb{F}_q^{\times}$  e perciò il nucleo di  $\psi$  è banale. Quindi  $\psi$  è iniettiva e deve essere anche suriettiva, essendo un endomorfismo. Nello specifico, possiamo affermare che ogni elemento di  $\mathbb{F}_q$  ha un'unica radice cubica in  $\mathbb{F}_q$ .

Per ogni  $y \in \mathbb{F}_q$ , esiste esattamente un elemento  $x \in \mathbb{F}_q$  tale che  $(x, y)$  appartenga alla curva, ovvero  $x$  è l'unica radice cubica di  $y^2 - B$ . Dato che ci sono  $q$  valori per  $y$ , otteniamo  $q$  punti. Considerando anche il punto  $\infty$  si ottiene

$$|E(\mathbb{F}_q)| = q + 1$$

<sup>2</sup>Dato un campo  $K$ , indichiamo con  $K^{\times}$  il gruppo moltiplicativo degli elementi non nulli.

da cui la tesi. □

Un'utile proprietà delle curve supersingolari è che il calcolo del multiplo di un punto può essere effettuato più velocemente del solito.

Consideriamo una curva supersingolare  $E$  definita su  $\mathbb{F}_q$  e sia  $P = (x, y)$  un punto di  $E(\mathbb{F}_{q^n})$  per un qualche  $n \geq 1$ . Sia  $k$  un intero e supponiamo di voler computare  $kP$ . Questo può essere fatto rapidamente tramite *duplicazioni successive* ma può essere fatto ancora più rapidamente sfruttando le proprietà delle curve supersingolari.

Supponiamo  $a = 0$ . Allora, si può dimostrare che  $\phi_q^2 + q = 0$ , da cui

$$q(x, y) = -\phi_q^2(x, y) = (x^{q^2}, -y^{q^2}).$$

Il calcolo di  $x^{q^2}$  e di  $-y^{q^2}$  rientra nell'aritmetica di un campo finito, che generalmente è più rapida dei calcoli su curve ellittiche.

**Osservazione.** La *supersingularità* significa che non ci sono punti di ordine  $p$  con coordinate nella chiusura algebrica del campo, quindi rappresenta una proprietà delle curve ellittiche definite su campi algebricamente chiusi. Se abbiamo due curve ellittiche  $E_1$  e  $E_2$  definite su un certo campo e tali che  $E_1$  possa essere trasformata in  $E_2$  tramite un cambio di variabili definito su una qualche estensione del campo, allora  $E_1$  è supersingolare se e solo se  $E_2$  è supersingolare.

Per esempio, nella Proposizione ??, la curva  $y_1^2 = x_1^3 + B$  può essere trasformata in  $y_2^2 = x_2^3 + 1$  grazie alla trasformazione  $x_2 = x_1/B^{1/3}$ ,  $y_2 = y_1/B^{1/2}$ . Di conseguenza anche la curva

$$y^2 = x^3 + 1$$

è supersingolare nel caso  $q \equiv 2 \pmod{3}$  e sarebbe bastato basare la dimostrazione della proposizione su questa curva specifica.

Mostriamo un'ultima proprietà delle curve *supersingolari*, che ne caratterizza fortemente i corrispondenti gruppi di torsione.

**Proposizione 1.12.** *Sia  $E$  una curva ellittica definita su  $\mathbb{F}_q$  e supponiamo  $a = q + 1 - |E(\mathbb{F}_q)| = 0$ . Sia  $N$  un intero positivo.*

*Se esiste un punto  $P \in E(\mathbb{F}_q)$  di ordine  $N$ , allora  $E[N] \subseteq E(\mathbb{F}_{q^2})$ .*

**Dimostrazione:** L'endomorfismo di Frobenius  $\phi_q$  soddisfa  $\phi_q^2 - a\phi_q + q = 0$ . Dato che  $a = 0$ , la relazione diventa

$$\phi_q^2 = -q.$$

### 1.3. CURVE SUPERSINGOLARI

---

Sia  $S \in E[N]$ . Dato che  $|E(\mathbb{F}_q)| = q + 1$  e dato che esiste un punto di ordine  $N$ , si ottiene  $N | q + 1$  o, equivalentemente,  $-q \equiv 1 \pmod{N}$ . Quindi

$$\phi_q^2(S) = -qS = 1 \cdot S.$$

Per le proprietà dell'endomorfismo di Frobenius segue che  $S \in E(\mathbb{F}_{q^2})$ . □

Quindi, grazie a questi risultati su questo tipo particolare di curve, il problema del logaritmo discreto su curve ellittiche *supersingolari*  $E(\mathbb{F}_q)$  con  $a = 0$  può essere riportato al calcolo del logaritmo discreto su  $\mathbb{F}_{q^2}^\times$  che risulta molto più semplice<sup>3</sup>.

---

<sup>3</sup>Questo trasporto del *logaritmo discreto* da un gruppo ad un altro è attuato tramite la tecnica nota come *MOV reduction*, per i dettagli si veda [MOV].

## Capitolo 2

# Divisori

L'obiettivo di questo capitolo è quello di dare una sintetica trattazione dei *divisori* su curve ellittiche e in particolare dei *divisori di funzioni*, grazie ai quali sarà poi possibile costruire la mappa bilineare *Weil Pairing* e dimostrarne le proprietà.

### 2.1 Definizioni ed esempi

Sia  $E$  una curva ellittica definita su un campo  $K$ . Preso un punto  $P \in E(\overline{K})$ , definiamo un simbolo formale  $[P]$ . Un **divisore**  $D$  su  $E$  è una combinazione lineare finita di tali simboli, a coefficienti interi:

$$D = \sum_j a_j [P_j], \quad a_j \in \mathbb{Z}$$

Un divisore, quindi, è un elemento del gruppo abeliano libero generato dai simboli  $[P]$ . Il gruppo dei divisori è denotato con  $\text{Div}(E)$ . Si definiscono **grado** e **somma** di un divisore con

$$\begin{aligned} \deg\left(\sum_j a_j [P_j]\right) &= \sum_j a_j \in \mathbb{Z} \\ \text{sum}\left(\sum_j a_j [P_j]\right) &= \sum_j a_j P_j \in E(\overline{K}) \end{aligned}$$

La funzione *somma* semplicemente usa l'operazione su  $E$  per sommare i punti che si trovano all'interno dei simboli.

I divisori di grado 0 formano un importante sottogruppo di  $\text{Div}(E)$ , denotato con  $\text{Div}^0(E)$ . Con la funzione *somma* si ottiene un omomorfismo suriettivo

$$\text{sum} : \text{Div}^0(E) \longrightarrow E(\overline{K})$$

## 2.1. DEFINIZIONI ED ESEMPI

---

La suriettività vale perché, preso  $P \in E(\overline{K})$  e considerato il divisore  $[P] - [\infty]$  si ha

$$\text{sum}([P] - [\infty]) = P$$

Considerando il nucleo dell'omomorfismo, invece, questo consiste dei cosiddetti *divisori di funzioni*, che andremo ora a descrivere e a caratterizzare con il teorema 2.2.

Supponiamo che  $E$  sia definita dall'**equazione di Weierstrass**  $y^2 = x^3 + Ax + B$ .

Una **funzione su  $E$**  è una funzione razionale

$$f(x, y) \in \overline{K}(x, y)$$

definita su almeno un punto di  $E(\overline{K})$  e a valori in  $\overline{K} \cup \{\infty\}$  (quindi, per esempio, la funzione razionale  $\frac{1}{y^2 - x^3 - Ax - B}$  non va bene).

**Osservazione.** Lavorando con funzioni razionali può essere che l'espressione della funzione non sia definita in alcuni punti. Applicando però la funzione non su punti generici ma sui punti di una curva ellittica può succedere che in quelli stessi punti la funzione sia definita. Per descrivere meglio questo tecnicismo è meglio considerare degli esempi:

Consideriamo la curva ellittica  $E$  definita da  $y^2 = x^3 - x$ . La funzione  $f(x, y) = \frac{x}{y}$  non è definita in  $(0, 0)$ . Tuttavia, su  $E$  vale

$$\frac{x}{y} = \frac{y}{x^2 - 1},$$

espressione che è definita su  $(0, 0)$  e assume valore 0 nel punto.

Analogamente, la funzione  $y/x$  può essere modificata in  $(x^2 - 1)/y$  che in  $(0, 0)$  assume valore  $\infty$ .

Si può dimostrare che una funzione razionale può sempre essere trasformata in questo modo in maniera da ottenere un'espressione che non dà  $0/0$  ma un valore in  $\overline{K} \cup \{\infty\}$  univocamente determinato.

Proseguendo con le definizioni, una funzione ha uno **zero** nel punto  $P$  se in  $P$  assume il valore 0, ha un **polo** nel punto  $P$  se in esso assume il valore  $\infty$ . Nello specifico ci serve definire l'ordine di uno zero e di un polo.

**Definizione.** Sia  $P$  un punto; si può dimostrare che esiste una funzione  $u_P$ , detta **uniformatore** in  $P$  (*uniformizer*), con  $u_P(P) = 0$  e tale che ogni funzione razionale  $f(x, y)$  può essere espressa nella forma

$$f = u_P^r g, \quad \text{con } r \in \mathbb{Z} \quad \text{e} \quad g(P) \neq 0, \infty.$$

Si definisce **ordine** di  $f$  in  $P$  con

$$\text{ord}_P(f) = r.$$

**Esempio 2.1.1.** Consideriamo la curva  $y^2 = x^3 - x$  e mostriamo che la funzione  $y$  è un uniformatore in  $(0, 0)$ . Sui punti della curva, presa  $f(x, y) = x$ , si ha

$$x = y^2 \frac{1}{x^2 - 1},$$

e  $\frac{1}{x^2 - 1}$  è non nullo e finito in  $(0, 0)$ . Quindi

$$\text{ord}_{(0,0)}(x) = 2.$$

Considerando altre funzioni razionali, in maniera analoga si ottiene

$$\text{ord}_{(0,0)}(x/y) = 1.$$

Quest'ultimo risultato è concorde con i precedenti calcoli che mostravano che la funzione  $x/y$  su  $E$  si annulla in  $(0, 0)$ .

Osserviamo inoltre che  $\text{ord}_{(0,0)}(y) = 1$ .

**Osservazione.** Su un qualsiasi punto finito  $P = (x_0, y_0)$  su una curva ellittica, l'uniformatore  $u_P$  può essere ottenuto dall'equazione della retta passante per  $P$  e non tangente alla curva nel punto. Una scelta immediata è  $u_P = x - x_0$  quando  $y_0 \neq 0$  e  $u_P = y$  se  $y_0 = 0$ .

**Esempio 2.1.2.** Consideriamo il punto  $P = (-2, 8)$  sulla curva  $y^2 = x^3 + 72$ .

La retta  $x + 2 = 0$  passa per  $P$  e quindi possiamo prendere  $u_P(x, y) = x + 2$ . La funzione

$$f(x, y) = x + y - 6$$

si annulla in  $P$ . Calcoliamo l'ordine di  $P$  come zero di  $f$ :

L'equazione della curva può essere riscritta come

$$(y + 8)(y - 8) = (x + 2)^3 - 6(x + 2)^2 + 12(x + 2),$$

da cui si può riscrivere  $f$

$$f(x, y) = (x + 2) + (y - 8) = \underbrace{(x + 2)}_{u_P^1} \underbrace{\left(1 + \frac{(x + 2)^2 - 6(x + 2) + 12}{y + 8}\right)}_{g \text{ con } g(P) \neq 0, \infty}$$

e quindi vale  $\text{ord}_P(f) = 1$ .

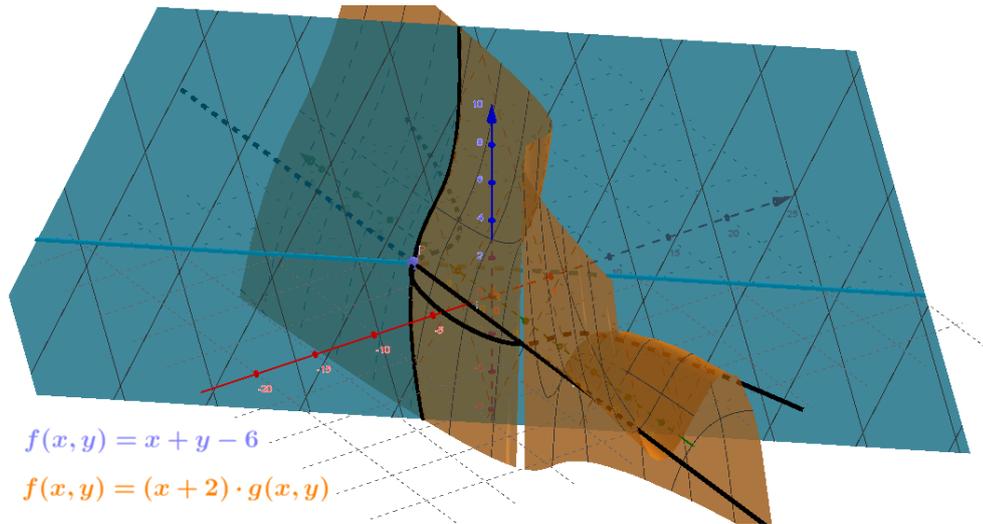


Figura 2.1: Le due superfici descritte dalle due espressioni di  $f(x, y)$  dell'Esempio 2.1.2.

Volendo dare un'interpretazione geometrica dell'ordine di una funzione in un punto  $P$ , osserviamo le Figure 2.1 e 2.2.

Nella prima figura sono rappresentate le due superfici descritte dalle due espressioni di  $f(x, y)$ , che ovviamente presentano lo stesso comportamento sui punti dello spazio tali che  $y^2 = x^3 + 72$ . Nel grafico è evidenziata la curva ottenuta come intersezione delle due superfici (che chiamiamo  $C$ ), che coincide con i punti le cui coordinate soddisfano l'equazione della curva.

La superficie descritta da  $f(x, y) = x + y - 6$  contiene la retta nel piano  $xy$   $x + y - 6 = 0$  mentre quella descritta dall'espressione trasformata  $f(x, y) = (x + 2) \cdot g(x, y)$  contiene la retta ottenuta da dall'uniformatore  $u_P$ , ovvero  $x + 2 = 0$ .

Si può osservare che entrambe le superfici (e le loro parti sul piano  $xy$ , Figura 2.3 (a)) sono secanti alla curva ellittica nel punto  $P$ , come lo è anche la curva nello spazio  $C$ . Questa considerazione è concorde col fatto che  $\text{ord}_P(f) = 1$ .

Se ora consideriamo la funzione  $t(x, y) = 3\frac{x+2}{4} - y + 8$ , questa può essere riscritta come

$$t(x, y) = (x + 2)^2 \cdot g_t(x, y) \quad \text{con } g_t(P) \neq 0,$$

da cui si ottiene  $\text{ord}_P(t) = 2$ .

Valgono delle considerazioni analoghe a quelle fatte per  $f(x, y)$ , ma in questo caso si osserva che le due superfici, come la curva ottenuta dalla loro intersezione, sono tangenti alla curva ellittica nel punto  $P$ , concordemente al fatto che l'ordine di  $t$  nel punto è 2. Di conseguenza, intersecando la superficie  $t(x, y) = 3\frac{x+2}{4} - y + 8$  col piano  $xy$  si ha che la retta ottenuta è proprio la tangente alla curva ellittica nel punto  $P$  (Figura 2.3 (b)).

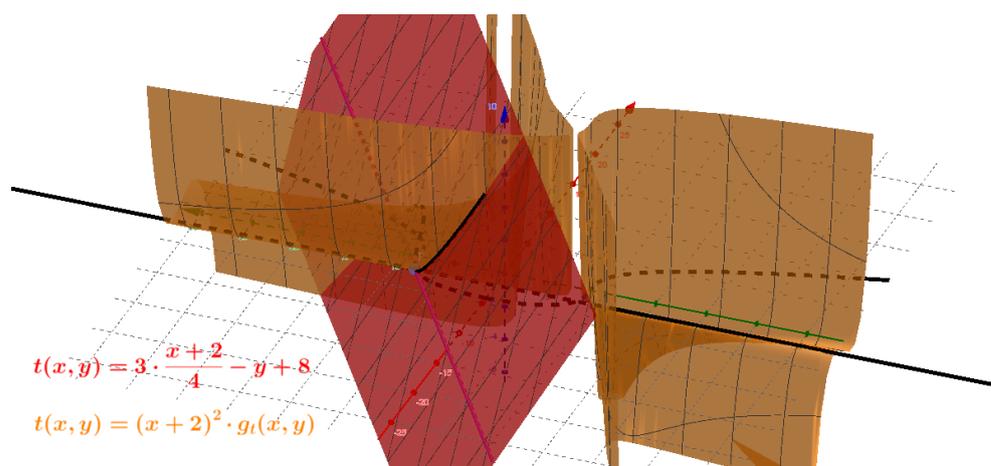
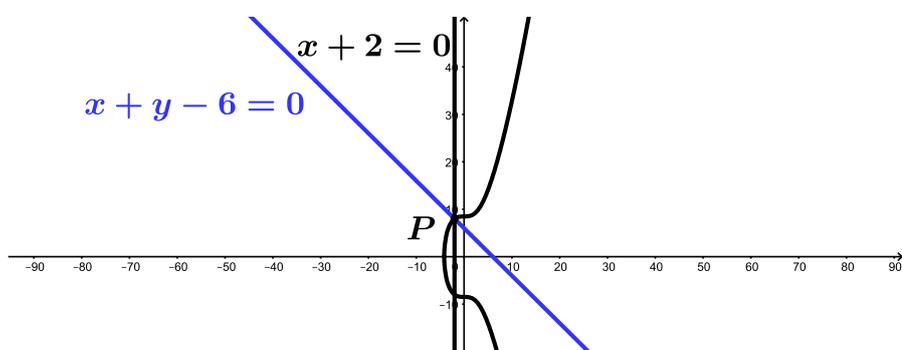
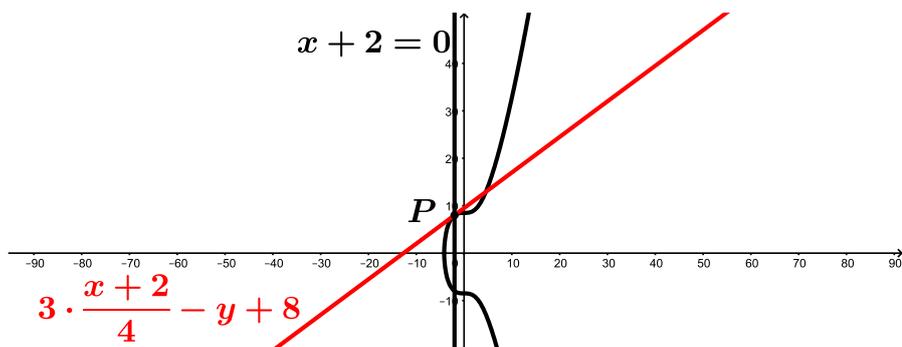


Figura 2.2: Le due superfici descritte dalle due espressioni di  $t(x, y)$  dell'Esempio 2.1.2.



(a) In blu la retta  $f(x, y) = 0$ .



(b) In rosso la retta  $t(x, y) = 0$ .

Figura 2.3: La rappresentazione sul piano  $xy$  delle due rappresentazioni grafiche dell'Esempio 2.1.2.

**Definizione.** Data una funzione  $f$  su  $E$ , non identicamente nulla, si definisce **divisore** di  $f$

$$\text{div}(f) = \sum_{P \in E(\bar{K})} \text{ord}_P(f)[P] \in \text{Div}(E).$$

Come dimostrato nella seguente proposizione, questa è una somma finita e di conseguenza definisce effettivamente un divisore.

**Proposizione 2.1.** *Sia  $E$  una curva ellittica e sia  $f$  una funzione su  $E$  non identicamente nulla. Allora*

1.  $f$  ha un numero finito di zeri e di poli
2.  $\deg(\text{div}(f)) = 0$
3. se  $f$  non ha né zeri né poli (e quindi  $\text{div}(f) = 0$ ), allora  $f$  è costante

Osserviamo che è importante considerare punti a coordinate in  $\overline{K}$ . Infatti, è facile costruire esempi di funzioni non costanti senza zeri o poli in punti di  $E(K)$  o esempi di funzioni che presentano zeri ma non poli in  $E(K)$ .

I divisori di funzione sono detti **divisori principali**.

**Esempio 2.1.3.** Supponiamo che  $P_1, P_2, P_3$  siano tre punti di  $E$  allineati lungo la retta  $ax + by + c = 0$ . Allora la funzione

$$f(x, y) = ax + by + c$$

si annulla in  $P_1, P_2$  e  $P_3$ . Se  $b \neq 0$  allora  $f$  ha un polo triplo in  $\infty$ . Di conseguenza vale

$$\text{div}(ax + by + c) = [P_1] + [P_2] + [P_3] - 3[\infty].$$

La retta passante per  $P_3 = (x_3, y_3)$  e  $-P_3 = (x_3, -y_3)$  è  $x - x_3 = 0$ . Il divisore della funzione  $x - x_3$  è

$$\text{div}(x - x_3) = [P_3] + [-P_3] - 2[\infty]. \quad (2.1)$$

Di conseguenza

$$\text{div}\left(\frac{ax + by + c}{x - x_3}\right) = \text{div}(ax + by + c) - \text{div}(x - x_3) = [P_1] + [P_2] - [-P_3] - [\infty].$$

Poiché su  $E$  vale  $P_1 + P_2 = -P_3$ , allora si può riscrivere

$$[P_1] + [P_2] = [P_1 + P_2] + [\infty] + \text{div}\left(\frac{ax + by + c}{x - x_3}\right) \quad (2.2)$$

Quindi, in generale, considerati due divisori del tipo  $[P_1], [P_2]$  la loro somma si può riscrivere come il divisore della somma dei due punti più il divisore  $[\infty]$  più il divisore di un'opportuna funzione.

**Teorema 2.2.**

Sia  $E$  una curva ellittica e sia  $D$  un divisore su  $E$  con  $\deg(D) = 0$ . Allora esiste una funzione  $f$  su  $E$  con

$$\operatorname{div}(f) = D$$

se e solo se

$$\operatorname{sum}(D) = \infty$$

Ovvero, dato  $D \in \operatorname{Div}^0(E)$ ,  $D$  è un divisore principale se e solo se  $\operatorname{sum}(D) = \infty$ .

**Dimostrazione:** Abbiamo visto nell'Esempio 2.1.3 che una somma  $[P_1] + [P_2]$  può essere sostituita con

$$[P_1] + [P_2] = [P_1 + P_2] + [\infty] + \operatorname{div}(g_1),$$

con  $g_1$  funzione opportuna. Osserviamo anche che

$$\operatorname{sum}(\operatorname{div}(g_1)) = P_1 + P_2 - (P_1 + P_2) - \infty = \infty.$$

L'equazione (2.1) mostra che, quando  $P_1 + P_2 = \infty$ , si può scrivere

$$[P_1] + [P_2] = 2[\infty] + \operatorname{div}(g_2)$$

con  $g_2$  funzione opportuna (funzione con zeri in  $P_1$  e  $P_2$ ). Anche in questo caso vale

$$\operatorname{sum}(\operatorname{div}(g_2)) = \infty.$$

Considerazioni analoghe valgono anche riguardo alla somma dei termini con coefficiente negativo.

Di conseguenza, considerato un divisore  $D$  come nelle ipotesi, essendo esso definito da una somma finita di termini è possibile reiterare più volte i procedimenti appena evidenziati in maniera da vedere che esistono due punti  $P$  e  $Q$  su  $E$ , una funzione  $h$  e un intero  $n$  tali che

$$D = [P] - [Q] + n[\infty] + \operatorname{div}(h).$$

Inoltre, dato che  $h$  è il quoziente di prodotti di funzioni  $g$  con  $\operatorname{sum}(\operatorname{div}(g)) = \infty$ , allora vale anche

$$\operatorname{sum}(\operatorname{div}(h)) = \infty.$$

Per la Proposizione 2.1 si ha  $\deg(\operatorname{div}(h)) = 0$  e quindi

$$0 = \deg(D) = 1 - 1 + n + 0 = n.$$

Da cui otteniamo

$$D = [P] - [Q] + \operatorname{div}(h)$$

e quindi

$$\operatorname{sum}(D) = P - Q + \operatorname{sum}(\operatorname{div}(h)) = P - Q.$$

Supponiamo ora  $\operatorname{sum}(D) = \infty$ .

Allora  $P - Q = \infty$  e quindi  $P = Q$  e  $D = \operatorname{div}(h)$ . △

Inversamente, supponiamo  $D = \operatorname{div}(f)$  per una qualche funzione  $f$ . Allora

$$[P] - [Q] = \operatorname{div}(f/h).$$

Tramite il seguente lemma otteniamo che vale  $P = Q$  e quindi  $\operatorname{sum}(D) = \infty$ . □

**Lemma 2.3.** *Siano  $P, Q \in E()$  e supponiamo esista una funzione  $h$  su  $E$  con*

$$\operatorname{div}(h) = [P] - [Q].$$

Allora  $P = Q$ .

La dimostrazione è alquanto lunga e non prettamente necessaria ai fini del presente elaborato ed è stata quindi omessa. Per una sua trattazione si rimanda a [WL].

**Corollario 2.4.** *La mappa*

$$\operatorname{sum} : \operatorname{Div}^0(E)/(\operatorname{divisori\ principali}) \longrightarrow E(\overline{K})$$

*è un isomorfismo di gruppi.*

**Dimostrazione:** Come già osservato, da  $\operatorname{sum}([P] - [\infty]) = P$  segue che la mappa  $\operatorname{sum}$  da  $\operatorname{Div}^0(E)$  a  $E(\overline{K})$  è suriettiva. Il Teorema 2.2 afferma che il nucleo coincide con il sottogruppo dei divisori principali, da cui segue immediatamente la tesi. □

In altri termini, il Corollario evidenzia che la legge del gruppo su  $E(\overline{K})$  corrisponde alla legge del gruppo su  $\operatorname{Div}^0(E)$  quozientato sui divisori principali.

**Esempio 2.1.4.** La dimostrazione del Teorema 2.2 ci dà un algoritmo per trovare una funzione con divisore dato (di grado 0 e somma  $\infty$ ).

Consideriamo la curva ellittica  $E$  su  $\mathbf{F}_{11}$  data da

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

e sia

$$D = [(0, 0)] + [(2, 4)] + [(4, 5)] + [(6, 3)] - 4[\infty].$$

$D$  ha grado 0 e un facile calcolo mostra che  $\text{sum}(D) = \infty$ . Cerchiamo la funzione che mi definisce  $D$  come divisore.

La retta per  $(0, 0)$  e  $(2, 4)$  è  $y - 2x = 0$  ed è tangente alla curva  $E$  in  $(2, 4)$ . Si ha

$$\text{div}(y - 2x) = [(0, 0)] + 2[(2, 4)] - 3[\infty].$$

La retta verticale passante per  $(2, 4)$  è  $x - 2 = 0$  e vale

$$\text{div}(x - 2) = [(2, 4)] + [(2, -4)] - 2[\infty].$$

Di conseguenza

$$D = [(2, -4)] + \text{div}\left(\frac{y - 2x}{x - 2}\right) + [(4, 5)] + [(6, 3)] - 3[\infty].$$

Analogamente si ottiene

$$[(4, 5)] + [(6, 3)] = [(2, 4)] + [\infty] + \text{div}\left(\frac{y + x + 2}{x - 2}\right),$$

da cui

$$D = [(2, -4)] + \text{div}\left(\frac{y - 2x}{x - 2}\right) + [(2, 4)] + \text{div}\left(\frac{y + x + 2}{x - 2}\right) - 2[\infty].$$

Dall'espressione ottenuta in precedenza per  $\text{div}(x - 2)$  otteniamo

$$\begin{aligned} D &= \text{div}(x - 2) + \text{div}\left(\frac{y - 2x}{x - 2}\right) + \text{div}\left(\frac{y + x + 2}{x - 2}\right) \\ &= \text{div}\left(\frac{(y - 2x)(y + x + 2)}{x - 2}\right) \end{aligned}$$

Scomponendo il numeratore è poi possibile semplificare la funzione:

$$\begin{aligned} (y - 2x)(y + x + 2) &= y^2 - xy - 2x^2 + 2y - 4x \\ &= x^3 - xy - 2x^2 + 2y \quad (\text{poiché } y^2 = x^3 + 4x) \\ &= (x - 2)(x^2 - y). \end{aligned}$$

E quindi

$$D = \text{div}(x^2 - y).$$

□

## 2.2 Weil Pairing

In questa sezione andiamo a costruire il Weil Pairing e proviamo le sue proprietà base. Ricordiamo che  $n$  è un intero non divisibile per la caratteristica del campo  $K$  e che  $E$  è una curva ellittica tale che

$$E[n] \subseteq E(K).$$

Vogliamo costruire una mappa

$$e_n : E[n] \times E[n] \longrightarrow \mu_n,$$

dove  $\mu_n$  è l'insieme delle radici  $n$ -esime dell'unità in  $\overline{K}$ . Per il Corollario 1.5 l'assunzione  $E[n] \subseteq E(K)$  implica che  $\mu_n \subseteq K$ .

Sia  $T \in E[n]$ . Per il Teorema 2.2, esiste una funzione  $f$  tale che

$$\text{div}(f) = n[T] - n[\infty]. \quad (2.3)$$

Sia  $T' \in E[n^2]$  tale che  $nT' = T$ . Useremo il teorema 2.2 per mostrare che esiste una funzione  $g$  tale che

$$\text{div}(g) = \sum_{R \in E[n]} ([T' + R] - [R]).$$

Dobbiamo verificare che la somma dei punti del divisore sia  $\infty$ . Questo segue dal fatto che in  $E[n]$  ci sono  $n^2$  punti  $R$  (Teorema 1.2). I punti  $R$  nelle sommatorie  $\sum [T' + R]$  e  $\sum [R]$  si elidono a vicenda; di conseguenza la somma diventa  $n^2 T' = nT = \infty$ . Osserviamo che  $g$  non dipende dalla scelta di  $T'$  poiché due scelte diverse per  $T'$  differiscono per un elemento  $R \in E[n]$ .

Di conseguenza avremmo anche potuto scrivere

$$\text{div}(g) = \sum_{nT''=T} [T''] - \sum_{nR=\infty} [R].$$

Denotiamo con  $f \circ n$  la funzione che prende un punto, lo moltiplica per  $n$  e poi ci applica  $f$ . I

punti  $P = T' + R$  con  $R \in E[n]$  sono i punti tali che  $nP = T$ . Dall'equazione 2.3 segue che

$$\operatorname{div}(f \circ n) = n \left( \sum_{R \in E[n]} [T' + R] \right) - n \left( \sum_{R \in E[n]} [R] \right) = \operatorname{div}(g^n).$$

Quindi, a meno di un fattore moltiplicativo costante, possiamo assumere che

$$f \circ n = g^n.$$

Sia  $S \in E[n]$  e sia  $P \in E(\overline{K})$ . Allora

$$g(P + S)^n = f(n(P + S)) = f(nP) = g(P)^n.$$

Di conseguenza vale

$$\frac{g(P + S)}{g(P)} \in \mu_n.$$

L'ultima affermazione segue dal fatto che  $g(P + S)/g(P)$  è indipendente dalla scelta di  $P$ <sup>1</sup>.

**Definizione.** Definiamo il **Weil Pairing** con

$$e_n(S, T) = \frac{g(P + S)}{g(P)}. \quad (2.4)$$

Poiché  $g$  è determinata dal suo divisore a meno di un fattore scalare moltiplicativo, questa definizione non dipende dalla scelta di  $g$ . Osserviamo inoltre che l'equazione 2.4 è indipendente dalla scelta del punto ausiliario  $P$ .

Enunciamo e dimostriamo col seguente teorema le principali proprietà del Weil Pairing.

**Teorema 2.5.**

*Sia  $E$  una curva ellittica definita su un campo  $K$  e sia  $n$  un intero positivo. Supponiamo che la caratteristica di  $K$  non divida  $n$ . Allora il Weil pairing*

$$e_n : E[n] \times E[n] \longrightarrow \mu_n$$

soddisfa le seguenti proprietà:

1.  $e_n$  è bilineare su entrambe le variabili, ovvero

$$e_n(S_1 + S_2, T) = e_n(S_1, T)e_n(S_2, T)$$

---

<sup>1</sup>La prova formale è un po' tecnica e si basa su proprietà della topologia di Zariski.

$e$

$$e_n(S, T_1 + T_2) = e_n(S, T_1)e_n(S, T_2)$$

per ogni  $S, S_1, S_2, T, T_1, T_2 \in E[n]$ .

2.  $e_n$  è non degenerare su ogni variabile. Questo significa che, se  $e_n(S, T) = 1$  per ogni  $T \in E[n]$  allora  $S = \infty$  e che, se  $e_n(S, T) = 1$  per ogni  $S \in E[n]$  allora  $T = \infty$ .
3.  $e_n(T, T) = 1$  per ogni  $T \in E[n]$ .
4.  $e_n(T, S) = e_n(S, T)^{-1}$  per ogni  $S, T \in E[n]$ .
5.  $e_n(\sigma S, \sigma T) = \sigma(e_n(S, T))$  per ogni  $\sigma$  automorfismo di  $\overline{K}$  tale che  $\sigma$  è l'identità sui coefficienti di  $E$  (se  $E$  è espressa tramite l'equazione di Weierstrass, si ha  $\sigma(A) = A$  e  $\sigma(B) = B$ ).
6.  $e_n(\alpha(S), \alpha(T)) = e_n(S, T)^{\deg(\alpha)}$  per ogni  $\alpha$  endomorfismo separabile di  $E$ . Se i coefficienti di  $E$  giacciono in un campo finito  $F_q$ , allora la proprietà vale anche quando  $\alpha$  è l'endomorfismo di Frobenius  $\phi_q^2$ .

**Dimostrazione:** (1) Poiché  $e_n$  è indipendente dalla scelta di  $P$ , utilizziamo la 2.4 con  $P$  e  $P + S_1$  per ottenere

$$\begin{aligned} e_n(S_1, T)e_n(S_2, T) &= \frac{g(P + S_1)}{g(P)} \frac{g(P + S_1 + S_2)}{g(P + S_1)} \\ &= \frac{g(P + S_1 + S_2)}{g(P)} \\ &= e_n(S_1 + S_2, T). \end{aligned}$$

Questo prova la linearità nella prima variabile.

Supponiamo  $T_1, T_2, T_3 \in E[n]$  con  $T_1 + T_2 = T_3$ . Per  $1 \leq i \leq 3$ , siano  $f_i, g_i$  le funzioni utilizzate precedentemente per definire  $e_n(S, T_i)$ . Per il Teorema 2.2, esiste una funzione  $h$  tale che

$$\operatorname{div}(h) = [T_3] - [T_1] - [T_2] + [\infty].$$

L'Equazione 2.3 implica

$$\operatorname{div}\left(\frac{f_3}{f_1 f_2}\right) = n \operatorname{div}(h) = \operatorname{div}(h^n).$$

Quindi esiste una costante  $c \in \overline{K}^*$  tale che

$$f_3 = c f_1 f_2 h^n.$$

<sup>2</sup>In realtà vale per ogni endomorfismo, separabile o no; vedere [EA]

Di conseguenza si ottiene

$$g_3 = c^{1/n}(g_1)(g_2)(h \circ n).$$

Dalla definizione di  $e_n$  si ottiene

$$\begin{aligned} e_n(S, T_1 + T_2) &= \frac{g_3(P + S)}{g_3(P)} = \frac{g_1(P + S)}{g_1(P)} \frac{g_2(P + S)}{g_2(P)} \frac{h(n(P + S))}{h(nP)} \\ &= e_n(S, T_1)e_n(S, T_2), \end{aligned}$$

poiché  $nS = \infty$ , si ha  $h(n(P + S)) = h(nP)$ . Così si è dimostrata la linearità nella seconda variabile.

(2) Supponiamo  $T \in E[n]$  tale che  $e_n(S, T) = 1$  per ogni  $S \in E[n]$ . Questo implica che  $g(P + S) = g(P)$  per ogni  $P$  e per ogni  $S \in E[n]$ . Si può dimostrare che esiste una funzione  $h$  tale che  $g = h \circ n$ .<sup>3</sup> Di conseguenza vale

$$(h \circ n)^n = g^n = f \circ n.$$

Dato che la moltiplicazione per  $n$  è suriettiva su  $E(\overline{K})$ , si ha  $h^n = f$ . Quindi

$$n \operatorname{div}(h) = \operatorname{div}(f) = n[T] - n[\infty],$$

da cui  $\operatorname{div}(h) = [T] - [\infty]$ . Per il Teorema 2.2, si ottiene  $T = \infty$ . Questo prova una metà della proprietà (2). L'altra metà segue direttamente da (4) e dalla proprietà di essere non degenerare in  $T$ .

(3) Indichiamo con  $\tau_{jT}$  l'operazione di sommare  $jT$ ; quindi  $f \circ \tau_{jT}$  denota la funzione  $P \mapsto f(P + jT)$ . Il divisore di  $f \circ \tau_{jT}$  è  $n[T - jT] - n[-jT]$ . Quindi

$$\operatorname{div} \left( \prod_{j=0}^{n-1} f \circ \tau_{jT} \right) = \sum_{j=0}^{n-1} (n[(1-j)T] - n[-jT]) = 0.$$

Ovvero la produttoria

$$\prod_{j=0}^{n-1} f \circ \tau_{jT} \text{ è costante.}$$

L' $n$ -esima potenza della funzione  $\prod_{j=0}^{n-1} g \circ \tau_{jT'}$  (dove  $T'$  è tale che  $nT' = T$ ) è il prodotto

---

<sup>3</sup>Questa proprietà deriva da risultati su curve ellittiche su  $C$ . Nel dettaglio si veda [WL], Proposizione 9.34.

delle  $f$  di cui sopra, composto con la moltiplicazione per  $n$ , e quindi è costante:

$$\begin{aligned} \left( \prod_{j=0}^{n-1} g \circ \tau_{jT'} \right)^n &= \prod_{j=0}^{n-1} f \circ n \circ \tau_{jT'} \\ &= \prod_{j=0}^{n-1} f \circ \tau_{jT} \circ n \quad (\text{poiché } nT' = T). \end{aligned}$$

Provato che quest'ultimo prodotto è costante, segue che anche  $\prod_{j=0}^{n-1} g \circ \tau_{jT'}$  è costante.<sup>4</sup> Quindi assume lo stesso valore in  $P$  e in  $P + T'$ , e di conseguenza si ottiene

$$\prod_{j=0}^{n-1} g(P + T' + jT') = \prod_{j=0}^{n-1} g(P + jT').$$

Cancellando i termini comuni (e assumendo che  $P$  sia scelto in maniera tale che tutti i termini siano finiti e non nulli) si ottiene

$$g(P + nT') = g(P).$$

Poiché  $nT' = T$ , questo implica

$$e_n(T, T) = \frac{g(P + T)}{g(P)} = 1.$$

(4) Dalle proprietà (1) e (3) si ottiene

$$\begin{aligned} 1 &= e_n(S + T, S + T) = e_n(S, S)e_n(S, T)e_n(T, S)e_n(T, T) \\ &= e_n(S, T)e_n(T, S) \end{aligned}$$

da cui segue immediatamente

$$e_n(T, S) = e_n(S, T)^{-1}.$$

(5) Sia  $\sigma$  un automorfismo di  $\bar{K}$  tale che  $\sigma$  è l'identità sui coefficienti di  $E$ . Supponiamo di applicare  $\sigma$  ad ogni oggetto nella costruzione di  $e_n$ . Si ha

$$\text{div}(f^\sigma) = n[\sigma T] - n[\infty]$$

e similmente per  $g^\sigma$ , dove  $f^\sigma$  e  $g^\sigma$  denotano le funzioni ottenute applicando  $\sigma$  ai coefficienti delle

---

<sup>4</sup>Grazie alla connessione di  $E$  nella topologia di Zariski.

funzioni razionali che definiscono  $f$  e  $g$ . Di conseguenza

$$\sigma(e_n(S, T)) = \sigma\left(\frac{g(P+S)}{g(P)}\right) = \frac{g^\sigma(\sigma P + \sigma S)}{g^\sigma(\sigma P)} = e_n(\sigma S, \sigma T).$$

(6) Sia  $\{Q_1, \dots, Q_k\} = \text{Ker}(\alpha)$ . Dato che  $\alpha$  è un morfismo separabile,  $k = \text{deg}(\alpha)$ . Siano

$$\text{div}(f_T) = n[T] - n[\infty], \quad \text{div}(f_{\alpha(T)}) = n[\alpha(T)] - n[\infty]$$

e

$$g_T^n = f_T \circ n, \quad g_{\alpha(T)}^n = f_{\alpha(T)} \circ n.$$

Come in (3), denotiamo con  $\tau_Q$  l'operazione di sommare  $Q$ . Otteniamo

$$\text{div}(f_T \circ \tau_{-Q_i}) = n[T + Q_i] - n[Q_i].$$

Di conseguenza

$$\begin{aligned} \text{div}(f_{\alpha(T)} \circ \alpha) &= n \sum_{\alpha(T'')=\alpha(T)} [T''] - n \sum_{\alpha(Q)=\infty} [Q] \\ &= n \sum_i ([T + Q_i] - [Q_i]) \\ &= \text{div}\left(\prod_i (f_T \circ \tau_{-Q_i})\right). \end{aligned}$$

Per ogni  $i$ , prendiamo  $Q'_i$  con  $nQ'_i = Q_i$ . Allora

$$g_T(P - Q'_i)^n = f_T(nP - Q_i)$$

e di conseguenza

$$\begin{aligned} \text{div}\left(\prod_i (g_T \circ \tau_{-Q'_i})^n\right) &= \text{div}\left(\prod_i f_T \circ \tau_{-Q_i} \circ n\right) \\ &= \text{div}(f_{\alpha(T)} \circ \alpha \circ n) \\ &= \text{div}(f_{\alpha(T)} \circ n \circ \alpha) \\ &= \text{div}(g_{\alpha(T)} \circ \alpha)^n. \end{aligned}$$

Perciò

$$\prod_i g_T \circ \tau_{-Q'_i} \text{ e } g_{\alpha(T)} \circ \alpha$$

hanno gli stessi divisori e quindi differiscono per una costante  $C$ .

La definizione di  $e_n$  porta a

$$\begin{aligned}
 e_n(\alpha(S), \alpha(T)) &= \frac{g_{\alpha(T)}(\alpha(P + S))}{g_{\alpha(T)}(\alpha(P))} \\
 &= \prod_i \frac{g_T(P + S - Q'_i)}{g_T(P - Q'_i)} \quad \text{la costante } C \text{ si cancella} \\
 &= \prod_i e_n(S, T) \quad \text{poiché } P \text{ e } P - Q'_i \text{ danno lo stesso valore di } e_n \\
 &= e_n(S, T)^k = e_n(S, T)^{\deg(\alpha)}.
 \end{aligned}$$

Quando  $\alpha$  è l'endomorfismo di Frobenius  $\phi_q$ , la proprietà (5) implica

$$e_n(\phi_q(S), \phi_q(T)) = \phi_q(e_n(S, T)) = e_n(S, T)^q$$

poiché  $\phi_q$  è la potenza  $q$ -esima sugli elementi di  $\overline{\mathbf{F}}_q$ . Si ha che  $q = \deg(\phi_q)$ , che dimostra (6) nel caso  $\alpha = \phi_q$ .

Questo completa la dimostrazione del Teorema 2.5. □

## 2.3 Aspetto computazionale

Precedentemente abbiamo visto come esprimere un divisore di grado 0 e somma  $\infty$  come divisore di una funzione opportuna. Questo metodo è sufficiente a calcolare il Weil Pairing per esempi di piccole dimensioni. Per dimensioni maggiori conviene ricorrere a qualche accorgimento per evitare un quantità elevata di calcoli. Inoltre, la definizione data precedentemente per il Weil pairing coinvolge una funzione  $g$ , il cui divisore include contributi da tutti gli  $n^2$  punti di  $E[n]$ . Quando  $n$  è grande questo può portare a dagli ostacoli computazionali.

Si dimostra il seguente teorema, che dà una definizione alternativa del Weil Pairing  $e_n$ .

### **Teorema 2.6.**

*Siano  $S, T \in E[n]$ . Indichiamo con  $D_S$  e  $D_T$  i divisori di grado 0 tali che*

$$\text{sum}(D_S) = S \quad \text{e} \quad \text{sum}(D_T) = T$$

*e tali che  $D_S$  e  $D_T$  non abbiano punti in comune.*

*Siano  $f_S$  e  $f_T$  funzioni tali che*

$$\text{div}(f_S) = nD_S \quad \text{e} \quad \text{div}(f_T) = nD_T.$$

Allora il Weil Pairing è dato da

$$e_n(S, T) = \frac{f_T(D_S)}{f_S(D_T)}.$$

(Ricordiamo che  $f(\sum a_i [P_i]) = \prod f(P_i)^{a_i}$ .)

**Osservazione.** Una scelta naturale per i divisori è

$$D_S = [S] - [\infty], \quad D_T = [T + R] - [R]$$

per un qualche punto casuale  $R$ . In questo modo si ha

$$e_n(S, T) = \frac{f_S(R)f_T(S)}{f_S(T + R)f_T(\infty)}.$$

**Esempio 2.3.1.** Sia  $E$  una curva ellittica su  $\mathbf{F}_7$  definita da

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

Si ha

$$E(\mathbf{F}_7)[3] = \mathbb{Z}_3 \oplus \mathbb{Z}_3$$

che coincide con tutto  $E(\mathbf{F}_7)$ .

Calcoliamo

$$e_3((0, 3), (5, 1)) :$$

Per calcolare i due divisori che definiscono il Weil Pairing prendiamo  $R = (6, 1)$ , da cui otteniamo  $(5, 1) +_E (6, 1) = (3, 6)$ .

Siano

$$D_{(0,3)} = [(0, 3)] - [\infty], \quad D_{(5,1)} = [(3, 6)] - [(6, 1)].$$

Con qualche calcolo si può mostrare che

$$\operatorname{div}(y - 3) = 3D_{(0,3)}, \quad \operatorname{div}\left(\frac{4x - y + 1}{5x - y - 1}\right) = 3D_{(5,1)}.$$

Quindi possiamo prendere

$$f_{(0,3)} = y - 3, \quad f_{(5,1)} = \frac{4x - y + 1}{5x - y - 1}.$$

Si ottiene

$$f_{(0,3)}(D_{(5,1)}) = \frac{f_{(0,3)}(3, 6)}{f_{(0,3)}(6, 1)} = \frac{6 - 3}{1 - 3} \equiv 2 \pmod{7}.$$

Analogamente

$$f_{(5,1)}(D_{(0,3)}) = 4.$$

Perciò

$$e_3((0, 3), (5, 1)) = \frac{4}{2} \equiv 2 \pmod{7}$$

e, effettivamente, il numero 2 è una radice cubica dell'unità, dato che  $2^3 \equiv 2 \pmod{7}$ .

Riguardo al calcolo di  $f_{(5,1)}(\infty)$  ci sono vari modi per effettuarlo. In maniera intuitiva si può osservare che  $y$  ha un polo di ordine 3 in  $\infty$  mentre  $x$  ha un polo di ordine 2. Perciò, i termini  $-y$  al numeratore e al denominatore prevalgono quando  $(x, y) \rightarrow \infty$ , e quindi il rapporto tende a 1. Un procedimento differente consiste nel trasformare l'equazione della curva in forma omogenea e usare le coordinate proiettive<sup>5</sup>:

$$f_{(5,1)}(x_0 : x_1 : x_2) = \frac{x_0 + 4x_1 - x_2}{x_0 + 5x_2 - x_2}$$

e quindi

$$f_{(5,1)}(\infty) = f_{(5,1)}(0 : 0 : 1) = 1.$$

---

<sup>5</sup>Per una trattazione più approfondita delle curve ellittiche in coordinate proiettive si rimanda a [WL].

## Capitolo 3

# Sistema basato sull'identità

In questo capitolo viene chiarita l'idea sottostante a un sistema crittografico basato sull'identità, evidenziando le differenze sostanziali con un sistema a chiave pubblica e uno a chiave privata. Si mostra in parte anche l'idea proposta da Shamir per uno schema di firma digitale basato sull'identità. Vengono infine presentate le prime definizioni di un sistema IBE, che saranno poi adattate nel dettaglio al sistema di Boneh e Franklin.

Un sistema di cifratura IBE (*Identity-Based Encryption Scheme*), e più in generale un sistema crittografico basato sull'Identità, rappresenta un sistema crittografico asimmetrico che nacque dall'idea di creare un sistema crittografico a chiave pubblica dove però la chiave pubblica potesse essere una stringa arbitraria.

In questo modo ad ogni coppia di utenti risulta possibile comunicare in sicurezza e verificare le reciproche firme digitali senza lo scambio di chiave private o pubbliche, senza la necessità di mantenere una *key directory* e senza i servizi di un ente esterno.

Lo schema presuppone l'esistenza di un *Centro di Generazione di Chiavi* affidabile (KGC, *Key Generation Center*) il cui scopo principale è quello di verificare la corretta identità dei nuovi utenti e di fornire, quando questi si uniscono alla rete, una *smart card* personalizzata. Questa *smart card* deve disporre, oltre che degli opportuni hardware, di programmi per la cifratura e decifratura di messaggi e di programmi che permettano la generazione firme digitali e la verifica dell'autenticità dei messaggi che si ricevono, a prescindere dall'identità dell'altro utente con cui si sta interagendo.

In questo modo, quando nuovi utenti si uniscono alla rete non c'è bisogno di aggiornare le schede preesistenti e i vari centri di controllo non devono né coordinare le loro attività né tenere una lista degli utenti.

Anche se i centri dovessero venire chiusi dopo aver distribuito tutte le schede, la rete continuerebbe a funzionare in maniera decentralizzata per un periodo indefinito.

---

Lo schema è basato su un sistema crittografico a chiave pubblica con una principale modifica: invece di generare una coppia casuale di chiavi pubbliche e private e pubblicare una di queste due, l'utente prende il suo nome e il suo indirizzo nella rete come corrispondente chiave pubblica. Ogni combinazione di informazioni opportune (nome, indirizzo, numero dell'ufficio, numero telefonico...) può essere usata come chiave, purché identifichi in maniera univoca l'utente, in modo che quest'ultimo non possa ripudiarla in seguito, e (purché) possa essere velocemente reperibile alle altre parti.

Quando Alice vuole mandare un messaggio a Bob, firma il messaggio con la chiave privata presente nella sua scheda personale, lo cifra utilizzando la chiave pubblica di Bob ottenuta dalla sua identità Bob<sub>id</sub> (per esempio nome e indirizzo della rete), aggiunge la propria identità al messaggio e lo invia. Quando Bob lo riceve, lo decifra utilizzando la chiave privata presente nella sua scheda e poi verifica la firma utilizzando l'identità del mittente come chiave di verifica.

La necessità di avere un centro esterno KGC si fa evidente nel processo di computare le chiavi private. Se queste fossero calcolate dagli utenti stessi allora, visto che non c'è niente di speciale nell'identità dell'utente, se A fosse in grado di computare la chiave privata relativa alla chiave pubblica "A" allora niente gli impedirebbe di calcolare anche le chiavi private relative alle chiavi pubbliche "B", "C" e così via. Di conseguenza lo schema non sarebbe più sicuro.

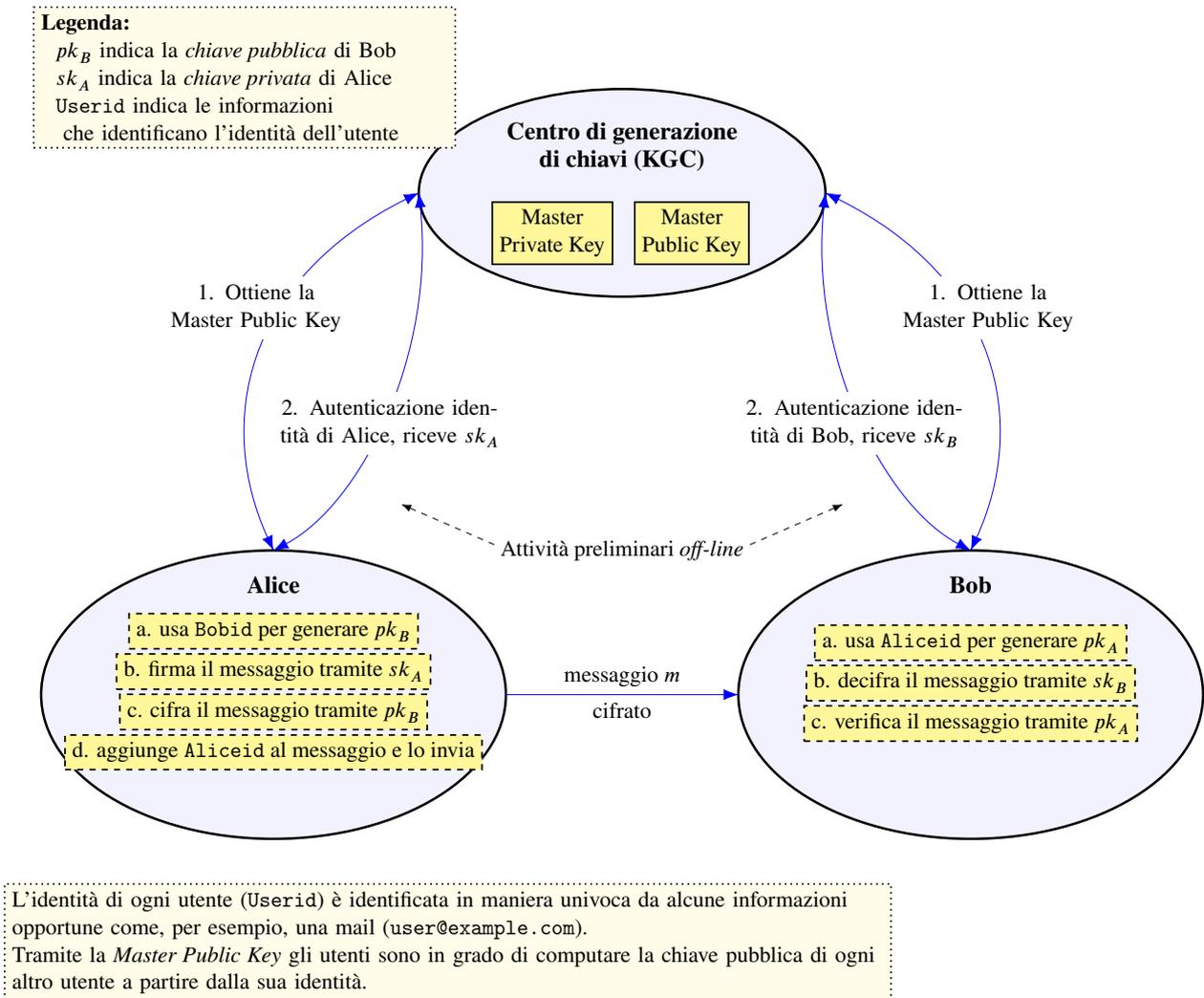
Similmente al ruolo di una Autorità Centrale, nell'interazione con l'utente per la consegna della *smart card* personale il centro KGC svolgerebbe il compito di autenticazione e verifica delle informazioni dell'identità.

Inoltre si potrebbe trovare in una posizione privilegiata data dalla conoscenza di informazioni segrete (come, per esempio, la fattorizzazione del prodotto di due primi grandi) che gli permetterebbero di computare le chiavi private di tutti gli utenti. Di conseguenza si può osservare che in un sistema basato sull'identità è intrinseco il metodo del *key-escrow*, una tecnica con cui la chiave necessaria a decifrare dei dati criptati è conservata da terze parti con un *acconto di garanzia* in modo che, in casi particolari, possa essere recuperata anche senza il consenso dei diretti interessati.

La sicurezza generale dello schema dipende dai seguenti aspetti:

- a) la sicurezza delle funzioni crittografiche utilizzate;
- b) la segretezza delle informazioni conservate al Centro di generazione di chiavi;
- c) la completezza dei controlli di identità effettuati dai centri prima di erogare le *smart card*;

Figura 3.1: Una rappresentazione schematica dei vari passaggi *online* e *offline* di uno schema di cifratura *identity based*.



- d) le precauzioni prese dagli utenti per prevenire la perdita, la duplicazione o un utilizzo non autorizzato della scheda personale.

Ovviamente il centro deve controllare attentamente le richieste per l'immissione di nuove *smart card* per evitare false dichiarazioni e deve provvedere alla sua stessa sicurezza. Gli utenti invece possono proteggere se stessi contro un uso non autorizzato della carta tramite un sistema di password oppure memorizzando parte della chiave.

Lo schema crittografico collega il messaggio con l'informazione necessaria per l'identificazione e mentre il possesso della carta lega effettivamente i con l'utente fisico.

---

### 3.0.1 Firma digitale *Identity-Based*

Quando propose la ricerca di uno schema crittografico basato sull'identità, Shamir espose l'idea per un semplice primo schema di firma digitale basato sull'identità. Lo schema è basato sulla verifica della condizione

$$s^e = i \cdot t^{f(t,m)} \pmod n$$

dove

- $m$  è il messaggio da firmare,
- $s, t$  costituiscono la firma digitale,
- $i$  rappresenta l'identità dell'utente,
- $n$  è il prodotto di due primi grandi,
- $e$  è un primo grande, coprimo con  $\Phi(n)$ ,
- $f$  è una funzione unidirezionale.

I parametri  $n, e$  e la funzione  $f$  sono scelti dal KGC e tutti gli utenti dispongono degli stessi valori di  $n, e$  e degli stessi algoritmi per computare  $f$  all'interno delle *smart card*. Questi valori possono anche essere resi pubblici ma la fattorizzazione di  $n$  deve essere nota solo al *Key Generation Center*. L'unica differenza tra gli utenti sono il valore di  $i$  e la corrispondente chiave segreta, data dall'unico numero  $g$  tale che

$$g^e = i \pmod n.$$

Il valore di  $g$  può essere facilmente calcolabile dal KGC ma, supponendo valida l'ipotesi RSA, nessun avversario è in grado di ottenerla, dato che

$$g = i^d \pmod n \quad \text{con} \quad d = e^{-1} \pmod{\Phi(n)}.$$

Per firmare un messaggio  $m$  l'utente prende un numero casuale  $r$  e calcola

$$t = r^e \pmod n.$$

La condizione da verificare può quindi essere riscritta come

$$s^e = g^e \cdot r^{ef(t,m)} \pmod n.$$

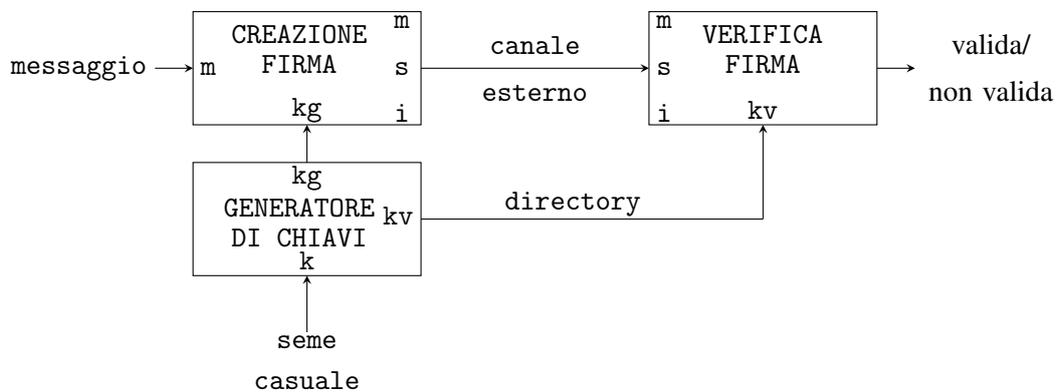
Dato che  $e$  è coprimo con  $\Phi(n)$  si può eliminare il fattore comune agli esponenti per ottenere

$$s = g \cdot r^{f(t,m)} \pmod n$$

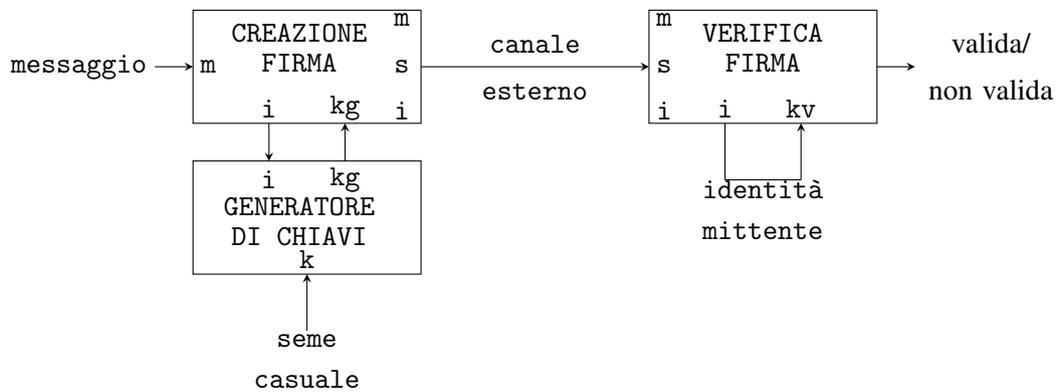
e quindi  $s$  può essere computato senza alcuna estrazione di radice.

I seguenti diagrammi rappresentano schematicamente il processo di firma digitale in un sistema a chiave pubblica e in uno basato sull'identità, dove  $kg$  indica la chiave per la generazione della firma,  $kv$  quella per la verifica,  $s$  la firma,  $m$  il messaggio e  $i$  l'identità dell'utente.

**Schema di firma digitale a chiave pubblica:**



**Schema di firma digitale basato sull'identità:**

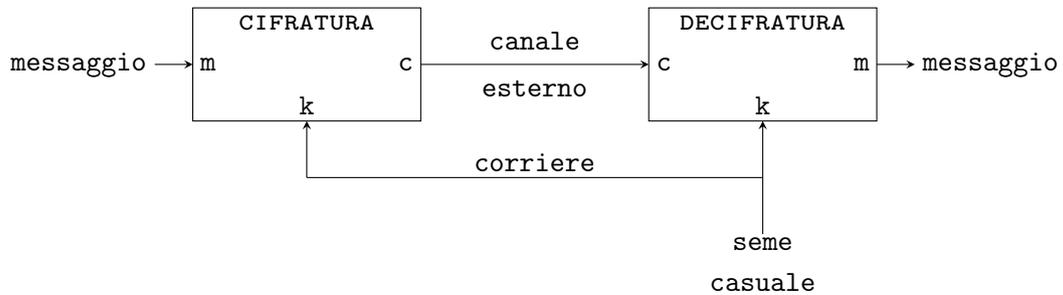


---

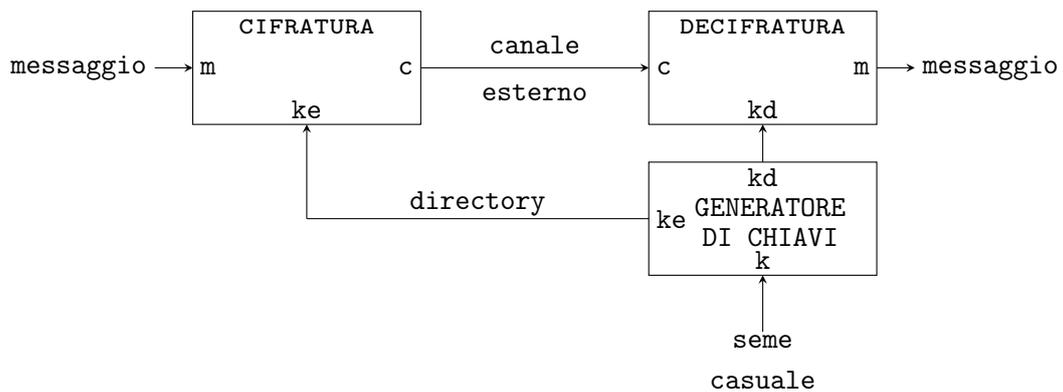
## Chiave privata, chiave pubblica e Identity-Based

La differenze principali tra i tre sistemi crittografici sono riassunte nei seguenti diagrammi.

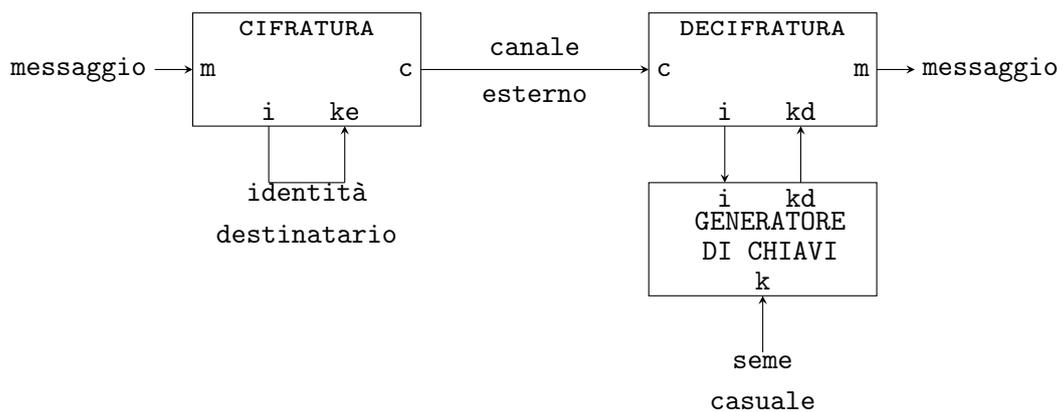
### Schema a chiave privata:



### Schema a chiave pubblica:



### Schema Identity-Based:



In tutti gli schemi il messaggio  $m$  è cifrato tramite la chiave  $ke$ , trasmesso come testo cifrato  $c$  attraverso il canale esposto e decifrato con la chiave  $kd$ . La scelta delle chiavi è basata sul seme casuale  $k$ .

- In uno schema a chiave privata si ha  $k_e = k_d = k$  e il canale esterno "sicuro" (solitamente un corriere) deve preservare sia la segretezza, sia l'autenticità della chiave.
- In uno schema a chiave pubblica le chiavi di cifratura e decifratura sono ottenute da  $k$  tramite funzioni diverse,  $k_e = f_e(k)$  e  $k_d = f_d(k)$ ; il canale esterno deve preservare solo l'autenticità della chiave.
- In uno schema basato sull'identità, la chiave di cifratura è data dall'identità del destinatario,  $k_e = i$ , mentre la chiave di decifratura è ottenuta da  $i$  e da  $k$ ,  $k_d = f(i, k)$ . Il canale esterno tra utenti e autorità centrale è eliminato e rimpiazzato da un'unica interazione con il KGC al momento dell'inserimento nella rete.

### Implementazione

Per ottenere uno schema in cui la chiave pubblica può essere una stringa arbitraria servono quattro algoritmi: *setup* genera i parametri globali del sistema e una chiave globale, *master key*; *extract* usa la chiave globale per generare la chiave privata corrispondente a un'arbitraria chiave pubblica  $ID \in \{0, 1\}^*$ ; *encrypt* cifra i messaggi utilizzando la chiave pubblica  $ID$ ; *decrypt* decodifica i messaggi tramite la chiave privata.

Per implementare l'idea dello schema basato sull'identità bisogna partire da uno schema a chiave pubblica con due proprietà in più: (1) quando il seme  $k$  è noto, si può trovare la chiave segreta per un parte non trascurabile delle chiavi pubbliche; (2) il problema di trovare il seme  $k$  a partire da una specifica coppia di chiavi pubblica e privata generata da  $k$  deve essere intrattabile.

**Osservazione.** Lo schema di cifratura classico RSA non rispetta simultaneamente queste condizioni: (1) se il modulo  $n$  è una funzione pseudorandom dell'identità dell'utente, allora nemmeno il centro KGC è in grado di fattorizzare  $n$  e di ottenere la chiave di decifratura  $d$  dalla chiave pubblica  $e$ . (2) se il modulo  $n$  è universale e il seme è la sua fattorizzazione segreta, allora chiunque conosca le chiavi pubblica e privata,  $e$  e  $d$ , può calcolare il seme.

Al momento di aprire la questione nel 1984 Shamir propose già uno schema per la firma e l'autenticazione digitale ([SA]) ma non uno per un sistema di crittografia basato sull'identità. Nel 1987 Tanaka presentò uno schema basato sul problema del logaritmo discreto e della fattorizzazione ([TA]), con l'inconveniente di essere sicuro finché una parte degli utenti non cospiri, e quindi uno schema legato alla crittografia *threshold*. Sono stati poi proposti anche altri sistemi Identity-Based ma tutti non completamente utilizzabili.

Boneh e Franklin proposero uno schema completamente funzionante, con sicurezza *chosen ciphertext security in the random oracle model* (in un modello oracolare contro un attacco *chosen ciphertext*) basato su un analogo del problema computazionale di Diffie-Hellman (CDH).

Inoltre, a differenza di quanto avviene in un normale schema asimmetrico, all'avversario viene data maggior potenza rispetto a un normale attacco *chosen ciphertext*, poiché gli è permesso ottenere le chiavi private di arbitrarie chiavi pubbliche mentre attacca una specifica chiave pubblica ID. Anche con questo aiuto dovrebbe guadagnare un vantaggio comunque trascurabile nel forzare la sicurezza del sistema.

L'efficienza è comparabile a quella della cifratura di ElGamal in  $\mathbb{F}_p^*$  e può essere costruito a partire da una qualsiasi mappa bilineare  $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$  tra gruppi  $\mathbb{G}_1, \mathbb{G}_2$  purché la variante del CDH sia intrattabile in  $\mathbb{G}_1$ . Come esempio di questa costruzione si utilizza il *Weil Pairing* su curve ellittiche.

### 3.1 Prime definizioni

Uno schema di cifratura basato sull'identità (*identity-based encryption scheme*)  $\Pi$  è descritto tramite quattro algoritmi:

**Setup:** dato un parametro di sicurezza  $k$  genera i parametri del sistema,  $\text{params}$ , e la chiave globale *master-key*. Nei parametri del sistema sono inclusi la descrizione dello spazio dei messaggi in chiaro  $\mathcal{M}$  e dello spazio dei messaggi cifrati  $\mathcal{C}$ , oltre alla descrizione della mappa bilineare e dei vari gruppi in gioco. I parametri del sistema sono pubblici, mentre la chiave globale deve essere nota solo al centro KGC.

**Extract:** estrae la chiave privata da una determinata chiave pubblica. Dati in input  $\text{params}$ , *master-key* e una chiave pubblica arbitraria  $\text{ID} \in \{0, 1\}^*$  genera la corrispondente chiave privata di decifratura  $d$ .

**Encrypt:** dati in input i parametri, la chiave ID e  $M \in \mathcal{M}$  produce un testo cifrato  $C \in \mathcal{C}$ .

**Decrypt:** dai parametri del sistema, dal testo  $C \in \mathcal{C}$  e dalla chiave  $d$  ritorna  $M \in \mathcal{M}$ .

Setup:	( $k$ )	→	params, master-key
Extract:	(params, master-key, ID)	→	$d$
Encrypt:	(params, ID, $M$ )	→	$C \in \mathcal{C}$
Decrypt:	(params, $C$ , $d$ )	→	$M \in \mathcal{M}$

Gli algoritmi devono inoltre soddisfare la condizione di correttezza, ovvero, data la chiave privata  $d$  generata da Extract dalla chiave ID deve valere

$$\forall M \in \mathcal{M} : \text{Decrypt}(\text{params}, C, d) = M \text{ con } C = \text{Encrypt}(\text{params}, \text{ID}, M)$$

Il sistema IBE di Boneh e Franklin utilizza una mappa bilineare tra gruppi  $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ , dove  $\mathbb{G}_1$  e  $\mathbb{G}_2$  sono due gruppi di ordine  $q$  primo. Si dice mappa bilineare *ammissibile* una mappa che soddisfa le proprietà:

1. *Bilineare*:  $\hat{e}$  deve essere tale che  $\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$  per ogni  $P, Q \in \mathbb{G}_1$  e per ogni  $a, b \in \mathbb{Z}$ .
2. *Non degenera*: la mappa non può mandare tutte le coppie di  $\mathbb{G}_1 \times \mathbb{G}_1$  nell'identità di  $\mathbb{G}_2$ .
3. *Computabile*: ovvero esiste un algoritmo efficiente per calcolare  $\hat{e}(P, Q)$  per ogni  $P, Q \in \mathbb{G}_1$ .

In questo sistema  $\mathbb{G}_1$  è il sottogruppo dei punti di torsione di una curva ellittica su un campo  $\mathbb{F}_p$  mentre  $\mathbb{G}_2$  è un sottogruppo del gruppo moltiplicativo sul campo  $\mathbb{F}_{p^2}$ . La mappa bilineare ammissibile è costruita tramite il Weil Pairing.

### 3.1. PRIME DEFINIZIONI

---

## Capitolo 4

# Sicurezza

In questo capitolo vengono presentate le definizioni dei vari livelli di sicurezza in un sistema a chiave pubblica, per poi estenderle a un sistema basato sull'identità. In questo modo si vanno a sottolineare le nozioni (minime) di sicurezza che deve soddisfare il sistema IBE, la cui dimostrazione viene trattata nel Capitolo 6.

### Notazione

Nel presente capitolo utilizzeremo la seguente notazione riguardo alla sintassi di uno schema di cifratura; in un sistema asimmetrico questo è dato da una tripla di algoritmi  $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$  dove

- $\mathcal{K}$ , l'algoritmo *generatore di chiavi*, è un algoritmo probabilistico che prende in input un parametro di sicurezza  $k \in \mathbb{N}$  e restituisce una coppia  $(pk, sk)$  di chiavi, rispettivamente, pubblica e privata.
- $\mathcal{E}$ , l'algoritmo di *cifratura (encryption algorithm)*, è un algoritmo probabilistico che, data una chiave pubblica  $pk$  e un messaggio  $x \in \{0, 1\}^*$  produce un testo cifrato  $y$ .
- $\mathcal{D}$ , l'algoritmo di *decifratura (decryption algorithm)*, è un algoritmo deterministico che prende una chiave segreta  $sk$  e un testo cifrato  $y$  e restituisce un messaggio  $x \in \{0, 1\}^*$  oppure un simbolo speciale  $\perp$  che indica la non validità del testo cifrato.

Richiediamo che valga la *condizione di correttezza*, ovvero che per ogni coppia  $(pk, sk)$  prodotta da  $\mathcal{K}(1^k)$ , per ogni  $x \in \{0, 1\}^*$  e per ogni  $y$  output di  $\mathcal{E}_{pk}(x)$  si verifichi  $\mathcal{D}_{sk}(y) = x$ . Il pedice degli algoritmi  $\mathcal{E}$  e  $\mathcal{D}$  indica su quale chiave è computato l'output. Richiediamo inoltre che gli algoritmi  $\mathcal{K}$ ,  $\mathcal{E}$  e  $\mathcal{D}$  abbiano un tempo di esecuzione polinomiale.

---

Considerato un sistema crittografico a chiave pubblica (ma lo stesso si può dire di quelli a chiave privata), esso può rispettare o meno diverse nozioni teoriche di sicurezza. Nel seguente capitolo andremo ad analizzare e comparare la differente robustezza delle nozioni di sicurezza più comuni (ed effettivamente più utilizzate) per schemi di cifratura a chiave pubblica. Queste poi verranno adattate a sistemi crittografici basati sull'identità tenendo conto delle due principali differenze tra questi sistemi.

La prima differenza tra un sistema a chiave pubblica ed uno basato sull'identità è che nel primo l'avversario è sfidato su una chiave pubblica casuale, mentre nel secondo su una chiave pubblica  $ID$  di sua scelta. La seconda grande differenza consta nel fatto che in un attacco ad un sistema IBE l'avversario può essere in ossesso delle chiavi private di utenti  $ID_1, \dots, ID_n$  di sua scelta. Per questo la definizione di sicurezza deve essere ulteriormente rafforzata e deve permettere all'avversario di ottenere le chiavi private associate ad ogni identità  $ID_i$  di sua scelta, purché diversa dall'identità  $ID$  oggetto della sfida.

Una maniera comoda di organizzare e collegare le varie definizioni è quella di considerare separatamente i possibili *obiettivi* della sicurezza e le possibili *tipologie di attacco*, ottenendo così ogni definizione come una coppia di un particolare obiettivo e un particolare attacco.

I due principali obiettivi di sicurezza in un sistema a chiave pubblica sono la *non-malleabilità* e l'*indistinguibilità*. Nel presente lavoro ci soffermeremo maggiormente sul secondo in quanto è la definizione di sicurezza poi estesa ad un sistema basato sull'identità, mentre al primo faremo, per completezza, semplicemente qualche accenno<sup>1</sup>. L'indistinguibilità (in maniera abbreviata IND) formalizza l'incapacità dell'avversario di ottenere qualsiasi informazione sul testo in chiaro  $x$  relativo al testo cifrato  $y$ , oggetto della sfida. Riguarda quindi una nozione "forte" di privacy ed è spesso usata per l'ideazione di protocolli crittografici. La non-malleabilità, invece, formalizza l'incapacità dell'avversario, dato un testo cifrato  $y$ , di generare un altro testo cifrato  $y'$  tale che i corrispettivi testi in chiaro  $x, x'$  siano collegati in maniera significativa (per esempio  $x' = x + 1$ ), ovvero rappresenta un concetto di sicurezza per il quale i testi cifrati sono a prova di manomissione.

Riguardo all'altra "coordinata" delle definizioni di sicurezza consideriamo tre possibili attacchi. In ordine crescente di robustezza sono *chosen-plaintext attack* (CPA), *non-adaptive chosen-ciphertext attack* (CCA1) e *adaptive chosen-ciphertext attack* (CCA2).

In un attacco CPA l'avversario può ottenere le cifrature di messaggi in chiaro di sua scelta ed è una nozione che rispecchia la reale situazione di un sistema a chiave pubblica, in quanto l'avversario conosce la chiave pubblica e questa è sufficiente per generare questo tipo di attacchi. In un attacco CCA1, in aggiunta alla chiave pubblica, l'avversario ha accesso ad un oracolo

---

<sup>1</sup>Per una trattazione più esaustiva sulla *non-malleabilità* e sulle reciproche implicazioni con l'*indistinguibilità* si rimanda a [BDPR].

per la funzione di decifratura. All'avversario è permesso l'utilizzo dell'oracolo solamente nel periodo precedente al ricevere il testo cifrato  $y$  oggetto della sfida. Il termine *non-adattivo* sta a significare proprio il fatto che le richieste per l'oracolo di decifratura non possono dipendere dal testo  $y$  (in maniera informale questo tipo di attacco è anche chiamato "attacco della pausa pranzo" o "attacco di mezzanotte"). Nell'ultima tipologia di attacco, CCA2, l'avversario ha gli stessi benefici dell'attacco precedente (la chiave pubblica e l'accesso all'oracolo) però, essendo questo un attacco *adattivo*, le richieste oracolari di decifratura possono dipendere dal testo della sfida  $y$ , purché ovviamente siano diverse dal testo  $y$  stesso.

Si possono abbinare in ogni combinazione gli obiettivi {IND, NM} e i diversi attacchi {CPA, CCA1, CCA2}, ottenendo sei diverse nozioni di sicurezza:

IND-CPA, IND-CCA1, IND-CCA2, NM-CPA, NM-CCA1, NM-CCA2.

La sicurezza IND-CPA, nota anche come *sicurezza polinomiale*, considera un attacco insito nella definizione di sistema a chiave pubblica. Si verifica che un algoritmo di cifratura deterministico non può essere polinomialmente sicuro, mentre lo è un qualsiasi sistema di cifratura a chiave pubblica probabilistico (per la dimostrazione si rimanda a [GM]).

Per ogni coppia di nozioni di sicurezza  $\mathbf{A}, \mathbf{B} \in \{\text{IND-CPA}, \text{IND-CCA1}, \text{IND-CCA2}, \text{NM-CPA}, \text{NM-CCA1}, \text{NM-CCA2}\}$  si può dimostrare una delle seguenti relazioni:

- $\mathbf{A} \implies \mathbf{B}$ : se  $\Pi$  è uno schema di cifratura che soddisfa la nozione di sicurezza  $\mathbf{A}$ , allora  $\Pi$  soddisfa anche la nozione  $\mathbf{B}$ .
- $\mathbf{A} \not\Rightarrow \mathbf{B}$ : si può costruire uno schema crittografico  $\Pi$  che soddisfa la nozione di sicurezza  $\mathbf{A}$  ma che non soddisfa la nozione  $\mathbf{B}$  (sotto l'ipotesi che esista qualche schema che soddisfa la nozione  $\mathbf{A}$ ). Un risultato di questo tipo è detto anche *separazione*.

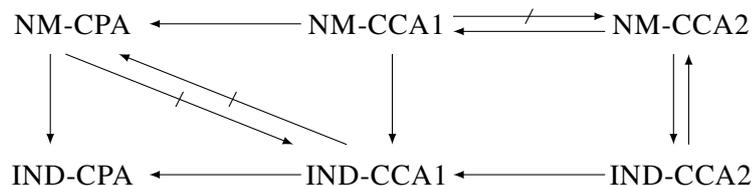


Figura 4.1: Diagramma delle implicazioni e separazioni tra le diverse nozioni di sicurezza.

Nella Figura 4.1 è rappresentato un diagramma dove sono messe in relazione le diverse definizioni di sicurezza. Una freccia semplice orientata che collega due vertici rappresenta un'implicazione da una nozione di sicurezza ad un'altra, mentre una freccia barrata rappresenta una separazione tra le due. Considerando il diagramma come un grafo e prese due nozioni  $\mathbf{A}$  e  $\mathbf{B}$ ,  $\mathbf{A}$  implica  $\mathbf{B}$  se e solo se esiste un cammino da  $\mathbf{A}$  a  $\mathbf{B}$  nel grafo.

---

Si può facilmente osservare che la sicurezza IND-CCA2 implica tutte le altre nozioni di sicurezza e questo supporta l'idea che la sicurezza contro un attacco *chosen-ciphertext* adattivo sia la nozione corretta su cui concentrarsi.

Le sei nozioni di sicurezza appena descritte possono essere poi inserite in un sistema oracolare randomizzato (*random-oracle model*, RO), fornendo sei definizioni corrispondenti. Si può dimostrare che le implicazioni e le separazioni mostrate nel diagramma in Figura 4.1 restano valide anche in un sistema oracolare.

## Sicurezza Semantica

Un'altra nozione per un sistema di cifratura a chiave pubblica è quella di *Sicurezza semantica*. In maniera informale, un sistema è *semanticamente sicuro* se tutto ciò che un intercettatore è in grado di calcolare riguardo al testo in chiaro dato il testo cifrato, lo può calcolare anche senza il testo cifrato. Si dimostra che ogni sistema a chiave pubblica *polinomialmente sicuro* gode anche di *sicurezza semantica*<sup>2</sup>. Quindi, schemi di cifratura probabilistici a chiave pubblica rispettano una versione della *segretezza perfetta* di Shannon inserita in un contesto di algoritmi con limiti polinomiali: considerando avversari con risorse con limiti polinomiali, su un insieme di messaggi la distribuzione di probabilità *a posteriori* (dopo aver intercettato il messaggio cifrato) è la stessa di quella *a priori*.

Sia  $f$  una qualsiasi funzione definita dallo spazio dei messaggi  $M$  in un qualche insieme  $V$  e diciamo che  $f(m)$  rappresenta una qualche informazione riguardo al messaggio  $m \in M$  ( $f$  può essere la funzione identità, un predicato Booleano, una funzione hash...).

Vogliamo che estrarre una qualsiasi informazione sui messaggi dalla loro cifratura sia *difficile* anche quando sia nota la distribuzione di probabilità sullo spazio dei messaggi.

Per ogni  $m \in M$ , sia  $p_m = \Pr(x = m | x \in M)$ . Consideriamo l'insieme immagine  $f(M)$  e definiamo  $p^M = \max_{v \in V} \left( \sum_{m \in f^{-1}(v)} p_m \right)$  e  $v^M$  un valore in  $f(M)$  che raggiunge il massimo delle probabilità. Sia  $E$  un algoritmo di cifratura e sia noto all'avversario. Consideriamo le seguenti tre sfide:

**Sfida1:** Si prende casualmente un messaggio  $m \in M$  (ogni  $x \in M$  ha probabilità  $p_x$  di essere preso). In questa prova all'avversario viene chiesto di indovinare il valore di  $f(m)$  senza sapere quale sia  $m$ .

Se l'avversario rispondesse sempre  $v^M$  vincerebbe con probabilità  $p^M$ . Non c'è alcuna strategia che possa dare all'avversario una maggiore probabilità di vittoria.

---

<sup>2</sup>Per una trattazione più formale della *sicurezza semantica*, che non è obiettivo di questo elaborato, e della sua relazione con la *sicurezza polinomiale* si rimanda a [GM].

Sfida2: Si prende casualmente  $m \in M$ . Si calcola una cifratura  $\alpha \in E(m)$  e la si invia all'avversario. L'avversario deve indovinare  $f(m)$ .

Sfida3: Si permetta all'avversario di prendere una funzione  $f_E$  definita su  $M$ . Si prende casualmente  $m \in M$ . Si calcola una cifratura  $\alpha \in E(m)$  e la si invia all'avversario. L'avversario deve indovinare  $f_E(m)$ .

In maniera informale, si dice che  $\Pi$  è un sistema di cifratura a chiave pubblica *semanticamente sicuro* se l'avversario non può vincere la Sfida3 con maggiore probabilità rispetto alla Sfida1.

## 4.1 Definizioni di sicurezza

In questa sessione diamo le definizioni formali di sicurezza in uno schema asimmetrico. Nella loro formalizzazione consideriamo un avversario  $A$  come una coppia di algoritmi probabilistici,  $A = (A_1, A_2)$ , dove  $A$  ha tempo polinomiale di esecuzione se  $A_1$  e  $A_2$  sono algoritmi polinomiali. Questa impostazione corrisponde ad "eseguire"  $A$  in due passaggi, dove nel primo l'avversario, data la chiave pubblica, genera delle "istanze di prova" e nel secondo passaggio è sfidato su un testo cifrato  $y$ , generato in funzione dell'obiettivo della specifica sfida. Inoltre  $A_1$  può generare delle informazioni  $s$  da passare a  $A_2$ .

In un attacco *chosen plaintext* l'avversario può cifrare testi in chiaro di sua scelta. Questo tipo di attacco è inevitabile in un sistema a chiave pubblica, poiché, conoscendo la chiave pubblica, l'avversario può produrre testi cifrati di messaggi in chiaro di sua scelta in maniera autonoma.

In un attacco *chosen ciphertext* si concede all'algoritmo  $A_1$  l'accesso alla chiave pubblica e a un oracolo di decifratura, mentre a  $A_2$  solo la chiave pubblica. In sostanza l'oracolo può essere utilizzato per generare le "istanze di prova" ma viene poi revocato prima che inizi la sfida.

In un attacco *chosen ciphertext* adattivo (CCA2) si dà accesso all'oracolo di decifratura anche all'algoritmo  $A_2$ , con l'unica restrizione che non lo si può interrogare sul testo  $y$  oggetto della sfida. Quest'ultimo attacco rappresenta un modello di un attacco estremamente forte.

La motivazione della nozione di sicurezza contro un attacco adattivo *chosen ciphertext* sta nel fatto che la nozione standard di sicurezza semantica considera la sicurezza contro un avversario "passivo", ovvero che si limita ad intercettare messaggi. Non considera invece un attacco alla sicurezza mosso da un avversario "attivo", che è capace di influenzare il comportamento degli utenti della rete. Quindi la sicurezza IND-CCA fornisce un livello di sicurezza contro ogni tipologia di avversario.

Definiamo ora la nozione di indistinguibilità tramite una sfida. L'algoritmo  $A_1$  è processato con input la chiave pubblica  $pk$  e genera come output una tripla  $(x_0, x_1, s)$ : le prime due componenti sono messaggi della stessa lunghezza mentre l'ultima rappresenta le informazioni sullo

schema (includendo  $pk$ ) che si vogliono preservare. Preso un bit casuale  $b \in \{0, 1\}$ , si prende  $x_b$  e si determina il testo cifrato  $y$  oggetto della sfida cifrando  $x_b$  tramite la chiave pubblica  $pk$ . A questo punto è compito dell'algoritmo  $A_2$  determinare il valore del bit  $b$ , dati  $y$  e le informazioni  $s$ .

Per concisione definiamo simultaneamente l'indistinguibilità per CPA, CCA1 e CCA2. L'unica differenza sta nell'accessibilità o meno all'oracolo di decifratura per gli algoritmi  $A_1$  e  $A_2$ . La stringa "atk" sta ad indicare una delle scritture "cpa, cca1, cca2", mentre ATK indica il simbolo formale per "CPA, CCA1, CCA2". La scrittura  $\mathcal{O}_i = \varepsilon$ , con  $i \in \{1, 2\}$ , rappresenta la funzione  $\mathcal{O}_i$  che genera la stringa vuota  $\varepsilon$  con ogni input.

**Definizione.** Sia  $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$  uno schema di cifratura e sia  $A = (A_1, A_2)$  un avversario. Per  $\text{atk} \in \{\text{cpa}, \text{cca1}, \text{cca2}\}$  e  $k \in \mathbb{N}$  definiamo

$$\text{Adv}_{A, \Pi}^{\text{ind-atk}}(k) \stackrel{\text{def}}{=} 2 \cdot \Pr \left[ (pk, sk) \leftarrow \mathcal{K}(1^k); (x_0, x_1, s) \leftarrow A_1^{\mathcal{O}_1}(pk); b \leftarrow \{0, 1\}; \right. \\ \left. y \leftarrow \mathcal{E}_{pk}(x_b) : A_2^{\mathcal{O}_2}(x_0, x_1, s, y) = b \right] - 1$$

dove

$$\begin{aligned} \text{Se } \text{atk} = \text{cpa} \quad & \text{allora } \mathcal{O}_1(\cdot) = \varepsilon \quad \text{e } \mathcal{O}_2(\cdot) = \varepsilon \\ \text{Se } \text{atk} = \text{cca1} \quad & \text{allora } \mathcal{O}_1(\cdot) = \mathcal{D}_{sk}(\cdot) \quad \text{e } \mathcal{O}_2(\cdot) = \varepsilon \\ \text{Se } \text{atk} = \text{cca2} \quad & \text{allora } \mathcal{O}_1(\cdot) = \mathcal{D}_{sk}(\cdot) \quad \text{e } \mathcal{O}_2(\cdot) = \mathcal{D}_{sk}(\cdot) \end{aligned}$$

Ribadiamo che  $A_1$  genera i messaggi  $x_0, x_1$  con  $|x_0| = |x_1|$  e nello schema CCA2  $A_2$  non può chiedere all'oracolo la decifratura di  $y$ .

Lo schema  $\Pi$  è **sicuro** nel senso di IND-ATK se, preso un avversario  $A$  con tempo di esecuzione polinomiale, la funzione  $\text{Adv}_{A, \Pi}^{\text{ind-atk}}(\cdot)$  è trascurabile.

**Osservazione.** Riguardo un avversario  $A = (A_1, A_2)$  si può assumere, dove conveniente, che i messaggi  $x_0$  e  $x_1$  generati da  $A_1$  siano distinti. Intuitivamente questo non diminuisce il vantaggio dell'avversario perché il caso in cui i messaggi siano uguali va a dare un contributo nullo al vantaggio (se  $x_0 = x_1$  per indovinare da quale dei due provenga  $x_b$ ,  $A$  non può far altro che tirare ad indovinare).

Si può modificare l'avversario  $A$  in modo che i due messaggi siano distinti, stando attenti ad assicurarsi che nel caso in cui  $A_1$  generi due messaggi identici l'avversario modificato non guadagni qualche vantaggio, così che il vantaggio resti lo stesso dell'avversario iniziale.

**Proposizione 4.1.** Sia  $A = (A_1, A_2)$  un avversario che effettua un attacco IND-ATK contro lo schema di cifratura  $\Pi$ . Allora esiste un altro avversario  $B = (B_1, B_2)$  che effettua un attacco IND-ATK contro  $\Pi$  tale che i due messaggi (di uguale lunghezza) generati da  $B_1$  siano sempre

distinti,  $\text{Adv}_{B,\Pi}^{\text{ind-atk}}(k) = \text{Adv}_{A,\Pi}^{\text{ind-atk}}(k)$  e tale che il tempo di esecuzione di  $B$  sia uguale a quello di  $A$  a meno di un fattore moltiplicativo costante.

**Dimostrazione:** I due avversari  $A$  e  $B$  hanno accesso ad un oracolo  $\mathcal{O}_1$  nella prima fase e ad un oracolo  $\mathcal{O}_2$  nella seconda, istanziati a seconda della tipologia di attacco ATK. L'avversario  $B = (B_1, B_2)$  è definito come segue:

---

Algoritmo  $B_1^{\mathcal{O}_1}(pk)$   
 $(x_0, x_1, s) \leftarrow A_1^{\mathcal{O}_1}(pk);$   
 se  $x_0 \neq x_1$  allora  $d \leftarrow 0$  altrimenti  $d \leftarrow 1;$   
 $x'_0 \leftarrow x_0; \quad s' \leftarrow s \| d;$   
 se  $d = 0$  allora  $x'_1 \leftarrow x_1$ , altrimenti  $x'_1 \leftarrow \overline{x_0};$   
 restituisci  $(x'_0, x'_1, s')$

---

Algoritmo  $B_2^{\mathcal{O}_2}(x'_0, x'_1, s', y)$  dove  $s' = s \| d$   
 se  $d = 0$  allora  $c \leftarrow A_2^{\mathcal{O}_2}(x'_0, x'_1, s, y)$   
 altrimenti  $c \leftarrow \{0, 1\}$   
 restituisci  $c$

Definendo in questo modo  $x'_0$  e  $x'_1$  vale sempre  $x'_0 \neq x'_1$ . Inoltre, quando  $x'_0 = x'_1$   $B_2$  dà in output un bit casuale  $c$  in maniera che il suo vantaggio in questo caso sia nullo. Si verifica facilmente che il tempo di esecuzione di  $B$  è uguale a quello di  $A$  a meno di una costante moltiplicativa.

Mostriamo ora che  $\text{Adv}_{B,\Pi}^{\text{ind-atk}}(k) = \text{Adv}_{A,\Pi}^{\text{ind-atk}}(k)$ . Per farlo consideriamo le *prove* su cui si basano le definizioni dei vantaggi di  $A$  e di  $B$ , rispettivamente:

Prova1  $\stackrel{\text{def}}{=} (pk, sk) \leftarrow \mathcal{K}(1^k); (x_0, x_1, s) \leftarrow A_1(pk); b \leftarrow \{0, 1\};$   
 $y \leftarrow \mathcal{E}_{pk}(x_b); c \leftarrow A_2^{\mathcal{O}_2}(x_0, x_1, s, y)$   
 Prova2  $\stackrel{\text{def}}{=} (pk, sk) \leftarrow \mathcal{K}(1^k); (x_0, x_1, s) \leftarrow A_1(pk); b \leftarrow \{0, 1\};$   
 $y \leftarrow \mathcal{E}_{pk}(x_b); c \leftarrow B_2^{\mathcal{O}_2}(x'_0, x'_1, s \| d, y)$

Nella seconda prova  $x'_0, x'_1, d$  sono definiti in funzione di  $x_0, x_1$  come nella definizione dell'algoritmo  $B_1$ . Si denoti con  $\text{Pr}_1[\cdot] = \text{Pr}[\text{Prova1} : \cdot]$  la funzione probabilità nella Prova1, ed analogamente per la Prova2. Per definizione si ha

$$\text{Adv}_{A,\Pi}^{\text{ind-atk}}(k) = 2 \cdot \text{Pr}_1[b = c] - 1 \quad \text{e} \quad \text{Adv}_{B,\Pi}^{\text{ind-atk}}(k) = 2 \cdot \text{Pr}_2[b = c] - 1.$$

Quindi è sufficiente mostrare che  $\text{Pr}_1[b = c] = \text{Pr}_2[b = c]$ . Denotiamo con  $E$  l'evento in cui

$x_0 = x_1$  o, equivalentemente,  $d = 1$ . Allora

$$\begin{aligned}\Pr_1[b = c] &= \Pr_1[b = c|E] \cdot \Pr_1[E] + \Pr_1[b = c|\overline{E}] \cdot \Pr_1[\overline{E}] \\ \Pr_2[b = c] &= \Pr_2[b = c|E] \cdot \Pr_2[E] + \Pr_2[b = c|\overline{E}] \cdot \Pr_2[\overline{E}]\end{aligned}$$

Facciamo le seguenti osservazioni:

- $E$  dipende solo da  $A_1$ , da cui  $\Pr_1[E] = \Pr_2[E]$ .
- $\Pr_1[b = c|E] = 1/2$  poiché, quando  $E$  è vero,  $A_2$  non ha nessuna informazione riguardo a  $b$ . D'altro canto  $\Pr_2[b = c|E] = 1/2$  poiché quando  $E$  è vero  $B_2$  genera come output un bit casuale.
- $\Pr_1[b = c|\overline{E}] = \Pr_2[b = c|\overline{E}]$  perché in questo caso le due *prove* coincidono e quindi stiamo considerando in entrambi i casi l'output di  $A_2$ .

Da queste segue immediatamente  $\Pr_1[b = c] = \Pr_2[b = c]$  e quindi la tesi.  $\square$

## 4.2 Sicurezza in un sistema basato sull'identità

Consideriamo ora il caso di un sistema IBE.

La sicurezza in un sistema basato sull'identità si ottiene adattando al nuovo sistema le definizioni viste per un sistema a chiave pubblica, ricordando le due principali differenze, ovvero che la chiave pubblica su cui viene sfidato l'attaccante è di sua scelta e che quest'ultimo può essere in possesso delle chiavi private di utenti di sua scelta (eccetto, ovviamente, quella su cui viene sfidato).

Il sistema IBE finale dovrà rispettare la sicurezza contro un attacco adattivo *chosen ciphertext* (IND-ID-CCA), ovvero l'estensione ad un sistema basato sull'identità della nozione più forte di sicurezza finora trattata. Per una trattazione più lineare e più semplice da seguire, per ottenere questo schema si parte da uno schema di cifratura più semplice e meno sicuro, il quale poi viene gradualmente modificato per ottenere maggior sicurezza.

### 4.2.1 IND-ID-CPA

La dimostrazione della sicurezza del sistema di Boneh e Franklin utilizza inizialmente la nozione di sicurezza polinomiale in un sistema basato sull'identità, IND-ID-CPA. È un rafforzamento della definizione base con la possibilità da parte dell'avversario di avanzare richieste di estrazione di chiave privata e, essendo in un sistema IBE, di scegliere la chiave pubblica ID su cui essere sfidato.

Uno schema IBE  $\Pi$  è *polinomialmente sicuro*, IND-ID-CPA, se nessun avversario polinomiale ha un vantaggio non trascurabile contro lo sfidante nella sfida IND-ID-CPA:

**Setup:** lo sfidante processa  $\text{Setup}(k)$ . Dà i parametri del sistema all'avversario e tiene per sé la *master-key*.

**Fase 1:** l'avversario inoltra le richieste di estrazione di chiave privata  $ID_1, \dots, ID_n$ : lo sfidante ottiene la chiave privata  $d_i$  e la invia ad  $\mathcal{A}$ . Le richieste possono essere fatte in maniera adattiva, ovvero ogni richiesta  $ID_i$  può dipendere dalle risposte a  $ID_1, \dots, ID_n$ .

**Sfida:** terminata la Fase 1, l'avversario genera  $M_0, M_1 \in \mathcal{M}$  di uguale lunghezza e sceglie un'identità  $ID$  sulla quale essere sfidato. Lo sfidante calcola  $C = \text{Encrypt}(\text{params}, ID, M_b)$ , con  $b \in \{0, 1\}$  casuale, e lo invia ad  $\mathcal{A}$ .

**Fase 2:** l'avversario inoltra ulteriori richieste  $ID_1, \dots, ID_n$  come nella Fase 1, con  $ID_i \neq ID$ .

**Risposta:** l'avversario risponde con  $b' \in \{0, 1\}$  e vince se  $b = b'$ .

Definiamo il vantaggio di  $\mathcal{A}$  nell'attaccare lo schema  $\Pi$  tramite la funzione del parametro di sicurezza  $k$ :

$$\text{Adv}_{\Pi, \mathcal{A}}(k) = |\Pr[b = b'] - \frac{1}{2}|.$$

La probabilità è calcolata sui bit casuali usati dallo sfidante e dall'avversario.

**Definizione.** Un sistema IBE  $\Pi$  è semanticamente sicuro se per ogni avversario IND-ID-CPA  $\mathcal{A}$  polinomiale la funzione  $\text{Adv}_{\Pi, \mathcal{A}}(k)$  è trascurabile, ovvero  $\forall \epsilon > 0 \exists k_0$  tale che  $|\text{Adv}_{\Pi, \mathcal{A}}(k)| < \frac{1}{k^\epsilon}$  per  $k > k_0$ .

## 4.2.2 IND-ID-CCA

Il sistema finale sviluppato da Boneh e Franklin è caratterizzato da indistinguibilità per un attacco *chosen ciphertext* in un sistema *Identity-Based* (IND-ID-CCA). È analogo ad un attacco IND-ID-CPA ma con un avversario più forte che può effettuare anche richieste di decifratura durante l'attacco.

**Definizione.** Definiamo la sfida IND-ID-CCA tra un avversario  $\mathcal{A}$  e lo sfidante:

**Setup:** lo sfidante prende un parametro di sicurezza  $k$  e processa  $\text{Setup}$ . Dà i parametri del sistema  $\text{params}$  all'avversario e tiene per sé la chiave globale *master-key*.

**Fase 1:** l'avversario inoltra le richieste  $q_1, \dots, q_n$  dove  $q_i$  è una tra:

- Domanda di estrazione  $\langle \text{ID}_i \rangle$ : lo sfidante usa `Extract` per ottenere la corrispondente chiave privata  $d_i$  e la invia a  $\mathcal{A}$ .
- Domanda di decifratura  $\langle \text{ID}_i, C_i \rangle$ : tramite gli algoritmi `Extract` e `Decrypt` genera la chiave privata  $d_i$  corrispondente a  $\text{ID}_i$  e la usa per decifrare  $C_i$ .  $\mathcal{A}$  riceve il corrispondente testo in chiaro.

Le richieste possono essere avanzate in maniera adattiva.

**Sfida:** quando l'avversario  $\mathcal{A}$  termina la Fase 1, produce due testi in chiaro  $M_0, M_1 \in \mathcal{M}$  di uguale lunghezza e sceglie un'identità  $\text{ID}$  sulla quale essere sfidato. Lo sfidante prende un bit casuale  $b \in \{0, 1\}$ , calcola  $C = \text{Encrypt}(\text{params}, \text{ID}, M_b)$  e lo invia ad  $\mathcal{A}$ .

**Fase 2:** l'avversario inoltra altre richieste  $q_n, \dots, q_{n+m}$  come nella Fase 1. Ovviamente si applicano le restrizioni  $\text{ID}_i \neq \text{ID}$  per una richiesta di estrazione  $\langle \text{ID}_i \rangle$ , e  $\langle \text{ID}_i, C_i \rangle \neq \langle \text{ID}, C \rangle$  per una domanda di decifratura  $\langle \text{ID}_i, C_i \rangle$ . Anche in questo caso le richieste possono essere adattive.

**Risposta:** l'avversario risponde con  $b' \in \{0, 1\}$  e vince se  $b = b'$ .

Definiamo il vantaggio di  $\mathcal{A}$  nell'attaccare lo schema  $\Pi$  come nel caso della sfida IND-ID-CPA:

$$\text{Adv}_{\Pi, \mathcal{A}}(\mathbf{k}) = |\Pr[b = b'] - \frac{1}{2}|.$$

La probabilità è calcolata sui bit casuali usati dallo sfidante e dall'avversario.

**Definizione.** Un sistema IBE è sicuro contro un attacco adattivo *chosen ciphertext* se per ogni avversario polinomiale IND-ID-CCA la funzione  $\text{Adv}_{\Pi, \mathcal{A}}(\mathbf{k})$  è trascurabile.

## Capitolo 5

# Bilinear Diffie Hellman Problem

Il fine di questo capitolo è descrivere l'*ipotesi bilineare di Diffie-Hellman*, una modifica dell'*ipotesi Computazionale* su cui si basa il sistema IBE di Boneh e Franklin. Inizialmente viene descritto il collegamento tra il logaritmo discreto sulle curve ellittiche e il logaritmo discreto su un'estensione del campo di base e viene approfondita la *separazione* tra l'ipotesi computazionale di Diffie-Hellman e l'ipotesi decisionale, ovvero si evidenzia che quest'ultima risulta risolubile in gruppi particolari (come le curve ellittiche supersingolari utilizzate nel sistema IBE). Da queste considerazioni sorge la necessità di introdurre questa nuova ipotesi per il sistema crittografico.

Il problema del logaritmo discreto (DL, *discrete logarithm*) rappresenta, assieme al problema della fattorizzazione, uno dei principali problemi su cui si basa la crittografia a chiave pubblica, da cui l'importanza di gruppi che siano efficacemente computabili e sui quali il problema del logaritmo sia ritenuto *intrattabile*.

**Logaritmo discreto:** Sia  $G$  un gruppo (qui in notazione moltiplicativa) di ordine  $p$  primo e sia  $g$  un suo generatore. Dato  $a \in G$ , il *problema del logaritmo discreto* consiste nel trovare l'unico intero  $x \in [1, p - 1]$  tale che  $g^x = a \pmod p$ , ovvero  $x = \log_g a \pmod p$ .

Il sistema (ovvero il gruppo  $G$  con le caratteristiche e gli elementi appena descritti) soddisfa l'*ipotesi del logaritmo discreto* se nessun avversario polinomiale ha un vantaggio non trascurabile nella risoluzione del problema.

Tuttavia, dimostrare l'intrattabilità del logaritmo discreto in un gruppo qualsiasi è una questione difficile e tuttora aperta. Di conseguenza, per verificare la possibilità di utilizzare uno specifico gruppo per costruire uno schema crittografico solitamente si studiano gli algoritmi noti per risolvere il problema del logaritmo discreto in quello specifico gruppo. Per farlo si considerano due classi di algoritmi: *algoritmi generici*, che funzionano su qualsiasi gruppo e non sfruttano la sua particolare rappresentazione, e *algoritmi non generici*, la cui funzione è limitata al singolo gruppo considerato (o alla singola tipologia di gruppo).

---

Spesso però per provare la sicurezza di un sistema non è sufficiente considerare il problema del logaritmo discreto, ma ci si basa su altre due questioni collegate: il problema computazionale di Diffie-Hellman (CDH) e il problema decisionale di Diffie-Hellman (DDH). È noto che questi due problemi non sono più *difficili* del problema del logaritmo discreto.

Nel protocollo di Diffie-Hellman per lo scambio di chiavi Alice e Bob considerano un gruppo ciclico  $G$  ed un suo generatore  $g$ , prendono rispettivamente (e separatamente) due interi casuali  $a, b \in [1, |G|]$  e si scambiano  $g^a$  e  $g^b$ . La chiave segreta è data dall'elemento  $g^{ab}$ , facilmente computabile da entrambi. Per forzare lo schema un intercettatore "passivo" (ovvero che si limita ad intercettare lo scambio di messaggi) deve calcolare la funzione di Diffie-Hellman, definita come:  $\text{DH}_g(g^a, g^b) = g^{ab}$ .

**Diffie-Hellman computazionale:** Il gruppo  $G$  soddisfa l'*ipotesi di Diffie-Hellman computazionale* se nessun algoritmo *efficiente* (ovvero con tempo di esecuzione polinomiale) è in grado di calcolare  $\text{DH}_g(g^a, g^b)$ .

**Diffie-Hellman decisionale:** L'*ipotesi collegata*, l'*ipotesi decisionale di Diffie-Hellman*, assume che nessun algoritmo *efficiente* è in grado di distinguere le due distribuzioni  $\langle g^a, g^b, g^{ab} \rangle$  e  $\langle g^a, g^b, g^c \rangle$ , dove  $a, b, c$  sono presi casualmente in  $[1, |G|]$ .

L'*ipotesi DDH* considera assunzioni molto forti, più forti di quelle dell'*ipotesi CDH*. Tuttavia esistono gruppi in cui l'*ipotesi CDH* è supposta vera, mentre è risolubile il problema decisionale. Per esempio, consideriamo il gruppo  $\mathbb{Z}_p^*$  con  $p$  primo e  $g$  generatore. Il problema computazionale di Diffie-Hellman è supposto intrattabile in questo gruppo, però dati  $g^a, g^b$  è possibile utilizzare il simbolo di Legendre per distinguere  $\langle g^a, g^b, g^{ab} \rangle$  da  $\langle g^a, g^b, g^c \rangle$ . Per questo motivo la gran parte dei gruppi nei quali si ritiene che valga l'*ipotesi decisionale* ha come ordine un numero primo. Un esempio di gruppo in cui vale è quello del sottogruppo dei residui quadratici  $\mathcal{Q}_p$  di  $\mathbb{Z}_p^*$ , con  $p$  numero primo sicuro ( $p = 2p_1 + 1$  con  $p_1$  numero primo).

Basandosi sull'utilizzo delle curve ellittiche e delle applicazioni *pairings* si riescono ad ottenere gruppi dove DDH è risolubile mentre i problemi CDH e DL sono equivalenti e presumibilmente *intrattabili*. In questo modo, sotto l'*ipotesi della non risolubilità del logaritmo discreto*, questa costruzione *separa* il problema computazionale di Diffie-Hellman da quello decisionale<sup>1</sup>.

---

<sup>1</sup>Per una trattazione più completa sui tre problemi e sulle relazioni tra essi in gruppi di varia natura si rimanda a [DB], [MA], [JN], [MW] e lavori collegati. Riguardo alla *separazione* tra CDH e DDH, la costruzione in [MW] utilizza i numeri *B-smooth* e si basa sull'*ipotesi Smoothness Assumption*, mentre il lavoro di [JN] sfrutta proprietà della densità dei numeri primi nelle serie aritmetiche.

## 5.1 Applicazione alle Curve Ellittiche

Consideriamo una curva ellittica definita sul campo finito  $\mathbb{F}_{p^s}$  con  $p$  primo. In generale si conoscono solo algoritmi generici per risolvere il problema del logaritmo discreto sulle curve ellittiche. Tuttavia, in specifici casi, esistono algoritmi efficienti che spostano il problema DL dalla curva al gruppo moltiplicativo di un'estensione  $\mathbb{F}_{p^{rs}}$  del campo su cui è definita. Questi algoritmi sfruttano le applicazioni bilineari *pairings*<sup>2</sup>.

Queste funzioni mappano coppie di punti di torsione di ordine  $l$  ( $P, Q$ ) nelle radici  $l$ -esime dell'unità,  $\langle P, Q \rangle$ , assumendo valori nell'estensione  $\mathbb{F}_{p^{rs}}$ . La proprietà di bilinearità rappresenta semplicemente la proprietà

$$\langle aP, bQ \rangle = \langle P, Q \rangle^{ab}.$$

I *pairings* possono essere definiti per tutte le curve ellittiche, tuttavia possono essere computati efficientemente solo quando  $r$  è sufficientemente piccolo. Inoltre possono essere definiti su un'estensione di grado  $r$ , dove  $r$  è il più piccolo intero tale che  $l$  divide  $p^{rs} - 1$ . Il caso più semplice, spesso utilizzato nelle implementazioni che sfruttano i *pairings*, riguarda le curve supersingolari, con  $r = 2$  (che è proprio il caso utilizzato da Boneh e Franklin nella realizzazione del sistema IBE).

### Equivalenza tra DL e CDH<sup>3</sup>

Riguardo all'equivalenza tra il problema del logaritmo discreto e il problema computazionale di Diffie-Hellman, il principale risultato (di [MA]) ne dà le condizioni nel caso in cui sia nota la fattorizzazione dell'ordine del gruppo. Per un gruppo di ordine primo  $q$  il risultato può essere espresso dal seguente teorema:

**Teorema 5.1.** *Sia  $G$  un gruppo ciclico di ordine  $q$  primo. Sia  $E$  una curva ellittica definita su  $\mathbb{F}_q$  il cui ordine  $\mathcal{O}$  sia "B-smooth", per un qualche  $B$ . Allora il logaritmo discreto in  $G$  può essere calcolato utilizzando  $O(\log^2 q)$  chiamate ad un oracolo-DH e  $O((B/\log B) \log^2 q)$  operazioni del gruppo.*

**Definizione.** Un numero è detto *B-smooth* quando tutti i suoi fattori primi non sono maggiori di  $B$ .

Tuttavia, i risultati del teorema dipendono dall'esistenza di interi vicini a  $q$  che siano sufficientemente *smooth*. In più, anche se un tale numero  $\mathcal{O}$  esistesse, costruire una curva su  $\mathbb{F}_q$  con

<sup>2</sup>La costruzione del *Weil Pairing* fu introdotta in [MOV] per trasportare il problema DL da una curva supersingolare su un campo finito ad un'estensione del campo di grado basso. L'algoritmo per computare efficientemente il *Weil Pairing* fu originariamente proposto da Miller in [MV].

<sup>3</sup>Nel dettaglio vedere [MA], [MW].

cardinalità  $\mathcal{O}$  è un problema ritenuto difficile e quindi il teorema non porterebbe ad un algoritmo efficiente per provare l'equivalenza tra i due problemi DL e CDH.

Però, se si riesce a costruire un gruppo assieme alla curva ausiliaria (del Teorema) è possibile utilizzare il precedente risultato per provare un'equivalenza in tempo polinomiale tra DL e CDH, supponendo che il limite  $B$  sia polinomiale in  $\log q$ .

### Risolubilità di DDH

Consideriamo un punto di torsione  $P$  di ordine primo  $l$  su una curva ellittica e  $G$  il gruppo di ordine  $l$  generato da esso. Per risolvere il problema decisionale di Diffie-Hellman su questo tipo di gruppi ci si può avvalere delle mappe *pairings*. Sia  $(P, aP, bP, cP)$  una tupla del problema DDH. Si ha

$$\begin{aligned}\langle aP, bP \rangle &= \langle P, P \rangle^{ab}, \\ \langle P, cP \rangle &= \langle P, P \rangle^c.\end{aligned}$$

In generale,  $\langle P, P \rangle$  è 1 oppure è una radice primitiva  $l$ -esima dell'unità. Nel secondo caso, quando  $\langle P, P \rangle \neq 1$ , DDH è risolubile. Nel caso del *Weil Pairing*, però, si ha che per definizione  $\langle P, P \rangle$  è sempre uguale a 1 e quindi la mappa non risulta utile nel risolvere il problema.

Nel caso delle curve supersingolari definite su  $\mathbb{F}_p$ , per le proprietà delle mappe *pairing* (sia del *Weil pairing* che del *Tate pairing*) si ha che ogni punto di torsione  $P$  di ordine  $l$  a coordinate in  $\mathbb{F}_p$  è tale che  $\langle P, P \rangle = 1$ . È possibile però evitare questa conseguenza modificando il *pairing*. Per farlo si utilizza un endomorfismo, detto *distorsione*, per costruire il *pairing* modificato.

Questo endomorfismo,  $\Phi$ , mappa punti definiti sul campo di definizione in punti definiti su un'estensione del campo e in questo modo si ottiene che vale la proprietà  $\langle P, \Phi(P) \rangle \neq 1$ . Di conseguenza si può utilizzare il *pairing* modificato  $\langle \cdot, \Phi(\cdot) \rangle$  per risolvere il problema decisionale di Diffie-Hellman nel gruppo generato da  $P$ .

Consideriamo  $p$  primo tale che  $p \equiv 2 \pmod{3}$  e la curva su  $\mathbb{F}_p$  definita da  $y^2 = x^3 + 1$ . Allora, in questo caso, come endomorfismo di *distorsione* che soddisfa le condizioni richieste si può prendere

$$\begin{aligned}\Phi : E(\mathbb{F}_p) &\longrightarrow E(\mathbb{F}_{p^2}) \\ (x, y) &\longmapsto (\zeta x, y)\end{aligned}$$

dove  $\zeta \in \mathbb{F}_{p^2}$  rappresenta una radice cubica dell'unità (ovviamente diversa da 1), soluzione di  $\zeta^3 - 1 = 0$ .

### Separazione tra DDH e CDH

Utilizzando le osservazioni delle due sottosezioni precedenti si possono costruire gruppi dove il problema DDH è risolubile mentre i problemi DL e CDH sono equivalenti, in maniera che il

problema del logaritmo discreto sia comunque ritenuto *intrattabile*. Qui ne diamo solamente una breve esposizione.

Si sfruttano le curve supersingolari e quindi è possibile utilizzare la tecnica usata in [MOV] per trasportare il logaritmo discreto dalla curva a  $\mathbb{F}_{p^2}$ , che richiede tempo polinomiale. In questo modo il miglior algoritmo per il calcolo del logaritmo discreto sulle curve ellittiche ha un tempo subesponenziale in  $p^2$ . Considerando questo aspetto, si può scegliere di cercare un valore di  $q$  relativamente piccolo oppure un valore vicino a  $p$ .

Per prima cosa si sceglie il parametro  $B$ , poi si prendono numeri primi casuali minori di  $B$  e li si moltiplica finché il loro prodotto  $\omega_0$  diventa più grande di  $2^B$  e quindi rientra nell'intervallo  $[2^B, B \cdot 2^B]$ . Sia  $\omega = 3\omega_0$ ; si cerca il più piccolo primo  $q$  nella successione  $k\omega - 1$ . Trovato  $q$ , si cerca il più piccolo primo  $p$  nella successione  $4lq - 1$ . Supponiamo per il momento che  $k$  ed  $l$  siano minori di  $B$ .

Dato che  $q \equiv 2 \pmod{3}$ , la curva ellittica  $y^2 = x^3 + 1$  su  $\mathbb{F}_q$  è supersingolare ed ha ordine  $q + 1 = k\omega$ . Si verifica che  $q + 1$  è  $B$ -smooth, con  $B$  nell'ordine di  $\log(q)$ . In questo modo è possibile ottenere un "trasferimento" in tempo polinomiale tra i problemi DL e CDH in  $\mathbb{G}_q$ .

Osserviamo che questa costruzione è euristica, dato che non è stato provato che  $k$  ed  $l$  sono sufficientemente piccoli. Tuttavia, considerando la distribuzione dei numeri primi nelle sequenze aritmetiche e il Teorema di Dirichlet, si ha che, se  $a$  e  $b$  sono coprimi allora ci sono infiniti numeri primi nella successione  $a + b \cdot k$ . Inoltre, questa successione contiene asintoticamente  $n/(\phi(b) \log(n))$  primi. Di conseguenza, ci si aspetta che  $k$  ed  $l$  siano polinomiali rispetto al modulo dei numeri generati. In questa maniera si ottiene un algoritmo efficiente.

## 5.2 Ipotesi bilineare di Diffie-Hellman

Come osservato nel presente capitolo, grazie alla conoscenza della mappa bilineare  $\hat{e}$  si può ricondurre il problema del logaritmo discreto in  $\mathbb{G}_1$  ad un logaritmo discreto in  $\mathbb{G}_2$ . Di conseguenza i parametri di sicurezza devono essere scelti in maniera che il logaritmo discreto sia intrattabile in entrambi i gruppi. Inoltre abbiamo anche visto che il problema Decisionale di Diffie-Hellman è risolvibile in  $\mathbb{G}_1$ .

Per queste considerazioni il sistema crittografico IBE non può basarsi sull'ipotesi DDH ma si baserà su una variante del problema Computazionale di Diffie-Hellman.

**Problema Bilineare di Diffie-Hellman:** Siano  $\mathbb{G}_1, \mathbb{G}_2$  due gruppi di ordine primo  $q$ , sia  $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$  una mappa bilineare ammissibile e sia  $P$  un generatore di  $\mathbb{G}_1$ . Il problema bilineare di Diffie-Hellman (BDH, *Bilinear Diffie-Hellman*) in  $\langle \mathbb{G}_1, G_2, \hat{e} \rangle$  consiste nel calcolare  $W = \hat{e}(P, P)^{abc} \in \mathbb{G}_2$  dati  $\langle P, aP, bP, cP \rangle$  con  $a, b, c \in \mathbb{Z}_q$ .

Un algoritmo  $\mathcal{A}$  ha vantaggio  $\epsilon$  nel risolvere BDH in  $\langle G_1, G_2, \hat{e} \rangle$  se

$$\Pr [\mathcal{A}(P, aP, bP, cP) = \hat{e}(P, P)^{abc}] \geq \epsilon$$

dove la probabilità è calcolata sulle scelte casuali di  $a, b, c \in \mathbb{Z}_q$ , di  $P \in \mathbb{G}_1^*$  e sui bit casuali di  $\mathcal{A}$ .

I parametri del sistema BDH sono costruiti da un *generatore di parametri*  $\mathcal{G}$ , un algoritmo randomizzato che prende in input il parametro di sicurezza  $k \in \mathbb{Z}^+$  e in tempo polinomiale genera un primo grande  $q$ , la cui dimensione dipende da  $k$ , la descrizione di due gruppi  $\mathbb{G}_1, \mathbb{G}_2$  di ordine  $q$  e di una mappa ammissibile  $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ :

$$\mathcal{G}(1^k) = \langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e} \rangle.$$

Inoltre supponiamo che per  $i = 1, 2$  la descrizione di  $\mathbb{G}_i$  contiene anche algoritmi polinomiali per effettuare le operazioni in  $\mathbb{G}_i$  e un generatore di  $\mathbb{G}_i$  dal quale poter calcolare elementi casuali in  $\mathbb{G}_i$  in maniera uniforme.

**Ipotesi BDH:** Sia  $\mathcal{G}$  un generatore di parametri. Un algoritmo  $\mathcal{A}$  ha vantaggio  $\epsilon(k)$  nel risolvere il problema BDH per  $\mathcal{G}$  se per  $k$  sufficientemente grandi vale

$$\text{Adv}_{\mathcal{G}, \mathcal{A}}(k) = \Pr \left[ \mathcal{A}(q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, P, aP, bP, cP) = \hat{e}(P, P)^{abc} \mid \begin{array}{l} \langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e} \rangle \leftarrow \mathcal{G}(1^k) \\ P \leftarrow \mathbb{G}_1^*, a, b, c \leftarrow \mathbb{Z}_q^* \end{array} \right] \geq \epsilon(k)$$

BDH è ritenuto intrattabile nei gruppi generati da  $\mathcal{G}$  se  $\mathcal{G}$  soddisfa l'ipotesi BDH, ovvero se per ogni algoritmo randomizzato polinomiale  $\mathcal{A}$  si ha che  $\text{Adv}_{\mathcal{G}, \mathcal{A}}(k)$  è una funzione trascurabile.

**Osservazione.** Un algoritmo che riesca a risolvere il problema computazionale in  $\mathbb{G}_1$  o in  $\mathbb{G}_2$  è sufficiente per risolvere il problema bilineare di Diffie-Hellman in  $\langle \mathbb{G}_1, \mathbb{G}_2, \hat{e} \rangle$ , quindi l'ipotesi BDH non è più difficile di CDH.

## Capitolo 6

# Sistema IBE di Boneh e Franklin

In questo ultimo capitolo viene infine descritto nel dettaglio il sistema di Boneh e Franklin. Ne viene dimostrata la sicurezza IND-ID-CCA e ne vengono delineati i parametri e le mappe specifici per l'attuazione.

Riguardo alle definizioni preliminari degli algoritmi che compongono uno schema IBE si rimanda al Capitolo 3.

Per ottenere lo schema di cifratura basato sull'identità di Boneh e Franklin costruiremo preliminarmente uno schema più debole, `BasicIdent`, con sicurezza IND-ID-CPA dal quale poi otterremo uno schema completo `FullIdent` con sicurezza contro un attacco adattivo *chosen ciphertext* in un sistema oracolare.

Per la dimostrazione della sicurezza sarà necessario utilizzare anche un sistema a chiave pubblica con sicurezza IND-CPA, `BasicPub`.

### 6.1 `BasicIdent` e sicurezza IND-ID-CPA

Nella costruzione del sistema si utilizzano due funzioni hash  $H_1$  e  $H_2$ , nell'analisi di sicurezza saranno viste come oracoli random.

Sia  $\mathcal{G}$  un generatore di parametri che soddisfi l'ipotesi BDH.

Definiamo i quattro algoritmi che descrivono lo schema

`BasicIdent`:

**Setup:** dato  $k \in \mathbb{Z}^+$  parametro di sicurezza

$\mathcal{G}$  genera  $\langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e} \rangle \leftarrow k$  con  $\hat{e}$  mappa ammissibile.

Prende  $P \in \mathbb{G}_1$  generatore casuale e  $s \in \mathbb{Z}_q^*$  casuale; calcola  $P_{pub} = sP$ .

Prende due funzioni hash  $H_1 : \{0, 1\}^* \rightarrow \mathbb{G}_1^*$  e  $H_2 : \mathbb{G}_2 \rightarrow \{0, 1\}^n$ .

Lo spazio dei messaggi e dei testi cifrati sono  $\mathcal{M} \in \{0, 1\}^n$ ,  $C = \mathbb{G}_1^* \times \{0, 1\}^n$ .

$$\begin{array}{l} \text{master-key} = s \in \mathbb{Z}_q^* \\ \text{params} = \langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, n, P, P_{pub}, H_1, H_2 \rangle \end{array} \left| \leftarrow \text{setup}(k) \right.$$

**Extract:** data una stringa  $ID \in \{0, 1\}^*$  calcola  $Q_{ID} = H_1(ID) \in \mathbb{G}_1^*$  e la chiave privata  $d_{ID} = sQ_{ID}$ .

$$d_{ID} \leftarrow \text{extract}(\text{params}, \text{master-key}, ID)$$

**Encrypt:** dato il messaggio  $M \in \mathcal{M}$  e la chiave pubblica ID

$$\text{calcola } Q_{ID} = H_1(ID) \in \mathbb{G}_1^* \text{ e } g_{ID} = \hat{e}(Q_{ID}, P_{pub}) \in \mathbb{G}_2^*$$

prende  $r \in \mathbb{Z}_q^*$  casuale e genera il testo cifrato

$$C = \langle rP, M \oplus H_2(g_{ID}^r) \rangle \in \mathbb{G}_1^* \times \{0, 1\}^n$$

$$C \leftarrow \text{encrypt}(\text{params}, ID, M)$$

**Decrypt:** sia  $C = \langle U, V \rangle \in C$ ; per decrittare con la chiave privata  $d_{ID} \in \mathbb{G}_1^*$  calcola

$$V \oplus H_2(\hat{e}(d_{ID}, U)) = M$$

$$M \leftarrow \text{decrypt}(\text{params}, C, d)$$

**Consistenza:** Lo schema appena descritto è consistente. Per provarlo osserviamo che

$$\begin{aligned} \hat{e}(d_{ID}, U) &= \hat{e}(sQ_{ID}, rP) = \hat{e}(Q_{ID}, P)^{sr} = \hat{e}(Q_{ID}, sP)^r = \hat{e}(Q_{ID}, P_{pub})^r \\ &= g_{ID}^r \end{aligned}$$

Inoltre durante il processo di cifratura e decifrazione i messaggi  $M$  e  $V$  sono sommati modulo 2 bit a bit con la funzione hash  $H_2$  in  $\hat{e}(d_{ID}, U)$  e in  $g_{ID}^r$ .

$$\begin{array}{l} \text{encrypt } V = M \oplus H_2(g_{ID}^r) \\ \text{decrypt } V \oplus H_2(\hat{e}(d_{ID}, U)) = M \end{array}$$

**Sicurezza:** Lo schema BasicIdent è uno schema di cifratura basato sull'identità ed è semanticamente sicuro (IND-ID-CPA) se vale l'intrattabilità dell'ipotesi Bilineare di Diffie-Hellman.

Per dimostrare la sicurezza ci serviremo di uno schema di cifratura a chiave pubblica, BasicPub, composto da 3 algoritmi, e mostreremo che un attacco IND-ID-CPA a BasicIdent può essere

convertito in un attacco IND-CPA a BasicPub e quindi le richieste di estrazione di chiave privata non sono di aiuto all'avversario.

Poi proveremo che, se vale l'ipotesi BDH allora BasicPub è sicuro contro un attacco IND-CPA. Di conseguenza otterremo la sicurezza IND-ID-CPA di BasicIdent.

BasicPub:

**keygen:** dato  $k$  parametro di sicurezza, processa il generatore di parametri ed ottiene

$\langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e} \rangle \leftarrow \mathcal{G}(k)$  con  $\hat{e}$  mappa ammissibile e  $\mathbb{G}_1, \mathbb{G}_2$  gruppi di ordine  $q$ .

Prende  $s \in \mathbb{Z}_q^*$  casuale e pone  $P_{pub} = sP$ .

Prende  $Q_{ID} \in \mathbb{G}_1^*$  casuale.

Sceglie una funzione hash  $H_2 : \mathbb{G}_2 \rightarrow \{0, 1\}^n$ .

Pone

$$\begin{aligned} \text{public-key: } & \langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, n, P, P_{pub}, Q_{ID}, H_2 \rangle \\ \text{private-key: } & d_{ID} = sQ_{ID} \in \mathbb{G}_1^* \end{aligned}$$

**encrypt:** calcola  $g = \hat{e}(Q_{ID}, P_{pub}) \in \mathbb{G}_2^*$  e prende  $r \in \mathbb{Z}_q^*$  casuale; la cifratura di  $M \in \{0, 1\}^n$  è

$$C = \langle rP, M \oplus H_2(g^r) \rangle$$

**decrypt:** decifra  $C = \langle U, V \rangle$  con la chiave privata  $d_{ID} \in \mathbb{G}_1^*$  calcolando

$$V \oplus H_2(\hat{e}(d_{ID}, U)) = M$$

La costruzione di BasicPub è pressoché uguale a quella di BasicIdent con due sostanziali differenze:

- in BasicIdent  $Q_{ID} = H_1(\text{ID})$  mentre nel sistema a chiave pubblica  $Q_{ID}$  è preso in maniera casuale;
- la chiave pubblica coincide con i parametri del sistema Identity-Based con l'unica differenza che la funzione *hash*  $H_1$  è stata sostituita con  $Q_{ID}$ .

Con il seguente teorema andiamo a dimostrare la sicurezza del sistema IBE. Supponiamo che le funzioni *hash*  $H_1$  e  $H_2$  siano algoritmi oracolari e indichiamo con  $t_{\mathcal{A}}$  il tempo di esecuzione dell'algoritmo  $\mathcal{A}$ .

**Teorema 6.1.** *Supponendo che l'ipotesi BDH valga nei gruppi generati da  $\mathcal{G}$ , BasicIdent è uno schema di cifratura basato sull'identità semanticamente sicuro (IND-ID-CPA).*

*Oververo, supponendo che esista  $\mathcal{A}$  avversario IND-ID-CPA con vantaggio  $\epsilon(k)$  contro lo schema*

BasicIdent, e supponendo che  $\mathcal{A}$  effettui al massimo  $q_E > 0$  richieste di estrazione di chiave privata e  $q_{H_2}$  domande alla funzione  $H_2$ , allora esiste un avversario  $\mathcal{B}$  che risolve BDH nei gruppi generati da  $\mathcal{G}$  in tempo  $t_B = O(t_A)$  e con vantaggio almeno

$$\text{Adv}_{\mathcal{G}, \mathcal{B}}(k) \geq \frac{2 \varepsilon(k)}{e(1 + q_E) q_{H_2}}$$

Per la dimostrazione ci avvaliamo del seguente lemma, che mette in relazione la sicurezza IND-ID-CPA di BasicIdent con quella IND-CPA del sistema analogo a chiave pubblica. Consideriamo  $H_1 : \{0, 1\}^* \rightarrow G_1^*$  come un oracolo randomizzato.

**Lemma 6.2.** *Sia  $\mathcal{A}$  un avversario IND-ID-CPA con vantaggio  $\varepsilon(k)$  contro BasicIdent e supponiamo che esegua al massimo  $q_E$  richieste di estrazione di chiave privata. Allora esiste  $\mathcal{B}$  avversario IND-CPA, con tempo di esecuzione  $t_B = O(t_A)$ , con vantaggio contro BasicPub*

$$\text{Adv}_{\mathcal{G}, \mathcal{B}}(k) \geq \frac{\varepsilon(k)}{e(1 + q_E)}$$

**Dimostrazione:** Nell'attacco a IND-CPA lo sfidante processa l'algoritmo **keygen** di BasicPub e ottiene la chiave privata,  $d_{\text{ID}} = sQ_{\text{ID}}$ , che tiene per sé, e la chiave pubblica  $K_{\text{pub}} = \langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, n, P, P_{\text{pub}}, Q_{\text{ID}}, H_2 \rangle$ , che invia a  $\mathcal{B}$ . Quest'ultimo poi deve inviare due messaggi  $M_0, M_1$  allo sfidante e provare a indovinare il bit casuale  $b \in \{0, 1\}$  della cifratura di  $M_b$ . Per guadagnare un vantaggio non trascurabile sfrutta il vantaggio di  $\mathcal{A}$  contro la schema BasicIdent, simulando con lui una sfida IND-ID-CPA in cui deve simulare il comportamento dello sfidante. Per farlo  $\mathcal{B}$  deve creare dei parametri del sistema

$\langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, n, P, P_{\text{pub}}, H_1, H_2 \rangle$  da inviare ad  $\mathcal{A}$ ; coincidono con la chiave pubblica di BasicPub dove però  $Q_{\text{ID}}$  è sostituito dalla funzione  $H_1$ . Per poter rispondere alle eventuali domande che l'avversario può fare all'oracolo  $H_1$ ,  $\mathcal{B}$  ne controlla le risposte nel modo seguente:

**$H_1$ -queries:**  $\mathcal{B}$  tiene una lista  $H_1^{\text{list}}$  inizialmente vuota, i cui elementi sono tuple  $\langle \text{ID}_j, Q_j, b_j, \text{coin}_j \rangle$ . Quando  $\mathcal{A}$  effettua la domanda relativa a  $\langle \text{ID}_i \rangle$  opera così:

1. se  $\langle \text{ID}_i \rangle$  appare già in un elemento di  $H_1^{\text{list}}$  allora  $\mathcal{B}$  risponde  $H_1(\text{ID}_i) = Q_i \in \mathbb{G}_1^*$ ;
2. altrimenti,  $\mathcal{B}$  genera una moneta casuale  $\text{coin}_i \in \{0, 1\}$  tale che  $\Pr[\text{coin}_i = 0] = \delta$ , per un  $\delta$  che verrà descritto in seguito.
3. prende  $b_i \in \mathbb{Z}_q^*$  casuale, 

se $\text{coin}_i = 0$ , calcola $Q_i = b_i P$ se $\text{coin}_i = 1$ , calcola $Q_i = b_i Q_{\text{ID}}$
--

4.  $\mathcal{B}$  aggiunge  $\langle \text{ID}_i, Q_i, b_i, \text{coin}_i \rangle$  a  $H_1^{\text{list}}$  e risponde con  $H_1(\text{ID}_i) = Q_i$ .

La sfida IND-ID-CPA tra  $\mathcal{B}$  e l'avversario  $\mathcal{A}$  si svolge coi seguenti passaggi.

**Setup:**  $\mathcal{B}$  invia ad  $\mathcal{A}$  i parametri del sistema  $\langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, n, P, P_{\text{pub}}, H_1, H_2 \rangle$

**Fase 1:**  $\mathcal{A}$  effettua le richieste di estrazione di chiave privata  $\langle \text{ID}_i \rangle$ .  $\mathcal{B}$  esegue l'algoritmo oracolare  $H_1$  per ottenere  $H_1(\text{ID}_i) = Q_i$ , con  $\langle \text{ID}_i, Q_i, b_i, \text{coin}_i \rangle$  la corrispondente tupla in  $H_1^{\text{list}}$ .

1. Se  $\text{coin}_i = 1$ ,  $\mathcal{B}$  riporta un fallimento e termina l'attacco a BasicPub.
2. Se  $\text{coin}_i = 0$ , definisce la chiave privata  $d_i = b_i P_{\text{pub}}$  e la invia ad  $\mathcal{A}$ .

**Sfida:** terminata la **Fase 1**,  $\mathcal{A}$  genera due messaggi  $M_0, M_1$ , sceglie una chiave pubblica  $\text{ID}_{ch}$  su cui essere sfidato e li invia a  $\mathcal{B}$ .

$\mathcal{B}$  invia  $M_0, M_1$  al suo sfidante (BasicPub) che risponde con  $C = \langle U, V \rangle$ , testo cifrato di  $M_c$  con  $c$  casuale in  $\{0, 1\}$ .

Processa l'oracolo  $H_1$  per ottenere  $H_1(\text{ID}_{ch}) = Q \in \mathbb{G}_1^*$ ; sia  $\langle \text{ID}_{ch}, Q, b, \text{coin} \rangle$  la corrispondente tupla in  $H_1^{\text{list}}$ .

1. Se  $\text{coin} = 0$ ,  $\mathcal{B}$  riporta un fallimento e termina l'attacco a BasicPub.
2. Se  $\text{coin} = 1$ , allora  $Q = bQ_{\text{ID}}$ . Sia  $b^{-1}$  l'inverso di  $b$  mod  $q$ ;  $\mathcal{B}$  pone  $C' = \langle b^{-1}U, V \rangle$  e lo invia ad  $\mathcal{A}$  come risposta alla sfida.  $C'$  risulta esattamente la cifratura di  $M_c$  in BasicIdent con la chiave pubblica  $\text{ID}_{ch}$ .

**Fase 2:** come nella **Fase 1**  $\mathcal{A}$  avanza richieste di estrazione di chiave privata  $\langle \text{ID}_i \rangle$ , purché  $\text{ID}_i \neq \text{ID}_{ch}$ , e  $\mathcal{B}$  risponde nello stesso modo.

**Risposta:**  $\mathcal{A}$  risponde alla sfida con  $c'$  e  $\mathcal{B}$  risponde alla sfida con BasicPub con  $c'$ .

Osserviamo che nel secondo caso della **Fase 1**  $d_i$  è effettivamente la chiave privata associata a  $\text{ID}_i$ ; infatti da  $\text{coin}_i = 0$  si ha  $Q_i = b_i P$  e (ricordando che  $P_{\text{pub}} = sP$ ) vale

$$\begin{aligned} d_i &= b_i P_{\text{pub}} = b_i sP = s b_i P \\ &= s Q_i \end{aligned}$$

Nella decifratura di  $C' = \langle b^{-1}U, V \rangle$  tramite  $d_{ch}$  in BasicIdent bisogna effettuare il calcolo di  $\hat{e}(d_{ch}, b^{-1}U)$ , mentre nella decifratura di  $C$  in BasicPub con chiave  $d_{\text{ID}}$  il calcolo di  $\hat{e}(d_{\text{ID}}, U)$ .

Valendo in questo caso  $Q = bQ_{\text{ID}}$ , si ottiene

$$\begin{aligned}\hat{e}(d_{ch}, b^{-1}U) &= \hat{e}(sQ, b^{-1}U) = \hat{e}(sb^{-1}Q, U) = \hat{e}(sQ_{\text{ID}}, U) \\ &= \hat{e}(d_{\text{ID}}, U)\end{aligned}$$

e quindi le due decifrate coincidono.

Se l'algoritmo  $\mathcal{B}$  non si abortisce allora le risposte alle richieste a  $H_1$  si comportano come in un attacco reale, in quanto ogni risposta è distribuita in maniera uniforme in  $\mathbb{G}_1^*$ . Il testo cifrato  $C'$  dato ad  $\mathcal{A}$  è effettivamente la cifratura di BasicIdent di  $M_c$  con  $c \in \{0, 1\}$  casuale e quindi, per le ipotesi sull'algoritmo  $\mathcal{A}$ , si ha

$$\left| \Pr[c = c'] - \frac{1}{2} \right| \geq \epsilon$$

dove la probabilità è fatta sui bit casuali usati da  $\mathcal{A}$ ,  $\mathcal{B}$  e dallo sfidante.

Per concludere la dimostrazione del Lemma bisogna calcolare la probabilità che l'algoritmo  $\mathcal{B}$  non si interrompa. Supponiamo che  $\mathcal{A}$  effettui  $q_E$  richieste ad  $H_1$ , allora la probabilità che  $\mathcal{B}$  non si abortisca durante la **Fase 1** o la **Fase 2** è  $\delta^{q_E}$  e che non lo faccia durante il passaggio della **Sfida** è  $(1 - \delta)$ . Quindi la probabilità che non lo faccia lungo tutta la sfida nella sua totalità risulta essere  $\delta^{q_E}(1 - \delta)$ , che viene massimizzato per

$$\delta_{opt} = 1 - \frac{1}{q_E + 1}$$

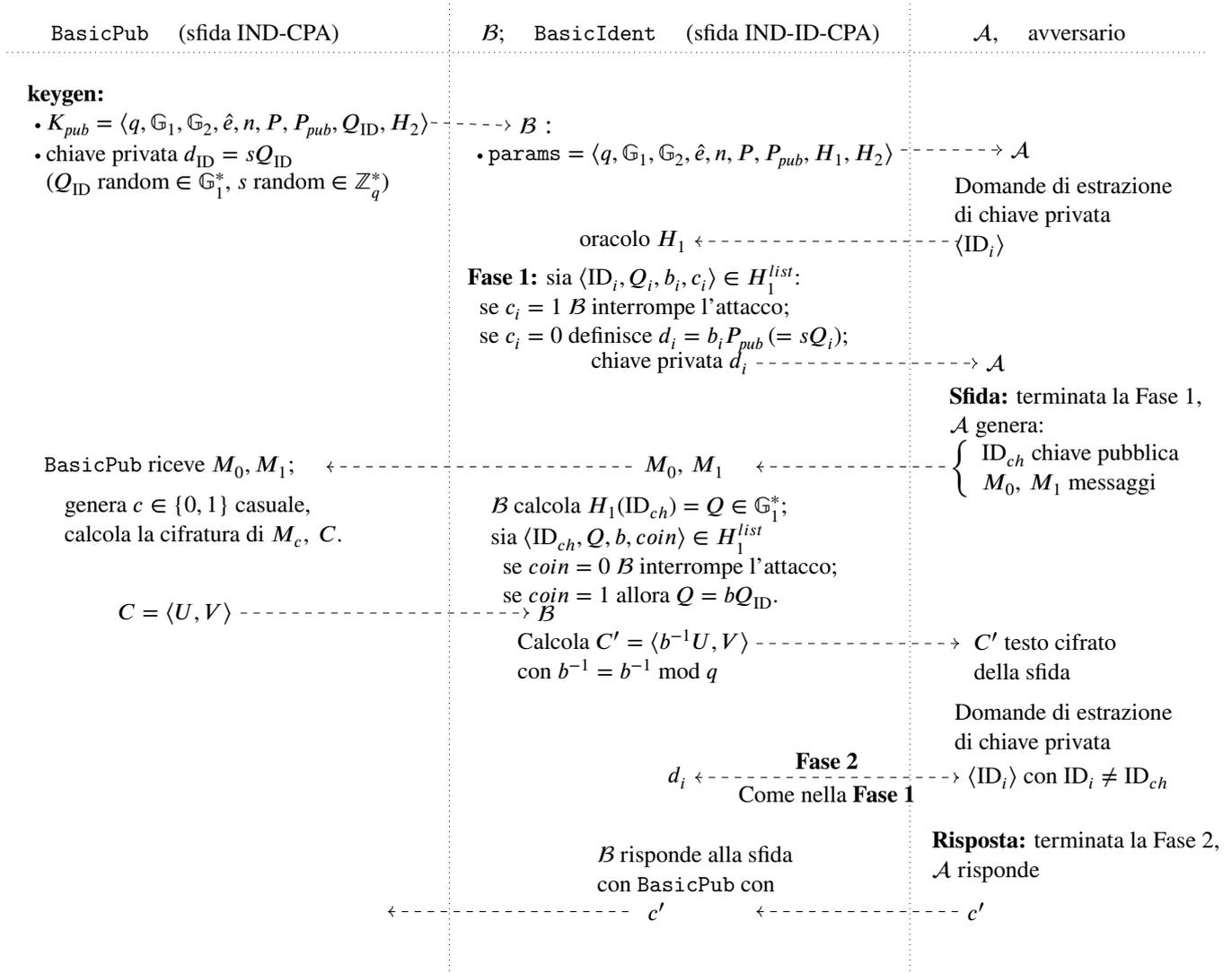
con il quale la probabilità che  $\mathcal{B}$  non si abortisca durante il processo diventa

$$\begin{aligned}\delta_{opt}^{q_E}(1 - \delta_{opt}) &= \left( \frac{q_E}{q_E + 1} \right)^{q_E} \left( \frac{1}{q_E + 1} \right) \\ &\geq \frac{1}{e(q_E + 1)}.\end{aligned}$$

Di conseguenza il vantaggio di  $\mathcal{B}$  risulta almeno  $\frac{\epsilon}{e(q_E + 1)}$ . □

Figura 6.1: Schema che rappresenta le fasi delle sfide del Lemma 6.2.

Una freccia  $\dashrightarrow$  indica un trasferimento di dati.



Per concludere la dimostrazione del Teorema 6.1 utilizziamo anche il seguente lemma, che prova la sicurezza IND-CPA di BasicPub nell'ipotesi dell'intrattabilità di BDH.

**Lemma 6.3.** *Sia  $H_2$  un oracolo randomizzato da  $\mathbb{G}_2$  a  $\{0, 1\}^n$ . Sia  $\mathcal{A}$  un avversario IND-CPA con vantaggio  $\epsilon(k)$  contro BasicPub e supponiamo che effettui  $q_{H_2}$  domande a  $H_2$ . Allora esiste un algoritmo  $\mathcal{B}$  con tempo  $t_{\mathcal{B}} = O(t_{\mathcal{A}})$  che risolve il problema BDH per  $\mathcal{G}$  con vantaggio almeno  $2\epsilon(k)/q_{H_2}$ .*

**Dimostrazione:**  $\mathcal{B}$  riceve i parametri BDH  $\langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e} \rangle$  prodotti da  $\mathcal{G}$  e i dati BDH  $\langle P, P_1, P_2, P_3 \rangle = \langle P, aP, bP, cP \rangle$ , dove  $P$  è un punto casuale di  $\mathbb{G}_1^*$  e  $a, b, c$  sono casuali in  $\mathbb{Z}_q^*$  con  $q$  ordine dei gruppi.

Per trovare la soluzione  $D = \hat{e}(P, P)^{abc} \in \mathbb{G}_2$ ,  $\mathcal{B}$  interagisce con  $\mathcal{A}$  simulando lo sfidante in una sfida IND-CPA contro BasicPub. Per poterlo fare deve poter rispondere alle domande che  $\mathcal{A}$  può rivolgere all'oracolo  $H_2$  in ogni momento. Ne simula il comportamento in questo modo:

**$H_2$ -queries:**  $\mathcal{B}$  tiene una lista di tuple  $H_2^{list}$  inizialmente vuota, i cui elementi sono  $\langle X_j, H_j \rangle$ . Quando  $\mathcal{A}$  inoltra la richiesta  $\langle X_i \rangle$ :

1. se  $\langle X_i \rangle$  appare già nell'elemento  $\langle X_i, H_i \rangle$  della lista allora  $\mathcal{B}$  risponde  $H_2(X_i) = H_i$ ;
2. altrimenti,  $\mathcal{B}$  prende una stringa casuale  $H_i \in \{0, 1\}^n$ , aggiunge a  $H_2^{list}$  la tupla  $\langle X_i, H_i \rangle$  e risponde  $H_2(X_i) = H_i$ .

La sfida procede con i seguenti passaggi:

**Setup:**  $\mathcal{B}$  pone  $P_{pub} = P_1$  e  $Q_{ID} = P_2$ , in questo modo si  $P_{pub} = aP$  e  $Q_{ID} = bP$ . Crea la chiave pubblica  $K_{pub} = \langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, n, P, P_{pub}, Q_{ID}, H_2 \rangle$  e la invia ad  $\mathcal{A}$ .

**Sfida:**  $\mathcal{A}$  genera due testi di uguale lunghezza  $M_0, M_1$  su cui essere sfidato.  $\mathcal{B}$  prende una stringa casuale  $R \in \{0, 1\}^n$ , definisce  $C = \langle P_3, R \rangle$  e lo invia ad  $\mathcal{A}$  come oggetto della sfida.

**Risposta:** l'algoritmo  $\mathcal{A}$  genera la sua risposta  $c' \in \{0, 1\}$ .  $\mathcal{B}$  prende una tupla casuale  $\langle X_j, H_j \rangle$  di  $H_2^{list}$  e dà  $X_j$  come risposta al problema BDH.

Osserviamo intanto che la chiave privata, ovviamente non nota nemmeno a  $\mathcal{B}$ , è  $d_{ID} = sQ_{ID} = aQ_{ID} = abP$ , da cui si ottiene

$$\hat{e}(d_{ID}, P_3) = \hat{e}(abP, cP) = \hat{e}(P, P)^{abc} = D.$$

Di conseguenza la decifratura di  $C$  è  $R \oplus H_2(\hat{e}(d_{ID}, P_3)) = R \oplus H_2(D)$ .

Bisogna ora dimostrare che  $\mathcal{B}$  risponde  $D$  al problema BDH con probabilità almeno  $\epsilon/q_{H_2}$ . Affinché questa succeda  $D$  deve comparire in un elemento di  $H_2^{list}$ ; chiamiamo  $\mathcal{H}$  l'evento in cui

l'avversario  $\mathcal{A}$  richiede  $H_2(D)$  durante la simulazione, e quindi  $D$  viene aggiunto alla lista. Concludiamo la dimostrazione grazie ai prossimi due lemmi: nel primo si compara il comportamento di  $\mathcal{A}$  nella simulazione (dove  $\mathcal{B}$  funge sia da sfidante che da oracolo) con quello in un attacco reale, per mostrare che  $\mathcal{H}$  ha uguale probabilità; nel secondo si va a dare una limitazione inferiore a  $\Pr[\mathcal{H}]$ , da cui seguirà la tesi del Lemma 6.3.

**Lemma 6.3.1.**  *$\Pr[\mathcal{H}]$  nella simulazione è uguale a  $\Pr[\mathcal{H}]$  in una attacco reale.*

**Dimostrazione:** Chiamiamo  $\mathcal{H}_l$  l'evento in cui  $\mathcal{A}$  effettua la richiesta  $H_2(D)$  nelle prime  $l$  domande all'oracolo  $H_2$ . Dimostriamo per induzione che  $\Pr[\mathcal{H}_l]$  nella simulazione è uguale a  $\Pr[\mathcal{H}_l]$  in una attacco reale per ogni  $l \geq 0$ .

" $l = 0$  : " ovviamente  $\Pr[\mathcal{H}_0] = 0$  in entrambi i casi.

" $l - 1 \Rightarrow l$  : " consideriamo  $\Pr[\mathcal{H}_{l-1}]$  uguale in entrambe le situazioni. Osserviamo che

$$\begin{aligned} \Pr[\mathcal{H}_l] &= \Pr[\mathcal{H}_l | \mathcal{H}_{l-1}] \Pr[\mathcal{H}_{l-1}] + \Pr[\mathcal{H}_l | \neg \mathcal{H}_{l-1}] \Pr[\neg \mathcal{H}_{l-1}] \\ &= \Pr[\mathcal{H}_{l-1}] + \Pr[\mathcal{H}_l | \neg \mathcal{H}_{l-1}] \Pr[\neg \mathcal{H}_{l-1}] \end{aligned}$$

Quindi basta dimostrare che  $\Pr[\mathcal{H}_l | \neg \mathcal{H}_{l-1}]$  è uguale nella simulazione e nell'attacco reale. Fino a che vale  $\neg \mathcal{H}_{l-1}$  il punto di vista di  $\mathcal{A}$  nella simulazione è lo stesso che avrebbe in un attacco contro un vero sfidante con un vero oracolo, quindi la chiave pubblica e l'oggetto della sfida hanno la stessa distribuzione dell'attacco reale. Similmente, tutte le risposte date dall'oracolo  $H_2$  sono indipendenti e distribuite uniformemente in  $\{0, 1\}^n$ . Di conseguenza  $\Pr[\mathcal{H}_l | \neg \mathcal{H}_{l-1}]$  nella simulazione è uguale a  $\Pr[\mathcal{H}_l | \neg \mathcal{H}_{l-1}]$  in un attacco reale e quindi, per ipotesi di induzione, vale lo stesso per  $\Pr[\mathcal{H}_l]$  e per  $\Pr[\mathcal{H}]$ .

□

**Lemma 6.3.2.** *In un attacco reale  $\Pr[\mathcal{H}] \geq 2\varepsilon$ .*

**Dimostrazione:** In un attacco reale, se  $\mathcal{A}$  non avanza la richiesta  $H_2(D)$  allora la decifrazione di  $C$  è indipendente dalle conoscenze di  $\mathcal{A}$  e quindi vale  $\Pr[c = c' | \neg \mathcal{H}] = 1/2$ . Per le ipotesi su  $\mathcal{A}$  sappiamo che in un attacco reale  $|\Pr[c = c'] - 1/2| \geq \varepsilon$ .

Per ottenere la tesi andiamo a limitare  $\Pr[c = c']$ :

$$\begin{aligned}
 \Pr[c = c'] &= \Pr[c = c' | \neg \mathcal{H}] \Pr[\neg \mathcal{H}] + \Pr[c = c' | \mathcal{H}] \Pr[\mathcal{H}] \\
 &\leq \Pr[c = c' | \neg \mathcal{H}] \Pr[\neg \mathcal{H}] + \Pr[\mathcal{H}] = \frac{1}{2} \Pr[\neg \mathcal{H}] + \Pr[\mathcal{H}] \\
 &\leq \frac{1}{2} + \frac{1}{2} \Pr[\mathcal{H}] \\
 \Pr[c = c'] &\geq \Pr[c = c' | \neg \mathcal{H}] \Pr[\neg \mathcal{H}] = \frac{1}{2} (1 - \Pr[\mathcal{H}]) \\
 &\geq \frac{1}{2} + \frac{1}{2} \Pr[\mathcal{H}]
 \end{aligned}$$

Di conseguenza si ottiene

$$\varepsilon \leq \left| \Pr[c = c'] - 1/2 \right| \leq \frac{1}{2} \Pr[\mathcal{H}]$$

da cui  $\Pr[\mathcal{H}] \geq 2\varepsilon$ . □

Per concludere la dimostrazione del Lemma 6.3 basta osservare che  $D$  appare in una tupla di  $H_2^{list}$  con probabilità almeno  $2\varepsilon$  e la cardinalità della lista è data dal numero di richieste effettuate, ovvero  $q_{H_2}$ . Quindi  $\mathcal{B}$  risponde correttamente al problema BDH con probabilità almeno  $2\varepsilon/q_{H_2}$ . □

**Dimostrazione Teorema 6.1:** Per dimostrare la sicurezza di BasicIdent nell'ipotesi BDH basta mettere assieme i due lemmi per ottenere:

$$\begin{aligned}
 \text{ipotesi BDH} &\stackrel{\text{Lemma 6.3}}{\implies} \text{BasicPub IND-CPA sicuro} \\
 &\stackrel{\text{Lemma 6.2}}{\implies} \text{BasicIdent IND-ID-CPA sicuro}
 \end{aligned}$$

Riguardo al vantaggio dell'avversario, componendo i vantaggi ottenuti nei due lemmi si ottiene

$$\text{Adv}_{\mathcal{G}, \mathcal{B}}(\mathbf{k}) \geq \frac{2\varepsilon(\mathbf{k})}{e(1 + q_E) q_{H_2}}$$

□

## 6.2 FullIdent e sicurezza IND-ID-CCA

A partire dallo schema BasicIdent costruiamo FullIdent, uno schema basato sull'identità con sicurezza IND-ID-CCA in un modello oracolare basato sull'ipotesi BDH. Per farlo utilizziamo la seguente trasformazione.

Sia  $\Pi$  uno schema probabilistico a chiave pubblica e denotiamo con  $\Pi_{pk}(M; r)$  la cifratura di  $M$

utilizzando i bit casuali  $r$  e la chiave pubblica  $pk$ .

Prese due funzioni crittografiche *hash*

$$H_3 : \{0, 1\}^n \times \{0, 1\}^n \longrightarrow \mathbb{Z}_q^*$$

$$H_4 : \{0, 1\}^n \longrightarrow \{0, 1\}^n$$

e presa  $\sigma \in \{0, 1\}^n$  casuale, definiamo la cifratura nello schema ibrido  $\Pi^{hy}$  con

$$\Pi_{pk}^{hy}(M) = \langle \Pi_{pk}(\sigma; H_3(\sigma, M)), H_4(\sigma) \oplus M \rangle.$$

**Osservazione.** Si dimostra che se  $\Pi$  è uno schema di cifratura unidirezionale (*one-way encryption scheme*) allora  $\Pi^{hy}$  è un sistema sicuro contro un attacco *chosen ciphertext* (IND-CCA) in un modello oracolare<sup>1</sup>. Dato che la sicurezza semantica implica sistemi di cifratura *one-way* allora la trasformazione è efficace anche se applicata ad un sistema  $\Pi$  semanticamente sicuro (IND-CPA).

Nel caso di BasicPub nella cifratura dei messaggi si ottiene il seguente schema ibrido:

$$\begin{aligned} \text{BasicPub}^{hy}(M) &= \langle \text{BasicPub}(\sigma; H_3(\sigma, M)), H_4(\sigma) \oplus M \rangle \\ &\in \mathbb{G}_1^* \times \{0, 1\}^n \times \{0, 1\}^n \end{aligned}$$

Applichiamo ora la trasformazione al caso di sistemi basati sull'identità. Gli algoritmi **Setup** e **Extract** di FullIdent sono pressoché uguali a quelli di BasicIdent. Siano  $k$  un parametro di sicurezza e  $\mathcal{G}$  un generatore di parametri BDH.

FullIdent:

**Setup:**

$\mathcal{G}$  genera  $\langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e} \rangle \leftarrow \mathcal{G}(k)$ .

Prende  $P \in \mathbb{G}_1$  generatore casuale e  $s \in \mathbb{Z}_q^*$  casuale; calcola  $P_{pub} = sP$ .

Prende funzioni hash

$$H_1 : \{0, 1\}^* \rightarrow \mathbb{G}_1^*$$

$$H_2 : \mathbb{G}_2 \rightarrow \{0, 1\}^n$$

$$H_3 : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \mathbb{Z}_q^*$$

$$H_4 : \{0, 1\}^n \longrightarrow \{0, 1\}^n$$

---

<sup>1</sup>Nel dettaglio vedere Fujisaki-Okamoto [FO].

Lo spazio dei messaggi e dei testi cifrati sono  $\mathcal{M} = \{0, 1\}^n$  e  $\mathcal{C} = \mathbb{G}_1^* \times \{0, 1\}^n \times \{0, 1\}^n$ .

$$\begin{array}{l} \text{master-key} = s \in \mathbb{Z}_q^* \\ \text{params} = \langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, n, P, P_{pub}, H_1, H_2, H_3, H_4 \rangle \end{array} \left| \leftarrow \text{setup}(k) \right.$$

**Extract:** data una stringa  $ID \in \{0, 1\}^*$  calcola

$$\begin{aligned} Q_{ID} &= H_1(ID) \in \mathbb{G}_1^* \\ d_{ID} &= sQ_{ID} \text{ chiave privata} \end{aligned}$$

**Encrypt:** dato il messaggio  $M \in \mathcal{M}$  con la chiave pubblica ID

prende  $\sigma \in \{0, 1\}^n$  casuale

pone  $r = H_3(\sigma, M)$

calcola il testo cifrato  $C$ , ponendo  $g_{ID} = \hat{e}(Q_{ID}, P_{pub}) \in \mathbb{G}_2$

$$C = \langle rP, \sigma \oplus H_2(g_{ID}^r), M \oplus H_4(\sigma) \rangle$$

**Decrypt:** sia  $C = \langle U, V, W \rangle \in \mathcal{C}$ . Se  $U \notin \mathbb{G}_1^*$  respinge il testo, altrimenti per decriptare il messaggio effettua i seguenti passaggi:

1. calcola  $V \oplus H_2(\hat{e}(d_{ID}, U)) = \sigma$ ;
2. calcola  $W \oplus H_4(\sigma) = M$ ;
3. pone  $r = H_3(\sigma, M)$  e controlla che  $U = rP$ , se non si verifica rigetta il messaggio,
4. altrimenti fornisce  $M$  come decifratura di  $C$ .

Osserviamo che il primo passaggio di **Decrypt** è valido perché

$$\begin{aligned} \hat{e}(d_{ID}, U) &= \hat{e}(sQ_{ID}, rP) = \hat{e}(Q_{ID}, sP)^r = \hat{e}(Q_{ID}, P_{pub})^r \\ &= g_{ID}^r \end{aligned}$$

**Sicurezza:** la verifica della sicurezza di FullIdent utilizza un risultato di Fujisaki-Okamoto ([FO]), che serve ad ottenere la sicurezza IND-CCA dello schema ibrido a partire dalla sicurezza IND-CPA dello schema iniziale BasicPub. Fujisaki e Okamoto dimostrano che, se esiste un avversario IND-CCA con vantaggio non trascurabile contro BasicPub<sup>hy</sup> allora BasicPub non è uno schema sicuro *one-way*. Per i nostri fini ci basta una versione più debole del teorema in cui si dimostra la sicurezza IND-CPA dello schema BasicPub.

**Teorema 6.4. (Fujisaki-Okamoto)** *Supponiamo che  $\mathcal{A}$  sia un avversario IND-CCA con vantaggio  $\epsilon(k)$  quando attacca BasicPub<sup>hy</sup>, con tempo di esecuzione  $t(k)$  e che effettui al massimo  $q_D$  richieste di estrazione di decrittazione e al massimo  $q_{H_3}$  e  $q_{H_4}$  domande agli oracoli  $H_3, H_4$ . Allora esiste  $\mathcal{B}$  avversario IND-CPA di BasicPub con tempo di esecuzione  $t_1(k)$  e vantaggio  $\epsilon_1(k)$ :*

$$\epsilon_1(k) \geq FO_{adv}(\epsilon(k), q_{H_4}, q_{H_3}, q_D) = \frac{1}{2(q_{H_4} + q_{H_3})} [(\epsilon(k) + 1)(1 - 2/q)^{q_D} - 1]$$

$$t_1(k) \leq FO_{time}(t(k), q_{H_4}, q_{H_3}) = t(k) + O((q_{H_4} + q_{H_3})n)$$

Come nel resto del capitolo  $n$  rappresenta la lunghezza dei messaggi e  $q$  l'ordine dei gruppi  $\mathbb{G}_1, \mathbb{G}_2$ .

Enunciamo ora il teorema riguardante la sicurezza di FullIdent.

**Teorema 6.5.** *Supponiamo che le funzioni hash  $H_1, H_2, H_3, H_4$  siano degli oracoli randomizzati. Allora, supponendo che valga l'ipotesi BDH nei gruppi generati da  $\mathcal{G}$ , FullIdent è uno schema di cifratura basato sull'identità sicuro contro un attacco "chosen ciphertext" (IND-ID-CCA).*

*Ovvero, sia  $\mathcal{A}$  un avversario IND-ID-CCA con vantaggio  $\epsilon(k)$  e tempo di esecuzione  $t(k)$  contro lo schema FullIdent. Supponiamo che  $\mathcal{A}$  effettui al massimo  $q_E$  richieste di estrazione di chiave privata,  $q_D$  richieste di decifratura e  $q_{H_2}, q_{H_3}, q_{H_4}$  domande agli oracoli  $H_2, H_3, H_4$ . Allora esiste un algoritmo  $\mathcal{B}$  che risolve il problema BDH per  $\mathcal{G}$  in un tempo di esecuzione  $t_1(k)$  e con un vantaggio  $Adv_{\mathcal{G}, \mathcal{B}}$ :*

$$Adv_{\mathcal{G}, \mathcal{B}}(k) \geq 2FO_{adv} \left( \frac{\epsilon(k)}{e(1 + q_E + q_D)}, q_{H_4}, q_{H_3}, q_D \right) / q_{H_2}$$

$$t_1(k) \leq FO_{time}(t(k), q_{H_4}, q_{H_3})$$

dove le funzione  $FO_{adv}$  e  $FO_{time}$  sono le stesse funzioni del Teorema 6.4.

Per dimostrare il teorema utilizziamo il seguente lemma che, analogamente a quanto fatto con il Lemma 6.2, trasferisce la sicurezza *chosen ciphertext* di un sistema a chiave pubblica a quella di un sistema basato sull'identità.

**Lemma 6.6.** *Sia  $\mathcal{A}$  un avversario IND-ID-CCA con vantaggio  $\epsilon(k)$  contro FullIdent. Supponiamo che  $\mathcal{A}$  effettui al massimo  $q_E > 0$  richieste di estrazione di chiave privata e al massimo  $q_D$  richieste di decifratura.*

Allora esiste un avversario IND-CCA  $\mathcal{B}$  con tempo di esecuzione  $t_{\mathcal{B}} = O(t_{\mathcal{A}})$  e vantaggio contro  $\text{BasicPub}^{hy}$  di almeno  $\frac{\epsilon(k)}{e(1+q_E+q_D)}$ .

**Dimostrazione:** Sia  $\mathcal{B}$  un avversario IND-CCA contro  $\text{BasicPub}^{hy}$ . Lo sfidante processa  $\text{keygen}$  e ottiene la chiave privata  $d_{\text{ID}} = sQ_{\text{ID}}$  che tiene per sé, e la chiave pubblica

$K_{\text{pub}} = \langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, n, P, P_{\text{pub}}, Q_{\text{ID}}, H_2, H_3, H_4 \rangle$  che invia a  $\mathcal{B}$ .

Durante l'attacco  $\mathcal{B}$  interagisce con  $\mathcal{A}$  simulando lo sfidante in un attacco contro FullIdent. Per poterlo fare deve simulare le risposte oracolari e le richieste di estrazione di chiave privata e di decifratura.

**$H_1$ -queries:** esattamente come nel Lemma 6.2. Quando  $\mathcal{A}$  effettua la domanda  $\langle \text{ID}_i \rangle$ :

1. se  $\langle \text{ID}_i \rangle$  appare già in  $\langle \text{ID}_i, Q_i, b_i, \text{coin}_i \rangle \in H_1^{\text{list}}$  allora  $\mathcal{B}$  risponde  $H_1(\text{ID}_i) = Q_i \in \mathbb{G}_1^*$ ;
2. altrimenti,  $\mathcal{B}$  genera una moneta casuale  $\text{coin}_i \in \{0, 1\}$  con  $\Pr[\text{coin}_i = 0] = \delta$ .
3. prende  $b_i \in \mathbb{Z}_q^*$  casuale,

se  $\text{coin}_i = 0$ , calcola  $Q_i = b_i P$

se  $\text{coin}_i = 1$ , calcola  $Q_i = b_i Q_{\text{ID}}$

4.  $\mathcal{B}$  aggiunge  $\langle \text{ID}_i, Q_i, b_i, \text{coin}_i \rangle$  alla lista e risponde  $H_1(\text{ID}_i) = Q_i$ .

La simulazione di attacco a FullIdent avviene così:

**Setup:** come nel Lemma 6.2  $\mathcal{B}$  crea i parametri del sistema dalla chiave pubblica  $K_{\text{pub}}$ , a cui aggiunge le due funzioni *hash* oracolari  $H_3, H_4$ .

**Fase 1:**  $\mathcal{A}$  inoltra richieste di uno dei seguenti tipi:

- o domanda di **estrazione** di chiave privata  $\langle \text{ID}_i \rangle$ : gestite come nel Lemma 6.2.  $\mathcal{B}$  ottiene  $H_1(\text{ID}_i) = Q_i$  da  $H_1^{\text{list}}$ ; sia  $\langle \text{ID}_i, Q_i, b_i, \text{coin}_i \rangle$  la corrispondente tupla,
  1. se  $\text{coin}_i = 1 \rightarrow \text{stop}$ ,  $\mathcal{B}$  interrompe l'attacco;
  2. se  $\text{coin}_i = 0 \rightarrow d_i = b_i P_{\text{pub}} \in G_1^*$  chiave privata  $sQ_i$  corrispondente a  $\text{ID}_i$ .
- o domanda di **decifratura**  $\langle \text{ID}_i, C_i \rangle$ , con  $C_i = \langle U_i, V_i, W_i \rangle$ :
  - 1)  $\mathcal{B}$  esegue la  $H_1$ -query ed ottiene  $H_1(\text{ID}_i) = Q_i$  con  $\langle \text{ID}_i, Q_i, b_i, \text{coin}_i \rangle$  la corrispondente tupla;
  - 2) se  $\text{coin}_i = 0$ , ottiene la chiave privata  $d_i$  e di conseguenza la decifratura di  $C_i$  da inviare ad  $\mathcal{A}$ ;
  - 3) se  $\text{coin}_i = 1$  pone  $C'_i = \langle b_i U_i, V_i, W_i \rangle$ . Inoltra la richiesta di decifratura  $\langle C'_i \rangle$  allo sfidante di  $\text{BasicPub}^{hy}$  ed invia la risposta ricevuta a  $\mathcal{A}$ .

**Sfida:** terminata la **Fase 1**  $\mathcal{A}$  genera la chiave pubblica  $ID_{ch}$  e i messaggi  $M_0, M_1$  su cui essere sfidato.

- 1)  $\mathcal{B}$  trasmette  $M_0, M_1$  allo sfidante e in risposta riceve la cifratura con  $\text{BasicPub}^{hy}$  di  $M_c$  con  $c \in \{0, 1\}$  casuale:  $C = \langle U, V, W \rangle$ .  
Esegue l'algoritmo  $H_1$ -query per ottenere  $H_1(ID_{ch}) = Q \in \mathbb{G}_1^*$ . Sia  $\langle ID_{ch}, Q, b, coin \rangle$  la corrispondente tupla.
- 2) se  $coin = 0$ ,  $\mathcal{B}$  termina l'attacco a  $\text{BasicPub}^{hy}$  e riporta un fallimento;
- 3) se  $coin = 1$  pone  $C' = \langle b^{-1}U, V, W \rangle$ , dove  $b^{-1}$  è l'inverso modulo  $q$ . Risponde ad  $\mathcal{A}$  con la sfida  $C'$ , che risulta essere la cifratura  $\text{FullIdent}$  di  $M_c$  tramite la chiave pubblica  $ID_{ch}$ .

**Fase 2:** esattamente come nella **Fase 1**  $\mathcal{B}$  risponde alle richieste di estrazione di chiave privata e alle richieste di decifratura. Se però la domanda coincide con  $C = \langle U, V, W \rangle$  la sfida termina e  $\mathcal{B}$  riporta un fallimento.

**Risposta:**  $\mathcal{A}$  dà la sua risposta  $c'$  per  $c$  e  $\mathcal{B}$  risponde con la stessa  $c'$ .

Osserviamo che, nel caso in cui  $coin_i = 1$  in una richiesta di decifratura allora la decifratura  $\text{FullIdent}$  di  $C_i$  con la chiave  $d_i$  è la stessa della decifratura  $\text{BasicPub}^{hy}$  di  $C'_i$  con la chiave  $d_{ID}$ , dove  $d_i = sQ_i$  è la chiave privata di  $\text{FullIdent}$  corrispondente a  $ID_i$ . Basta che gli algoritmi di decifratura diano lo stesso risultato nel primo passaggio per ottenere  $\sigma$  perché poi il messaggio in chiaro si ottiene nella stessa maniera:  $W_i \oplus H_4(\sigma) = M$ .

Dato che  $coin_i = 1$ , vale  $Q_i = b_i Q_{ID}$  e ricordiamo che  $d_{ID} = sQ_{ID}$  e  $d_i = sQ_i$ . Nelle due decifrate si ha

$$\begin{aligned} \text{FullIdent} : \quad & V_i \oplus H_2(\hat{e}(d_i, U_i)) = \sigma_F \\ \text{BasicPub}^{hy} : \quad & V_i \oplus H_2(\hat{e}(d_{ID}, b_i U_i)) = \sigma_B \end{aligned}$$

e le due  $\sigma$  ottenute coincidono poiché

$$\begin{aligned} \hat{e}(d_{ID}, b_i U_i) &= \hat{e}(sQ_{ID}, b_i U_i) = \hat{e}(sb_i Q_{ID}, U_i) = \hat{e}(sQ_i, U_i) \\ &= \hat{e}(d_i, U_i). \end{aligned}$$

Inoltre, le risposte alle  $H_1$ -queries sono indipendenti e uniformemente distribuite in  $\mathbb{G}_1^*$  e quindi sono uguali a quelle di un attacco reale, tutte le risposte alle domande di estrazione di chiave privata e di decifratura sono accettabili e il testo cifrato  $C'$  oggetto della sfida con  $\mathcal{A}$  è effettivamente la cifratura  $\text{FullIdent}$  di  $M_c$  per  $c \in \{0, 1\}$  casuale. Quindi, se  $\mathcal{B}$  non interrompe l'attacco, la visuale di  $\mathcal{A}$  è uguale a quella di un attacco reale e di conseguenza, per le ipotesi del lemma,  $\left| \Pr[c = c'] - \frac{1}{2} \right| \geq \epsilon$ .

Studiamo l'evento in cui l'attacco fallisca durante la simulazione e la sua probabilità. Questo può avvenire per tre ragioni diverse:

- i) durante la Fase 1 o nella Fase 2  $\mathcal{A}$  effettua una richiesta di estrazione di chiave privata che causa un fallimento (caso in cui  $coin_i = 1$ ), chiamiamo questo evento  $\mathcal{E}_1$ ;
- ii) per la sfida  $\mathcal{A}$  sceglie una chiave pubblica  $ID_{ch}$  che fa sì che  $\mathcal{B}$  si abortisca (caso  $coin = 0$ , con probabilità  $\delta$ ), chiamiamo questo evento  $\mathcal{E}_2$ ;
- iii) durante la **Fase 2**  $\mathcal{A}$  effettua una richiesta di decifratura  $\langle ID_i, C_i \rangle$ , con  $C_i = C$ , che fa in modo che  $\mathcal{B}$  avanzi la richiesta di decifratura  $\langle C \rangle$  allo sfidante di BasicPub<sup>hy</sup>, chiamiamo questo evento  $\mathcal{E}_3$ .

Vogliamo dimostrare che per la probabilità che  $\mathcal{B}$  non interrompa l'attacco vale

$$\Pr[\neg\mathcal{E}_1 \wedge \neg\mathcal{E}_2 \wedge \neg\mathcal{E}_3] \geq \delta^{q_E + q_D}(1 - \delta).$$

Lo dimostriamo per induzione sul numero massimo di domande  $q_E + q_D$  fatte dall'avversario. Sia  $i = q_E + q_D$  e sia  $\mathcal{E}^{0\dots i}$  l'evento in cui  $\mathcal{E}_1 \vee \mathcal{E}_3$  si verifica entro  $i$  domande,  $\neg\mathcal{E}^{0\dots i}$  rappresenta l'evento in cui né  $\mathcal{E}_1$  né  $\mathcal{E}_3$  si verificano nelle prime  $i$  domande,  $(\neg\mathcal{E}_1 \wedge \neg\mathcal{E}_3)$ . Sia invece  $\mathcal{E}^i$  l'evento in cui  $\mathcal{E}_1 \vee \mathcal{E}_3$  si avvera esattamente alla  $i$ -esima domanda.

Osserviamo che

$$\Pr[\neg\mathcal{E}_1 \wedge \neg\mathcal{E}_2 \wedge \neg\mathcal{E}_3] = \Pr[\neg\mathcal{E}_1 \wedge \neg\mathcal{E}_3 | \neg\mathcal{E}_2] \Pr[\neg\mathcal{E}_2] \geq \Pr[\neg\mathcal{E}_1 \wedge \neg\mathcal{E}_3 | \neg\mathcal{E}_2](1 - \delta).$$

Dimostriamo per induzione che

$$\Pr[\neg\mathcal{E}^{0\dots i} | \neg\mathcal{E}_2] \geq \delta^i.$$

" $i = 0$ ": caso banale poiché  $\Pr[\neg\mathcal{E}^{0\dots 0}] = 1$ .

" $i - 1 \Rightarrow i$ ": Si ha

$$\begin{aligned} \Pr[\neg\mathcal{E}^{0\dots i} | \neg\mathcal{E}_2] &= \Pr[\neg\mathcal{E}^{0\dots i} | \neg\mathcal{E}^{0\dots i-1} \wedge \neg\mathcal{E}_2] \Pr[\neg\mathcal{E}^{0\dots i-1} | \neg\mathcal{E}_2] \\ &= \Pr[\neg\mathcal{E}^i | \neg\mathcal{E}^{0\dots i-1} \wedge \neg\mathcal{E}_2] \Pr[\neg\mathcal{E}^{0\dots i-1} | \neg\mathcal{E}_2] \\ &\geq \Pr[\neg\mathcal{E}^i | \neg\mathcal{E}^{0\dots i-1} \wedge \neg\mathcal{E}_2] \delta^{i-1} \end{aligned}$$

L'ultima uguaglianza è giustificata dal fatto che, dato che l'evento non si è avverato nelle prime  $i - 1$  richieste, chiedere che non si avveri nelle prime  $i$  coincide col chiedere che non succeda nella  $i$ -esima.

Ora basta limitare

$$q_i = \Pr[\neg\mathcal{E}^i | \neg\mathcal{E}^{0\dots i-1} \wedge \neg\mathcal{E}_2],$$

ovvero limitare la probabilità che l'evento  $\mathcal{E}_1 \vee \mathcal{E}_3$  non si verifichi alla richiesta  $i$ -esima, sapendo che non si è verificato nelle prime  $i - 1$  e sapendo che non si è verificato  $\mathcal{E}_2$ . La richiesta può essere una richiesta di estrazione di chiave privata oppure una richiesta di decifratura, che in caso supponiamo si verifichi nella Fase 2, altrimenti non ha effetto su  $\mathcal{E}_3$ .

Consideriamo la tupla di  $H_1^{list} \langle \text{ID}_i, Q_i, b_i, \text{coin}_i \rangle$ , con  $H_1(\text{ID}_i) = Q_i$ . Se  $\text{coin}_i = 0$  la richiesta non può causare l'evento  $\mathcal{E}_1$ , né l'evento  $\mathcal{E}_3$  poiché  $\mathcal{B}$  inoltra la richiesta di decifratura a  $\text{BasicPub}^{hy}$  solo se  $\text{coin}_i = 1$ .

Consideriamo 4 casi separati e nei primi tre supponiamo  $\text{ID}_i \neq \text{ID}_{ch}$ .

**Caso 1:** la richiesta  $i$ -esima è la prima che contiene  $\text{ID}_i$  e quindi  $\Pr[\text{coin}_i = 0] = \delta$  e quindi  $q_i \geq \delta$ .

**Caso 2:** la chiave pubblica  $\text{ID}_i$  è apparsa in una precedente richiesta di estrazione di chiave privata. Dato che vale  $\neg \mathcal{E}^{0 \dots i-1}$ , allora questa precedente domanda non ha interrotto l'attacco e quindi deve valere  $\text{coin}_i = 0$ , da cui  $q_i = 1$ .

**Caso 3:**  $\text{ID}_i$  è apparsa in una precedente richiesta di decifratura. Per ipotesi questa richiesta non ha fatto avverare l'evento  $\mathcal{E}^{0 \dots i-1}$  e quindi o  $\text{coin}_i = 0$  o  $\text{coin}_i$  è indipendente dal punto di vista di  $\mathcal{A}$ . In ogni caso vale  $q_i \geq \delta$ .

**Caso 4:** ID coincide con la chiave pubblica  $\text{ID}_{ch}$  su cui è sfidato  $\mathcal{A}$ . Di conseguenza la richiesta  $i$ -esima non può essere una richiesta di estrazione di chiave privata ma è una richiesta di decifratura  $\langle \text{ID}_i, C_i \rangle$ . Poiché non si è verificato  $\mathcal{E}_2$  sappiamo che  $\text{coin}_i = 1$  e quindi  $\mathcal{B}$  invia allo sfidante di  $\text{BasicPub}^{hy}$  una domanda di decifratura  $C'_i$ . Dato che  $C_i \neq C'$  ( $U_i \neq b_i^{-1}U$ ), segue che  $C'_i \neq C$  ( $b_i U_i \neq U$ ), dove  $C$  è il testo cifrato su cui è sfidato  $\mathcal{B}$ . Di conseguenza non può verificarsi l'evento  $\mathcal{E}_3$  e quindi in questo caso  $q_i = 1$ . Quindi, a prescindere dal tipo di istanza  $i$ -esima vale  $q_i \geq \delta$ , da cui

$$\begin{aligned} \Pr[\neg \mathcal{E}^{0 \dots i} | \neg \mathcal{E}_2] &\geq \Pr[\neg \mathcal{E}^i | \neg \mathcal{E}^{0 \dots i-1} \wedge \neg \mathcal{E}_2] \delta^{i-1} \\ &\geq \delta^i \end{aligned}$$

Sostituendo ora  $i = q_E + q_D$  otteniamo la limitazione alla probabilità di successo di  $\mathcal{B}$ :

$$\Pr[\neg \mathcal{E}_1 \wedge \neg \mathcal{E}_2 \wedge \neg \mathcal{E}_3] \geq \delta^{q_E + q_D} (1 - \delta)$$

che è ottimizzata per  $\delta_{opt} = 1 - \frac{1}{q_E + q_D + 1}$  con il quale si ottiene

$$\Pr[\neg \mathcal{E}_1 \wedge \neg \mathcal{E}_2 \wedge \neg \mathcal{E}_3] \geq \frac{1}{e(1 + q_E + q_D)}.$$

Di conseguenza otteniamo la tesi del Lemma 6.6, ovvero che il vantaggio di  $\mathcal{B}$  è almeno  $\frac{\epsilon(k)}{e^{(1+q_E+q_D)}}$ .  $\square$

Possiamo quindi dimostrare il Teorema 6.5.

**Dimostrazione Teorema 6.5:** Componendo i risultati precedenti otteniamo che

per il Lemma 6.6 l'esistenza di un avversario IND-ID-CCA efficace contro FullIdent implica l'esistenza di un avversario efficace contro BasicPub<sup>hy</sup>;

per il Teorema 6.4 un avversario IND-CCA di BasicPub<sup>hy</sup> permette di costruire un avversario efficace contro BasicPub;

per il Lemma 6.3 tramite un avversario IND-CPA efficace contro BasicPub si può risolvere il problema BDH.

$$\begin{aligned} \text{ipotesi BDH} &\stackrel{\text{Lemma 6.3}}{\implies} \text{BasicPub IND-CPA sicuro} \\ &\stackrel{\text{Teorema 6.4}}{\implies} \text{BasicPub}^{hy} \text{ IND-CCA sicuro} \\ &\stackrel{\text{Lemma 6.6}}{\implies} \text{FullIdent IND-ID-CCA sicuro} \end{aligned}$$

Componendo le varie limitazioni si ottiene la tesi.  $\square$

## 6.3 Sistema IBE

Tramite i risultati appena ottenuti siamo riusciti a costruire uno schema basato sull'identità con sicurezza IND-ID-CCA. Per completare la costruzione dello schema IBE di Boneh e Franklin bisogna però apportare alcune modifiche, in maniera da ovviare ad alcune difficoltà tecniche e di efficienza del precedente schema, che sarà modificato nello schema definitivo FullIdent'.

Per concludere andremo poi a definire una versione modificata del *Weil Pairing* in maniera da ottenere una mappa bilineare *ammissibile* e descriveremo i parametri BDH.

### 6.3.1 FullIdent'

In ambito teorico abbiamo appena dimostrato che FullIdent è un sistema IBE con sicurezza IND-ID-CCA. Però nella pratica il sistema appena costruito non è di facile attuazione, soprattutto per l'utilizzo della funzione *hash*  $H_1 : \{0, 1\}^* \rightarrow \mathbb{G}_1^*$ .

Nel sistema di Boneh e Franklin come gruppo  $\mathbb{G}_1$  consideriamo il gruppo di torsione di ordine  $E[q]$  di una curva ellittica. Essendo un gruppo molto particolare è difficile costruire una funzione

*hash* che mappa direttamente dentro a questo gruppo. Di conseguenza dobbiamo rilassare i vincoli su  $H_1$  mappandola in un insieme  $A \subseteq \{0, 1\}^*$  invece che in  $E[q]$ . Poi mappiamo  $A$  in  $\mathbb{G}_1^*$  con una funzione di codifica deterministica.

**Codifiche ammissibili:** Sia  $\mathbb{G}_1$  un gruppo e  $A \subseteq \{0, 1\}^*$  un insieme finito. Una funzione di codifica  $L : A \rightarrow \mathbb{G}_1$  si dice *ammissibile* se soddisfa le seguenti:

1. Computabile: esiste un algoritmo deterministico efficiente per calcolare  $L(x) \forall x \in A$ .
2. *l-to-l*: ogni  $y$  in  $\mathbb{G}_1^*$  ha esattamente  $l$  controimmagini,  $|L^{-1}(y)| = l \forall y \in \mathbb{G}_1^*$ . Di conseguenza deve valere  $|A| = l \cdot |\mathbb{G}_1^*|$ .
3. campionabile: esiste un algoritmo randomizzato efficiente  $\mathcal{L}_S$  tale che, per ogni  $y \in \mathbb{G}_1^*$ ,  $\mathcal{L}_S(y)$  induce una distribuzione uniforme su  $L^{-1}(y)$ , ovvero  $\mathcal{L}_S(y)$  è un elemento uniformemente casuale in  $L^{-1}(y)$ .

Modifichiamo FullIdent per ovviare al problema dato dal mappare  $H_1$  direttamente in  $\mathbb{G}_1$ .

FullIdent' :

**Setup:** Come nello schema FullIdent tranne per  $H_1$  che è sostituita da una funzione *hash*  $H'_1 : \{0, 1\}^* \rightarrow A$ . Nei parametri del sistema è inclusa anche la descrizione di una mappa di codifica ammissibile  $L : A \rightarrow \mathbb{G}_1^*$ .

**Extract, Encrypt:** esattamente come in FullIdent, tranne per il calcolo di  $Q_{ID} = L(H'_1(ID)) \in \mathbb{G}_1^*$ .

**Decrypt:** come in FullIdent.

Con questa piccola modifica otteniamo uno schema effettivamente realizzabile e con sicurezza contro un attacco *chosen ciphertext*.

**Teorema 6.7.** *Sia  $\mathcal{A}$  un avversario IND-ID-CCA di FullIdent'. Supponiamo che  $\mathcal{A}$  abbia vantaggio  $\epsilon(k)$ , tempo di esecuzione  $t_{\mathcal{A}}$  e che effettui al massimo  $q_{H_1}$  richieste all'oracolo  $H'_1$ . Allora esiste un avversario IND-ID-CCA  $\mathcal{B}$  con vantaggio  $\epsilon(k)$  contro FullIdent e con tempo di esecuzione  $t_{\mathcal{B}} = t_{\mathcal{A}} + q_{H_1} \cdot t(L_S)$ .*

**Dimostrazione:** L'algoritmo  $\mathcal{B}$  attacca FullIdent utilizzando l'aiuto dell'algoritmo  $\mathcal{A}$ . Tutte le richieste di decifratura, di estrazione di chiave privata e tutte le domande alle funzioni *hash* oracolari fatte da  $\mathcal{A}$  vengono inviate direttamente allo sfidante FullIdent e le risposte vengono rimandate ad  $\mathcal{A}$  così come stanno.

Nel caso di richieste alla funzione *hash*  $H'_1$   $\mathcal{B}$  si deve comportare in maniera diversa, dato che ha accesso solo alla funzione  $H_1 : \{0, 1\}^* \rightarrow \mathbb{G}_1^*$ . Per rispondere a queste nuove domande tiene

una lista,  $(H'_1)^{list}$ , di tuple  $\langle ID_j, y_j \rangle$ . La lista inizialmente è vuota. Quando  $\mathcal{A}$  chiede all'oracolo  $H'_1$  la valutazione in  $ID_i$ ,  $\mathcal{B}$  procede con i seguenti passaggi:

1. se  $ID_i$  appare già in una tupla di  $(H'_1)^{list}$ , risponde  $H'_1(ID_i) = y_i \in A$ .
2. Altrimenti,  $\mathcal{B}$  interroga l'oracolo  $H_1$  per ottenere  $H_1(ID_i) = \alpha \in \mathbb{G}_1^*$ .
3.  $\mathcal{B}$  processa l'algoritmo di campionamento  $\mathcal{L}_S(\alpha)$  per ottenere un elemento casuale  $y \in L^{-1}(\alpha)$ .
4. Aggiunge la tupla  $\langle ID_i, y \rangle$  alla lista e risponde con  $H'_1(ID_i) = y \in A$ .

Dato che  $\alpha$  è distribuito in maniera uniforme in  $\mathbb{G}_1^*$  ed  $L$  è una mappa  $l$ -to- $l$  segue che  $y$  è distribuito in maniera uniforme in  $A$ .

La visione di  $\mathcal{A}$  è la stessa di quella in un attacco reale, a prescindere dalla richieste oracolare che avanza, di conseguenza  $\mathcal{B}$  ha lo stesso vantaggio  $\epsilon(k)$  nel vincere la sfida.  $\square$

### 6.3.2 Weil Pairing Modificato

Prendiamo un primo  $p \equiv 2 \pmod{3}$ , consideriamo  $q$  fattore primo di  $p + 1$  e sia  $E$  la curva ellittica su  $\mathbb{F}_p$  definita da  $y^2 = x^3 + 1$ .

Dato che  $x^3 + 1$  è una permutazione di  $\mathbb{F}_p$  segue che  $|E(\mathbb{F}_p)| = p + 1$ . Sia  $P \in E(\mathbb{F}_p)$  un punto di ordine  $q$  e sia  $\mathbb{G}_1$  il sottogruppo generato da  $P$ , ovvero il gruppo di torsione di ordine  $q$ ,  $\mathbb{G}_1 = E[q]$ .

Preso un elemento  $y_0 \in \mathbb{F}_p$  qualsiasi esiste un unico punto  $(x_0, y_0) \in E(\mathbb{F}_p)$ , con  $x_0 = (y_0^2 - 1)^{1/3} \in \mathbb{F}_p$ .

Sia  $\zeta \in \mathbb{F}_{p^2}$  una soluzione di  $x^3 - 1 = 0$ , con  $\zeta \neq 1$ . La mappa  $\phi(x, y) = (\zeta x, y)$  è un automorfismo del gruppo dei punti di  $E$ . Per ogni punto  $Q = (x, y) \in E(\mathbb{F}_p)$ , si ottiene  $\phi(Q) \in E(\mathbb{F}_{p^2})$  ma  $\phi(Q) \notin E(\mathbb{F}_p)$  e di conseguenza i due punti  $Q$  e  $\phi(Q)$  sono linearmente indipendenti e quindi generano il gruppo di torsione  $E[q] \simeq \mathbb{Z}_q \times \mathbb{Z}_q$ .

Sia  $\mathbb{G}_2$  il sottogruppo di  $\mathbb{F}_{p^2}^*$  di ordine  $q$ , ovvero il sottogruppo delle radici  $q$ -esime dell'unità,  $\mu_q$ . Il Weil Pairing su  $E$  è la mappa  $e : E[q] \times E[q] \rightarrow \mu_q$  e per ogni  $P, Q \in E[q]$  soddisfa  $e(Q, R) = 1$ , ovvero è degenere su  $E[q]$ .

Per ottenere una mappa non degenere definiamo il *Weil Pairing modificato*

$$\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$$

$$\hat{e}(P, Q) = e(P, \phi(Q))$$

Si ha che  $\hat{e}$  soddisfa le seguenti proprietà:

1. bilineare: per ogni  $P, Q \in \mathbb{G}_1$  e per ogni  $a, b \in \mathbb{Z}$  vale  $\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$ .
2. Non degenerare: se  $P$  è un generatore di  $\mathbb{G}_1$ , allora  $\hat{e}(P, P) \in \mathbb{F}_{p^2}^*$  è un generatore di  $\mathbb{G}_2$ .
3. Computabile: dati  $P, Q \in \mathbb{G}_1$  esiste un algoritmo efficiente per calcolare  $\hat{e}(P, Q) \in \mathbb{G}_2$ , l'algoritmo di Miller, con tempo di esecuzione comparabile all'esponenziazione in  $\mathbb{F}_p$ .

### 6.3.3 Parametri BDH

Sia  $k \in \mathbb{Z}^+$  un parametro di sicurezza. Il generatore di parametri BDH  $\mathcal{G}$  prende un primo casuale di  $k$  bit,  $q$ , e trova il più piccolo primo  $p$  tale che

$$p = 2 \bmod 3,$$

$$q \text{ divide } p + 1,$$

$$q^2 \text{ non divide } p + 1.$$

Sia  $p = lq - 1$ , con  $l$  non divisibile per  $q$  altrimenti  $q^2$  dividerebbe  $p + 1$ .

Il gruppo  $\mathbb{G}_1$  è il sottogruppo dei punti di torsione di ordine  $q$  della curva ellittica  $y^2 = x^3 + 1$  su  $\mathbb{F}_p$ .  $\mathbb{G}_2$  è il sottogruppo delle radici  $q$ -esime dell'unità in  $\mathbb{F}_{p^2}^*$ . La mappa bilineare  $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$  è data dal Weil Pairing modificato.

In questo modo abbiamo ottenuto i parametri del sistema

$$\langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e} \rangle.$$

### 6.3.4 MapToPoint

Per definire il sistema IBE FullIdent' dobbiamo ancora descrivere la *mappa di codifica ammissibile*  $L : A \rightarrow \mathbb{G}_1^*$ , che chiameremo MapToPoint. Prendiamo come insieme  $A = \mathbb{F}_p$ . Supponiamo di avere già una funzione *hash*  $H_1 : \{0, 1\}^* \rightarrow \mathbb{F}_p$ . La mappa MapToPoint:  $A \rightarrow \mathbb{G}_1^*$  opera in questo modo:

1. dato  $y_0 \in \mathbb{F}_p$  calcola  $x_0 = (y_0^2 - 1)^{1/3} = (y_0^2 - 1)^{(2p-1)/3}$ ;
2. pone  $Q = (x_0, y_0) \in E(\mathbb{F}_p)$  e  $Q_{ID} = lQ \in G_1$ ;
3. definisce come immagine di  $y_0$   $\text{MapToPoint}(y_0) = Q_{ID}$ .

Osserviamo che ci sono  $l - 1$  valori di  $y_0 \in \mathbb{F}_p$  per i quali vale  $lQ = l(x_0, y_0) = \infty$ , ovvero i punti diversi dal punto all'infinito di ordine che divide  $l$ . Chiamiamo  $B \subset \mathbb{F}_p$  l'insieme di questi  $y_0$ .

Se  $H_1(\text{ID})$  è uno di questi valori, allora  $Q_{ID}$  è l'elemento neutro di  $\mathbb{G}_1$ . Fortunatamente è improbabile che questo accada, la probabilità è  $1/q < 1/2^k$ , e quindi possiamo dire che  $H_1(\text{ID})$  genera

solo valori in  $\mathbb{F}_p - B$ ,  $H_1 : \{0, 1\}^* \rightarrow \mathbb{F}_p - B$ .

Applicando diverse funzioni *hash* a ID in maniera consecutiva si può estendere MapToPoint in maniera che possa gestire anche i valori  $y_0 \in B$ .

**Lemma 6.8.** *La funzione MapToPoint  $L : \mathbb{F}_p - B \rightarrow \mathbb{G}_1^*$  è una mappa di codifica ammissibile.*

**Dimostrazione:** La mappa è computabile in maniera efficiente ed è una funzione  $l - to - 1$ . Mostriamo che è anche campionabile.

Sia  $P$  un generatore di  $E(\mathbb{F}_p)$ . Dato  $Q \in \mathbb{G}_1^*$  l'algoritmo  $\mathcal{L}_S$  opera nella seguente maniera:

1. prende un  $b \in \{0, \dots, l - 1\}$  casuale,
2. calcola  $Q' = l^{-1}Q + bqP = (x, y)$  con  $l^{-1}$  inverso di  $l \bmod q$ ,
3. fornisce  $\mathcal{L}_S(Q) = y \in \mathbb{F}_p$ .

Quindi, come richiesto, l'algoritmo fornisce un elemento casuale preso tra gli  $l$  elementi di  $\text{MapToPoint}^{-1}(Q)$ . □

Quindi, utilizzando gli oggetti appena descritti e i risultati sulla sicurezza precedentemente ottenuti possiamo enunciare il seguente corollario:

**Corollario 6.9.** *Con i parametri generati da  $\mathcal{G}$ , la mappa di codifica MapToPoint e supponendo che  $\mathcal{G}$  soddisfi l'ipotesi BDH, il sistema FullIdent' è un sistema IBE con sicurezza chosen ciphertext in un modello oracolare.*

**Efficienza:** Gli algoritmi Setup e Extract sono piuttosto semplici e come operazione principale hanno una moltiplicazione sulla curva  $E$ . L'algoritmo Encrypt deve calcolare il Weil Pairing  $\hat{e}(Q_{ID}, P_{pub}) = g_{ID}$  e lo fa in maniera indipendente dal messaggio, quindi il calcolo può essere fatto una volta sola. Da lì in poi l'efficienza del sistema è pressoché uguale alla cifratura di ElGamal. La decifratura richiede il calcolo di  $\hat{e}(d_{ID}, U)$ , delle valutazioni delle funzioni *hash* e la somma bit a bit delle stringhe.

# Bibliografia

- [WL] Washington, Lawrence C. *Elliptic curves : number theory and cryptography*, Chapman and Hall/CRC, Boca Raton, 2008
- [BF] D. Boneh and M. Franklin. Identity based encryption from the Weil pairing. In *Advances in cryptology, Crypto 2001 (Santa Barbara, CA)*, volume 2139 of *Lecture Notes in Comput. Sci.*, pages 213–229. Springer-Verlag, Berlin, 2001.
- [SA] A. Shamir, "Identity-based cryptosystems and signature schemes", in *Advances in Cryptology - Crypto '84*, Lecture Notes in Computer Science, Vol. 196, Springer-Verlag, pp. 47-53, 1984.
- [DH] W. Diffie e M.E. Hellman, "New directions in cryptography", *IEEE Transactions on Information Theory* vol. 22, no. 6, pp. 644-654, 1976.
- [PS] Pass R. and SHelat A. *A course in criptography*, Pass/shelat 2010
- [TA] H. Tanaka, "A realization scheme for the identity-based cryptosystem", in *Advances in Cryptology - Crypto '87*, Lecture Notes in Computer Science, Vol. 293, Springer-Verlag, pp. 341-349, 1987.
- [JN] A. Joux, K. Nguyen, "Separating Decision Diffie-Hellman from Diffie-Hellman in cryptographic groups", *J. Cryptology* 16(4), pp. 239-247, 2003.
- [FO] E. Fujisaki and T. Okamoto, "Secure integration of asymmetric and symmetric encryption schemes", in *Advances in Cryptology - Crypto '99*, Lecture Notes in Computer Science, Vol. 1666, Springer-Verlag, pp. 537-554, 1999.
- [BDPR] M. Bellare, A. Desai, D. Pointcheval, P. Rogaway, "Relations among notions of security for public-key encryption schemes", in *Advances in Cryptology - Crypto '98*, Lecture Notes in Computer Science, Vol. 1462, Springer-Verlag, 1998.

## BIBLIOGRAFIA

---

- [RS] C. Rackoff, D. Simon, "Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack", in *Advances in Cryptology - Crypto '91*, Lecture Notes in Computer Science, Vol. 547, Springer-Verlag, 1991.
- [GM] S. Goldwasser and S. Micali, "Probabilistic encryption". *Journal of Computer and System Sciences*, 28:270-299, 1984.
- [DB] D. Boneh, "The decision Diffie-Hellman problem", in *Proc. Third Algorithmic Number Theory Symposium*, Lecture Notes in Computer Science, Vol. 1423, Springer-Verlag, pp. 48-63, 1998.
- [MA] U. Maurer, "Towards proving the equivalence of breaking the Diffie-Hellman protocol and computing discrete logarithms", in *Advances in Cryptology - Crypto '94*, Lecture Notes in Computer Science, Vol. 839, pp. 271-281, 1994.
- [MW] U. Maurer and S. Wolf. "The relationship between breaking the Diffie-Hellman protocol and computing discrete logarithms", *SIAM Journal on Computing*, 28(5):1689–1721, 1999.
- [MV] V. Miller, "Short programs for functions on curves", unpublished manuscript, 1986.
- [MOV] A. Menezes, T. Okamoto, S. Vanstone, "Reducing elliptic curve logarithms to logarithms in a finite field", *IEEE Tran. on Info. Th.*, Vol. 39, pp. 1639-1646, 1993.
- [EA] A. Enge, *Elliptic curves and their applications to cryptography: An introduction*. Kluwer Academic Publishers, Dordrecht, 1999.

# **Ringraziamenti**

Ringrazio un po' tutti quanti (così nessuno si sente escluso), in particolar modo Arianna per avermi sopportato in questi ultimi mesi...