POLITECNICO DI TORINO

Corso di Laurea
in Ingegneria Matematica

Tesi di Laurea Magistrale

Crittografia post quantistica basata sui reticoli: dagli schemi di firma di Lyubashevsky al protocollo ABDLOP



Relatori
Prof. Antonio Josè Di Scala

Candidato
Stefano Maria Caprioli

Anno Accademico 2025-2026

Sommario

La necessità di schemi crittografici resistenti ad avversari quantistici ha portato a un forte interesse verso le costruzioni basate sui reticoli, fondate sulla difficoltà computazionale di problemi quali Short Integer Solution (SIS) e Learning With Errors (LWE).

La presente tesi parte con lo studio degli schemi di firma di Lyubashevsky. Tali costruzioni consentono di ottenere firme efficienti e sicure, ma non forniscono direttamente le proprietà necessarie per la costruzione di sistemi avanzati di zero-knowledge proof. In particolare, l'assenza di una forma adeguata di special soundness limita l'utilizzo di tali protocolli come mattoni fondamentali per la realizzazione di prove non interattive e sistemi di credenziali anonime.

Per superare queste limitazioni viene analizzato lo schema di commitment ABDLOP, una costruzione che estende le firme di Lyubashevsky introducendo una struttura in grado di supportare proprietà di proof of knowledge. Il contributo principale della tesi consiste nello studio teorico e sperimentale di tale protocollo, con particolare attenzione alla scelta e all'analisi dei parametri crittografici. Vengono esaminate le relazioni tra dimensione dei moduli, varianza di distribuzioni gaussiane discrete, limiti di norma, corretta scelta del challenge space e probabilità di accettazione nel rejection sampling, evidenziando il loro impatto sulle proprietà di correttezza, zero-knowledge ed efficienza computazionale.

A supporto dell'analisi teorica è stata sviluppata un'implementazione in Python del protocollo ABDLOP, utilizzata per valutare sperimentalmente il comportamento del sistema al variare dei parametri. I risultati ottenuti consentono di individuare configurazioni che realizzano un compromesso empiricamente efficace tra sicurezza e prestazioni, fornendo indicazioni pratiche per l'impiego del framework in applicazioni crittografiche post-quantistiche.

Infine, sulla base delle proprietà garantite da ABDLOP, vengono presentate le principali costruzioni di prove non interattive a conoscenza zero basate su reticoli e la loro applicazione alla realizzazione di sistemi di credenziali anonime.

Indice

Elenco delle figure 5

1 Introduzione 7

2 Concetti preliminari 9
- 2.1 Reticoli . . . 9
  - 2.1.1 Definizioni fondamentali . . . 9
  - 2.1.2 Basi e riduzione . . . 10
  - 2.1.3 Problemi classici su reticoli . . . 10
  - 2.1.4 Problemi crittografici su reticoli . . . 11
- 2.2 Distribuzioni di probabilità . . . 14
  - 2.2.1 Rejection Sampling . . . 15
- 2.3 Zero-Knowledge Proofs . . . 17
  - 2.3.1 Interactive Proof (IP) Systems . . . 17
- 2.4 Σ-protocolli e Schemi di Identificazione . . . 20
- 2.5 Firme digitali . . . 24
- 2.6 Trasformazione di Fiat-Shamir . . . 25
- 2.7 Distanza statistica . . . 26

3 Firme digitali basate sui reticoli 27
- 3.1 Introduzione . . . 27
- 3.2 Schema di firma di Lyubashevsky . . . 27
  - 3.2.1 Algoritmo Gen . . . 28
  - 3.2.2 Algoritmo Sign . . . 28
  - 3.2.3 Algoritmo Verify . . . 28
- 3.3 Rejection Sampling e distribuzione gaussiana discreta . . . 29
- 3.4 Analisi dei parametri di sicurezza . . . 30
  - 3.4.1 Equivalenza tra LWE e SIS a bassa densità . . . 31
  - 3.4.2 Algoritmo Gen . . . 32
  - 3.4.3 Algoritmo Sign . . . 32
  - 3.4.4 Algoritmo Verify . . . 33

4 ABDLOP 37
4.1 Il protocollo di identificazione di Lyubashevsky . . . 37
4.2 Analisi come Σ-protocollo . . . 38
4.2.1 Completezza . . . 38
4.2.2 Honest-Verifier Zero-Knowledge . . . 38
4.2.3 Limiti della special soundness . . . 39
4.2.4 Conseguenze . . . 40
4.3 Definizione dello schema . . . 40
4.3.1 Ajtai commitment scheme . . . 40
4.3.2 BDLOP commitment scheme . . . 41
4.3.3 ABDLOP commitment scheme . . . 41
4.4 Proprietà di sicurezza . . . 44
4.4.1 Correttezza . . . 44
4.4.2 Hiding . . . 44
4.4.3 Binding . . . 45
4.4.4 Honest-Verifier Zero-Knowledge . . . 46
4.4.5 Knowledge Extractability . . . 47
4.5 ABDLOP come Proof of Knowledge . . . 48
4.6 Analisi dei parametri . . . 50
4.6.1 Parametri principali . . . 50
4.6.2 Il modulo q . . . 50
4.6.3 B_1, B_2 . . . 51
4.6.4 σ_1, σ_2 in relazione con il challenge space $\mathcal{C}$ ed η . . . 51
4.6.5 Vincoli di sicurezza . . . 52
4.6.6 Compromesso tra sicurezza ed efficienza . . . 52
4.7 Implementazione del protocollo ABDLOP . . . 53
4.7.1 Parametrizzazione . . . 54
4.7.2 Prima implementazione simmetrica: $m_1{=}m_2$ . . . 57
4.7.3 Seconda implementazione asimmetrica: $m_1 \neq m_2$ . . . 62
4.7.4 Conclusioni e scelta proposta dei parametri . . . 65

5 Dalle prove non interattive alle credenziali anonime 67
5.1 Proprietà richieste ai sistemi di credenziali anonime . . . 68
5.2 Il problema ISIS_{bin} . . . 68
5.3 Approximate Range Proofs . . . 70

6 Framework per Credenziali Anonime 73
6.1 Credenziali Anonime . . . 73
6.2 Il problema NTRU-ISIS_f . . . 76
6.3 Introduzione al framework . . . 79
6.4 Il sistema a credenziali anonime . . . 79

7 Conclusioni e sviluppi futuri 83

A Non Interactive Zero Knowledge Proofs 85
A.1 Non Interactive Zero Knowledge Proof $\Pi_{\text{NIZK}}^{\text{ISIS}}$ . . . 85
A.1.1 Correttezza . . . 94
A.1.2 Zero-Knowledge . . . 98
A.2 Non Interactive Zero Knowledge Proof $\Pi_{\text{NIZK}}^{\text{Com}}$ . . . 102
A.2.1 Multi-Proof Extractability e Trasformazione di Katsumata . . . 106
A.2.2 Zero-Knowledge . . . 112
A.2.3 Multi-Proof Extractability . . . 115

Elenco delle figure

4.1 Scaling del tempo del prover rispetto alla dimensione del reticolo md. . . . 59
4.2 Riduzione del tempo del prover al crescere di α, dovuta alla diminuzione del costo medio del rejection sampling. 60
4.3 Sicurezza del challenge space $\mathcal{C}$ per $(d, \tau) = (128,28)$ e $(d, \tau) = (256,22)$. . . 61

Elenco degli algoritmi

1 Rejection Sampling (Accept-Reject) . . . 15
2 Protocollo di Schnorr . . . 21
3 Simulatore HVZK per il protocollo di Schnorr . . . 22
4 Schema di identificazione canonico a 3-passi . . . 23
5 Esperimento per la sicurezza di uno schema di identificazione . . . 23
6 Esperimento di falsificazione della firma . . . 24
7 Gen . . . 28
8 Sign . . . 28
9 Verify . . . 28
10 Gen . . . 32
11 Sign . . . 32
12 Verify . . . 33
13 $\mathsf{Exp}_{pp}^{\mathsf{NTRU\text{-}ISIS}_f}(\mathcal{A})$ . . . 78
14 $\mathsf{Prove}_H^{\mathsf{ISIS}}$ . . . 93
15 $\mathsf{Verify}_H^{\mathsf{ISIS}}$ . . . 94
16 $\mathrm{LHC.Com}_i$ . . . 107
17 $\mathrm{LHC.Open}_i$ . . . 107
18 $\mathrm{LHC.Verify}_i$ . . . 108
19 $\mathsf{Prove}_H^{\mathsf{Com}}$ . . . 109
20 $\mathsf{Verify}_H^{\mathsf{Com}}$ . . . 110
21 $\mathrm{LHC.ZKSim}_i$ . . . 113
22 MultiExtract . . . 117

Capitolo 1

Introduzione

Negli ultimi anni, la crescente attenzione verso la sicurezza dei sistemi crittografici in presenza di avversari quantistici ha portato allo sviluppo di algoritmi basati su reticoli, considerati tra i più promettenti nell'ambito della crittografia post quantistica. In questo contesto si inseriscono i lavori Bootle et al. [2023] e Lyubashevsky et al. [2022].

Le credenziali anonime rappresentano uno strumento fondamentale per la salvaguardia della privacy: consentono ad un utente di dimostrare il possesso di determinate informazioni senza rivelarne il contenuto né la propria identità. Tuttavia, la progettazione di tali sistemi presenta difficoltà significative, dovute alla necessità di gestire strutture algebriche complesse e garantire allo stesso tempo proprietà di sicurezza come anonimato e resistenza ad attacchi quantistici.

Uno degli aspetti centrali affrontati nei lavori di Lyubashevsky è la costruzione di schemi di *zero-knowledge* efficienti per relazioni definite su reticoli. In particolare, si ha la necessità di utilizzare schemi di commitment che non solo nascondano il valore impegnato, ma che permettano anche di dimostrare proprietà strutturali su di esso. Ad esempio, nel caso in cui si voglia provare che per un segreto s valga $||s|| \leq \beta$.

Per affrontare tale problematica, Lyubashevsky introduce un nuovo schema di commitment, noto come *ABDLOP commitment scheme*. Questo schema costituisce il nocciolo degli schemi proposti in Bootle et al. [2023] e Lyubashevsky et al. [2022] e viene combinato con una serie di tecniche avanzate, tra cui prodotti scalari, dimostrazioni lineari e quadratiche su $\mathbb{Z}_q$. L'interazione di questi strumenti consente di ottenere un sistema efficiente per la costruzione di protocolli zero-knowledge su reticoli.

Una parte rilevante del lavoro sarà dunque dedicata allo studio delle tecniche di campionamento su reticoli, in particolare all'uso di distribuzioni gaussiane discrete e di algoritmi di campionamento di pre-immagini, che giocano un ruolo essenziale nella costruzione delle prove.

Dopo una rapida selezione di concetti preliminari utili alla comprensione della tesi, nel capitolo 3 verrà realizzata una breve disamina preliminare degli schemi di firma di Lyubashevsky basati su reticoli. Saranno poi ripresi i concetti già introdotti da Previtali [2025] (Appendix B) in cui si mostra che tali schemi non soddisfano pienamente la proprietà di special soundness e si mette in evidenza la necessità di schemi di commitment più avanzati.

In questo contesto si inserisce l'obbiettivo principale di questa tesi: la definizione e l'analisi delle proprietà dello schema di commitment ABDLOP, trattato nel capitolo 4. Dal punto di vista teorico verrà anche analizzata una parametrizzazione del protocollo, fornendo un criterio matematicamente rigoroso per la selezione dei parametri.

Accanto alla trattazione teorica, il capitolo 4 include anche una componente sperimentale. In particolare, verrà sviluppata un'implementazione in Python dello schema ABDLOP presentato, con l'obiettivo di valutare il comportamento dei parametri del sistema. Tale implementazione consentirà di osservare come la variazione di parametri influenzi il tempo di esecuzione e l'occupazione della memoria.

Nel capitolo 5 la tesi prosegue con una breve introduzione alle prove non interattive $\Pi^{\text{ISIS}}_{\text{NIZK}}$ e $\Pi^{\text{Com}}_{\text{NIZK}}$, nelle quali il protocollo ABDLOP viene reso non interattivo mediante la trasformazione di Fiat–Shamir. Tali prove verranno analizzate e dimostrate rigorosamente nell'appendice A, dove si verificheranno le proprietà di correttezza e zero-knowledge.

Infine, nel capitolo 6 verrà discusso come tali costruzioni possano essere utilizzate per la realizzazione di protocolli crittografici avanzati, in particolare sistemi di credenziali anonime. La tesi culmina quindi nella presentazione del framework proposto in Bootle et al. [2023].

Capitolo 2

Concetti preliminari

2.1 Reticoli

2.1.1 Definizioni fondamentali

Definizione 2.1.1 (Reticolo)**.** *Un* ***reticolo*** $\mathcal{L}$ *di rango* n *nello spazio euclideo* $\mathbb{R}^m$ *è l'insieme di tutte le combinazioni lineari con coefficienti interi di* n *vettori linearmente indipendenti* $b_1, \ldots, b_n \in \mathbb{R}^m$*:*

$$\mathcal{L} = \left\{ \sum_{i=1}^{n} z_i b_i \ : \ z_i \in \mathbb{Z} \right\}. \tag{2.1}$$

I vettori $b_1, \ldots, b_n$ *formano una* ***base*** $B = [b_1, \ldots, b_n]$ *del reticolo* $\mathcal{L}$*.*

Definizione 2.1.2 (Rango e dimensione)**.** *Il* ***rango*** *di un reticolo* $\mathcal{L}$ *è il numero* n *di vettori della sua base. La* ***dimensione*** *è lo spazio minimo* $\mathbb{R}^m$ *che contiene il reticolo.*

Definizione 2.1.3 (Reticolo duale)**.** *Sia* $\mathcal{L} \subset \mathbb{R}^m$ *un reticolo di rango* n*. Il* ***reticolo duale*** $\mathcal{L}^*$ *è definito come*

$$\mathcal{L}^* = \{ \mathbf{y} \in \mathbb{R}^m \ : \ \langle \mathbf{y}, \mathbf{x} \rangle \in \mathbb{Z} \text{ per ogni } \mathbf{x} \in \mathcal{L} \}, \tag{2.2}$$

dove $\langle \mathbf{y}, \mathbf{x} \rangle$ *denota il prodotto scalare euclideo tra* $\mathbf{y}$ *e* $\mathbf{x}$*.*

Se $B = [b_1, \ldots, b_n]$ *è una base di* $\mathcal{L}$*, allora* $\mathcal{L}^*$ *ha per base*

$$B^* = (B^\top)^{-1} \in \mathbb{R}^{n \times m}, \tag{2.3}$$

ossia i vettori $b_1^*, \ldots, b_n^*$ *soddisfano* $\langle b_i^*, b_j \rangle = \delta_{ij}$*, dove* δ_{ij} *è la delta di Kronecker.*

Definizione 2.1.4 (Volume di un reticolo)**.** *Il* ***volume*** *di un reticolo* $\mathcal{L}$ *con base* $B = [b_1, \ldots, b_n]$ *è:*

$$vol(\mathcal{L}) = |\det(B)|. \tag{2.4}$$

Esso misura la densità del reticolo nello spazio.

2.1.2 Basi e riduzione

Teorema 2.1.1 (Ortogonalizzazione di Gram–Schmidt)**.** *Sia $\{v_1, v_2, \ldots, v_n\}$ un insieme di vettori linearmente indipendenti in uno spazio vettoriale con prodotto scalare $\langle \cdot, \cdot \rangle$. Allora esiste un insieme di vettori ortonormali $\{u_1, u_2, \ldots, u_n\}$ generato dai vettori v_i mediante il procedimento di Gram–Schmidt, definito ricorsivamente da:*

$$\begin{aligned} u_1 &= \frac{v_1}{\|v_1\|}, \\ u_k &= \frac{v_k - \sum_{j=1}^{k-1} \langle v_k, u_j \rangle u_j}{\left\| v_k - \sum_{j=1}^{k-1} \langle v_k, u_j \rangle u_j \right\|}, \quad k = 2, \ldots, n, \end{aligned} \tag{2.5}$$

dove $\| \cdot \|$ denota la norma indotta dal prodotto scalare. I vettori $\{u_1, \ldots, u_n\}$ soddisfano le proprietà:

$$\langle u_i, u_j \rangle = \begin{cases} 1 & \text{se } i = j, \\ 0 & \text{se } i \neq j. \end{cases} \tag{2.6}$$

Definizione 2.1.5 (Base ridotta di un reticolo)**.** *Sia $L \subset \mathbb{R}^n$ un reticolo generato da una base $\mathbf{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$. Una base $\mathbf{B}$ è detta* ridotta *(in senso LLL) se soddisfa le seguenti condizioni:*

1. ***Condizione di Gram-Schmidt:*** *siano $\mathbf{b}_1^*, \ldots, \mathbf{b}_n^*$ i vettori ottenuti mediante ortogonalizzazione di Gram-Schmidt. Allora per ogni $i = 2, \ldots, n$,*

   $$|\mu_{i,j}| \leq \frac{1}{2} \quad \text{per ogni } j < i,$$

   dove $\mu_{i,j}$ sono i coefficienti della proiezione di $\mathbf{b}_i$ su $\mathbf{b}_j^*$.

2. ***Condizione di Lovász:*** *esiste un parametro $0.25 < \delta < 1$ tale che per ogni $i = 1, \ldots, n-1$*

   $$\delta \|\mathbf{b}_i^*\|^2 \leq \|\mathbf{b}_{i+1}^* + \mu_{i+1,i} \mathbf{b}_i^*\|^2.$$

Nota 1. *Intuitivamente, una base ridotta è composta da vettori relativamente corti e quasi ortogonali, rendendo più semplice approssimare il vettore più corto del reticolo e risolvere problemi computazionali come SVP e CVP.*

Teorema 2.1.2 (Relazione tra basi)**.** *Date due basi B e B' di $\mathcal{L}$, esiste una matrice invertibile $U \in \mathbb{Z}^{n \times n}$ tale che*

$$B' = B \cdot U. \tag{2.7}$$

2.1.3 Problemi classici su reticoli

Shortest Vector Problem (SVP)

Definizione 2.1.6 (SVP – Shortest Vector Problem)**.** *Dato un reticolo $\mathcal{L} \subset \mathbb{R}^m$, trovare un vettore non nullo $v \in \mathcal{L}$ di lunghezza minima:*

$$\|v\| = \lambda_1(\mathcal{L}) = \min\{\|x\| \ : \ x \in \mathcal{L} \setminus \{0\}\}. \tag{2.8}$$

Definizione 2.1.7 (SVP Approssimato)**.** *Per un fattore $\gamma \geq 1$, il problema γ-SVP consiste nel trovare $v \in \mathcal{L} \setminus \{0\}$ tale che*

$$\|v\| \leq \gamma \cdot \lambda_1(\mathcal{L}). \tag{2.9}$$

Nota 2. *SVP è NP-hard per fattori approssimati $\gamma = 1 + 1/poly(n)$. Questo costituisce la base di sicurezza dei protocolli di credenziali anonime su reticoli.*

Closest Vector Problem (CVP)

Definizione 2.1.8 (CVP – Closest Vector Problem)**.** *Dato un reticolo $\mathcal{L} \subset \mathbb{R}^m$ e un vettore target $t \in \mathbb{R}^m$, trovare un vettore $v \in \mathcal{L}$ tale che*

$$\|v - t\| = \min_{x \in \mathcal{L}} \|x - t\|. \tag{2.10}$$

Definizione 2.1.9 (CVP Approssimato)**.** *Per un fattore $\gamma \geq 1$, il problema γ-CVP consiste nel trovare $v \in \mathcal{L}$ tale che*

$$\|v - t\| \leq \gamma \cdot \min_{x \in \mathcal{L}} \|x - t\|. \tag{2.11}$$

Nota 3. *CVP è NP-hard anche per approssimazioni polinomiali.*

2.1.4 Problemi crittografici su reticoli

Oltre ai problemi classici su reticoli, come il **Shortest Vector Problem (SVP)** e il **Closest Vector Problem (CVP)**, la crittografia moderna basata su reticoli introduce varianti computazionali fondamentali per la costruzione di firme digitali, cifrature e Zero Knowledge Proofs. In questa sezione si presentano due problemi centrali: **SIS** e **LWE**.

Short Integer Solution (SIS)

Il problema *Short Integer Solution (SIS)* può essere visto come una generalizzazione dei problemi lineari su reticoli, modulata da un modulo q e limitata da una norma β.

Definizione 2.1.10 (SIS)**.** *Siano $n, m \in \mathbb{N}$, un modulo $q \in \mathbb{Z}$ e un parametro di norma $\beta > 0$. Dato un insieme di vettori $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, il problema SIS consiste nel trovare un vettore non nullo $\mathbf{x} \in \mathbb{Z}^m$ tale che*

$$\mathbf{Ax} = \mathbf{0} \pmod{q} \quad e \quad \|\mathbf{x}\| \leq \beta. \tag{2.12}$$

Definizione 2.1.11 (ℓ_2-$\mathrm{SIS}_{q,n,m,\beta}$)**.** *Sia $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$ una matrice scelta uniformemente. Il problema ℓ_2-$SIS_{q,n,m,\beta}$ consiste nel trovare un vettore non nullo $\mathbf{v} \in \mathbb{Z}^m \setminus \{\mathbf{0}\}$ tale che:*

$$\mathbf{Av} = \mathbf{0} \mod q \tag{2.13}$$

e

$$\|\mathbf{v}\|_{\ell_2} \leq \beta. \tag{2.14}$$

Per garantire l'esistenza di una soluzione, è necessario che:

$$\beta \geq \sqrt{m}\, q^{\frac{n}{m}}. \tag{2.15}$$

Definizione 2.1.12 (Distribuzione $\mathrm{SIS}_{q,n,m,d}$)**.** *Sia* $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$ *e sia* $\mathbf{s} \xleftarrow{\$} \{-d,\dots,0,\dots,d\}^m$*. La distribuzione* $SIS_{q,n,m,d}$ *restituisce la coppia:*

$$(\mathbf{A}, \mathbf{As}). \tag{2.16}$$

Definizione 2.1.13 (Problema di ricerca $\mathrm{SIS}_{q,n,m,d}$)**.** *Dato un campione* $(\mathbf{A}, \mathbf{t})$ *dalla distribuzione* $SIS_{q,n,m,d}$*, trovare un vettore* $\mathbf{s} \in \{-d,\dots,0,\dots,d\}^m$ *tale che:*

$$\mathbf{As} = \mathbf{t}. \tag{2.17}$$

Definizione 2.1.14 (Problema decisionale $\mathrm{SIS}_{q,n,m,d}$)**.** *Dato un campione* $(\mathbf{A}, \mathbf{t})$*, decidere con vantaggio non trascurabile se:*

$$(\mathbf{A}, \mathbf{t}) = (\mathbf{A}, \mathbf{As}) \tag{2.18}$$

per qualche $\mathbf{s} \xleftarrow{\$} \{-d,\dots,0,\dots,d\}^m$*, oppure se* $(\mathbf{A}, \mathbf{t})$ *è campionato uniformemente da:*

$$\mathbb{Z}_q^{n\times m} \times \mathbb{Z}_q^n. \tag{2.19}$$

Consideriamo ora un anello $\mathcal{R}$ (ad esempio $\mathbb{Z}[X]/(X^n+1)$) e sia $\mathcal{R}_q$ la sua riduzione modulo q.

Definizione 2.1.15 (Module-$\mathrm{SIS}_{\kappa,m,B}$)**.** *Siano dati i parametri* κ*,* $m > 0$ *e* $0 < B < q$*. Sia*

$$\mathbf{A} \xleftarrow{\$} \mathcal{R}_q^{\kappa\times m}$$

una matrice scelta uniformemente da $\mathcal{R}_q^{\kappa\times m}$*. Il problema* Module-SIS *con parametri* (κ, m, B) *consiste nel trovare un vettore*

$$\mathbf{z} \in \mathcal{R}_q^m \setminus \{0\} \quad \textit{tale che} \quad \mathbf{Az} = 0 \textit{ in } \mathcal{R}_q \textit{ e } \|\mathbf{z}\|_\infty \leq B.$$

Un algoritmo $\mathcal{A}$ *ha* advantage ϵ *nel risolvere* $MSIS_{\kappa,m,B}$ *se*

$$\mathbb{P}\big[0 < \|\mathbf{z}\|_\infty \leq B \wedge \mathbf{Az} = 0 \mid \mathbf{A} \xleftarrow{\$} \mathcal{R}_q^{\kappa\times m}, \mathbf{z} \leftarrow \mathcal{A}(\mathbf{A})\big] \geq \epsilon.$$

Una generalizzazione del problema SIS è il seguente:

Definizione 2.1.16 ($\mathrm{Game}^{\mathrm{ISIS}_f}$)**.** *Sia data:*

- *una matrice* $\mathbf{A} \in \mathbb{Z}_p^{n\times m}$ *scelta da una distribuzione nota,*
- *una funzione* $f : [N] \to \mathbb{Z}_p^n$*,*
- *accesso a un oracolo che, per un input casuale* $x \in [N]$*, restituisce una coppia* $(x, \mathbf{s})$ *con*

$$\mathbf{s} \in \mathbb{Z}^m, \quad \|\mathbf{s}\| \leq \beta, \quad e \quad \mathbf{As} = f(x) \mod p. \tag{2.20}$$

Il giocatore vince il gioco producendo una nuova coppia

$$(x', \mathbf{s}') \in [N] \times \mathbb{Z}^m \tag{2.21}$$

tale che

$$\|\mathbf{s}'\| \leq \beta' \quad e \quad \mathbf{A}\mathbf{s}' = f(x') \mod p, \tag{2.22}$$

dove $(x', \mathbf{s}')$ *non è stata mai restituita precedentemente dall'oracolo.*

Si può infatti ottenere il problema SIS a partire da ISIS_f se si sceglie $f(x) = \mathbf{0}$ per tutti $x \in [N]$ e si ignora l'oracolo.

Definizione 2.1.17 (Problema ISIS_f)**.** *Sia*

$$\mathsf{pp} := (q, d, n, m, N, k, B)$$

una tupla di parametri che dipendono dal parametro di sicurezza λ*. Sia inoltre*

$$f : [N] \to \mathcal{R}_q^n$$

una funzione efficientemente calcolabile.

L'assunzione di difficoltà di ISIS *relativa a* f *è definita tramite il* $Game_f^{ISIS}$*. Per un avversario* $\mathcal{A}$*, definiamo il vantaggio come:*

$$\text{Adv}_{\mathsf{pp}}^{\text{ISIS}_f}(\mathcal{A}) := \mathbb{P}[Game_{\mathsf{pp}}^{ISIS_f}(\mathcal{A}) \to 1].$$

L'assunzione di difficoltà ISIS *per i parametri* pp *e la funzione* f *afferma che, per ogni avversario PPT* $\mathcal{A}$*, il vantaggio* $\text{Adv}_{f,\mathsf{pp}}^{\text{ISIS}}(\mathcal{A})$ *è trascurabile.*

Learning With Errors (LWE)

Il problema *Learning With Errors (LWE)* introduce un piccolo errore nei prodotti scalari, rendendo la soluzione del vettore segreto difficile.

Definizione 2.1.18 (LWE)**.** *Siano* $n, m \in \mathbb{N}$*, un modulo* $q \in \mathbb{Z}$ *e una distribuzione di errore* χ *su* $\mathbb{Z}_q$*. Sia* $\mathbf{s} \in \mathbb{Z}_q^n$ *un vettore segreto scelto uniformemente. Dati campioni* $(\mathbf{a}_i, b_i) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ *tali che*

$$b_i = \langle \mathbf{a}_i, \mathbf{s} \rangle + e_i \pmod q, \quad e_i \sim \chi, \tag{2.23}$$

il problema LWE consiste nel determinare $\mathbf{s}$ *a partire dai campioni.*

Tale problema può anche essere definito utilizzando il concetto di distribuzione $LWE_{q,n}$.

Definizione 2.1.19 (Distribuzione $LWE_{q,n}$)**.** *Sia* $\mathbf{s} \leftarrow \mathbb{Z}_q^n$ *un vettore scelto uniformemente e sia* $e \leftarrow \mathbb{Z}_q$ *un errore piccolo.*

La distribuzione LWE è definita come:

$$A_{\mathbf{s}} = \left\{ (\mathbf{a}, b) \in \mathbb{Z}_q^n \times \mathbb{Z}_q \;\middle|\; \mathbf{a} \leftarrow \mathbb{Z}_q^n,\ b = \langle s, \mathbf{a} \rangle + e \mod q \right\}. \tag{2.24}$$

Definizione 2.1.20 (Problema di ricerca $LWE_{q,n,m}$)**.** *Trova* $\mathbf{s} \in \mathbb{Z}_q^n$ *date* m *coppie* $(\mathbf{a}, b) \in A_{\mathbf{s}}$.

Definizione 2.1.21 (Problema decisionale $LWE_{q,n,m}$)**.** *Date* m *coppie* $(\mathbf{a}, b)$ *distingui se provengono da una distribuzione* $LWE_{q,n}$ $A_{\mathbf{s}}$ *oppure da una distribuzione uniforme* $U(\mathbb{Z}_{\text{ll}}^{\ltimes} \times \mathbb{Z}_{\text{ll}})$.

Così come fatto con il problema SIS, definiamo ora il problema $\text{MLWE}_{n,m,\chi,q}$.

Definizione 2.1.22 (Module-$\text{LWE}_{n,m,\chi,q}$)**.** *Siano* $m \geq n > 0$, $q \in \mathbb{N}$ *e* χ *una distribuzione sull'anello* $\mathcal{R}_q$. *Il problema* Module-LWE *con parametri* (n, m, χ, q) *consiste nel distinguere tra le seguenti due distribuzioni:*

- $(\mathbf{A}, \mathbf{As})$, *dove* $\mathbf{A} \leftarrow \mathcal{R}_q^{n \times m}$ *uniforme e* $\mathbf{s} \leftarrow \chi^m$*;*
- $(\mathbf{A}, \mathbf{b})$, *dove* $\mathbf{A} \leftarrow \mathcal{R}_q^{n \times m}$ *e* $\mathbf{b} \leftarrow \mathcal{R}_q^n$ *uniforme.*

Per un avversario PPT $\mathcal{A}$*, definiamo il vantaggio:*

$$\text{Adv}_{n,m,\chi,q}^{\text{MLWE}}(\mathcal{A}) := \left| \mathbb{P}\left[\mathcal{A}(\mathbf{A}, \mathbf{As}) = 1\right] - \mathbb{P}\left[\mathcal{A}(\mathbf{A}, \mathbf{b}) = 1\right] \right|.$$

Il problema Module-LWE è detto difficile se tale vantaggio è trascurabile.

Relazione con SVP e CVP

SIS e LWE possono essere considerati problemi crittografici derivati dai classici problemi su reticoli:

- SIS è collegato al problema duale di CVP: trovare una soluzione corta a un sistema lineare modulo q.
- LWE introduce errori nei prodotti scalari, rendendo il problema equivalente a risolvere CVP approssimato in un reticolo ad alta dimensione.
- Entrambi i problemi costituiscono la base dei protocolli su reticoli.

2.2 Distribuzioni di probabilità

Per costruire prove e firme sicure su reticoli, è necessario campionare vettori in modo controllato.

Definizione 2.2.1. *La distribuzione Normale continua su* $\mathbb{R}^m$ *centrata in* $\mathbf{c}$ *avente deviazione standard* $\sigma > 0$ *è così definita:*

$$\rho_{\mathbf{c},\sigma}^m(\mathbf{x}) = \Big(\frac{1}{\sqrt{2\pi\sigma^2}}\Big)^m e^{-\frac{||\mathbf{x}-\mathbf{c}||^2}{2\sigma^2}}. \tag{2.25}$$

Definizione 2.2.2. *La distribuzione Normale discreta su $\mathbb{R}^m$ centrata in* $\mathbf{c}$ *avente deviazione standard $\sigma > 0$ è così definita:*

$$D^m_{\mathbf{c},\sigma}(\mathbf{x}) = \frac{\rho^m_{\mathbf{c},\sigma}(\mathbf{x})}{\sum_{\mathbf{z}\in\mathbb{Z}^m} \rho^m_\sigma(\mathbf{x})} = \frac{\rho^m_{\mathbf{c},\sigma}(\mathbf{x})}{\rho^m_\sigma(\mathbb{Z}^m)} \tag{2.26}$$

Nota 4. *Nel framework di Lyubashevsky, il campionamento da distribuzioni discrete gaussiane permette di:*

- *generare vettori brevi per firme e prove zero-knowledge senza rivelare informazioni segrete;*
- *garantire la sicurezza basata sulla difficoltà di problemi hard sui reticoli (LWE, SVP);*
- *ottenere distribuzioni statisticamente indistinguibili da quelle ideali, fondamentale per il concetto di zero-knowledge.*

2.2.1 Rejection Sampling

Il *rejection sampling* è una tecnica probabilistica utilizzata per campionare valori $x_1, ..., x_n$ da una distribuzione desiderata f (target distribution) a partire da una distribuzione per cui è facile ed efficiente ottenere campioni g (proposal distribution).

Rejection Sampling: Definizione e dimostrazione

Algoritmo 1: Rejection Sampling (Accept-Reject)

Input: Distribuzione proposal g, distribuzione target f, costante di normalizzazione M per cui valga $M \geq \frac{f(x)}{g(x)} \quad \forall x \in [0,1]$
Output: Campione $x \sim f$
while *true* **do**
  Genera candidato $x \sim g$;
  Calcola probabilità di accettazione $p = \frac{f(x)}{M \cdot g(x)}$;
  Genera $u \sim$ Uniforme(0,1);
  if $u \leq p$ **then**
    return x `// Accetta il candidato`
  end
  `// Altrimenti rifiuta e ripeti`
end

Teorema 2.2.1. *Siano f e g due distribuzioni di probabilità su uno spazio discreto $\mathcal{X}$, tali che $f(x) > 0, g(x) > 0$. Supponiamo inoltre che esista una costante $M \geq 1$ tale che*

$$f(x) \leq M \cdot g(x), \quad \forall x \in \mathcal{X}.$$

Considerando l'algoritmo 1 si può affermare che la distribuzione della variabile casuale X condizionata all'evento di accettazione è esattamente f, cioè

$$\mathbb{P}(X = x \mid \textit{accettato}) = f(x), \quad \forall x \in \mathcal{X}.$$

.

Dimostrazione. Sia A l'evento di accettazione. Per ogni $x \in \mathcal{X}$, si ha:

$$\mathbb{P}(X = x \cap A) = \mathbb{P}(X = x) \cdot \mathbb{P}\left(U \leq \frac{f(x)}{M \cdot g(x)} \,\middle|\, X = x\right).$$

Poiché $X \sim g$ e U è indipendente da X, segue che:

$$\mathbb{P}(X = x \cap A) = g(x) \cdot \frac{f(x)}{M \cdot g(x)} = \frac{f(x)}{M}.$$

Calcoliamo ora la probabilità totale di accettazione:

$$\mathbb{P}(A) = \sum_{x \in \mathcal{X}} \mathbb{P}(X = x \cap A) = \sum_{x \in \mathcal{X}} \frac{f(x)}{M} = \frac{1}{M} \sum_{x \in \mathcal{X}} f(x).$$

Poiché f è una distribuzione di probabilità, si ha $\sum_x f(x) = 1$, quindi:

$$\mathbb{P}(A) = \frac{1}{M}.$$

Applicando il teorema di Bayes:

$$\mathbb{P}(X = x \mid A) = \frac{\mathbb{P}(X = x \cap A)}{\mathbb{P}(A)} = \frac{\frac{f(x)}{M}}{\frac{1}{M}} = f(x).$$

□

Rejection Sampling nei protocolli basati sui reticoli

Nei protocolli di identificazione e nelle costruzioni zero-knowledge basate su reticoli proposte da Lyubashevsky, un problema cruciale è garantire che le risposte del prover siano statisticamente indipendenti dal segreto s. Infatti, nelle tipiche risposte del tipo

$$z = y + c \cdot s,$$

dove y è un vettore scelto casualmente secondo una distribuzione discreta (ad esempio gaussiana) e c è la challenge, la distribuzione di z dipende dal segreto s. Questa dipendenza può compromettere le proprietà di *zero-knowledge* del protocollo, poiché un verificatore potrebbe ottenere informazioni sul segreto analizzando la distribuzione delle risposte.

Per risolvere questo problema, Lyubashevsky introduce nel framework ABDLOP l'uso del **rejection sampling**. Il prover accetta la risposta z solo con una certa probabilità calcolata in modo da rendere la distribuzione finale di z praticamente indipendente da s. In questo modo, la risposta accettata può essere simulata senza conoscere il segreto, garantendo la proprietà fondamentale di zero-knowledge.

Il rejection sampling è quindi strettamente integrato nello *ABDLOP commitment scheme*, che costituisce il cuore del framework. Ogni volta che si desidera dimostrare proprietà sul segreto s senza rivelarlo, il rejection sampling permette di "pulire" la distribuzione delle risposte, correggendo la correlazione residua introdotta dall'addizione del termine $c \cdot s$.

2.3 Zero-Knowledge Proofs

Definizione 2.3.1 (Linguaggio)**.** *Sia Σ un alfabeto finito. Un* linguaggio *L su Σ è un sottoinsieme di Σ^*, cioè:*

$$L \subseteq \Sigma^* \tag{2.27}$$

dove Σ^* denota l'insieme di tutte le stringhe finite su Σ.

Definizione 2.3.2 (Classe NP)**.** *Un linguaggio L appartiene alla classe* NP *se esiste una macchina di Turing deterministica V e un polinomio $p(\cdot)$ tali che:*

$$x \in L \iff \exists w \in \{0,1\}^{\leq p(|x|)} \text{ tale che } V(x,w) = 1 \tag{2.28}$$

Definizione 2.3.3 (Relazione associata a un linguaggio in NP)**.** *Sia $L \in$ NP. Esiste una relazione binaria $R_L \subseteq \{0,1\}^* \times \{0,1\}^*$ tale che:*

$$x \in L \iff \exists w \in \{0,1\}^* \text{ tale che } (x,w) \in R_L \tag{2.29}$$

dove w è detto witness *(o certificato) per x.*

2.3.1 Interactive Proof (IP) Systems

Definizione 2.3.4 (Interazione tra funzioni)**.** *Siano $f, g : \{0,1\}^* \to \{0,1\}^*$. Una interazione di k round tra f e g su input $x \in \{0,1\}^*$ è una sequenza di stringhe $a_1, \ldots, a_k \in \{0,1\}^*$ tale che:*

$$a_i := \begin{cases} f(x, a_1, \ldots, a_{i-1}) & \text{se } i \text{ è dispari} \\ g(x, a_1, \ldots, a_{i-1}) & \text{se } i \text{ è pari} \end{cases} \tag{2.30}$$

L'output finale di f è denotato da:

$$out_V\langle f, g\rangle(x) \tag{2.31}$$

Definizione 2.3.5 (Classe IP)**.** *Un linguaggio L appartiene alla classe di complessità* IP *se esiste un algoritmo PPT V (verificatore) che può avere un'interazione di k round con una funzione $P : \{0,1\}^* \to \{0,1\}^*$ (prover) tale che valgono le seguenti proprietà:*

- ***Completezza.*** *Se $x \in L$, allora esiste un prover P tale che:*

$$\mathbb{P}\left[out_V\langle V, P\rangle(x) = 1\right] \geq \frac{2}{3} \tag{2.32}$$

- ***Soundness.*** *Se $x \notin L$, allora per ogni prover P vale:*

$$\mathbb{P}\left[out_V\langle V, P\rangle(x) = 1\right] \leq \frac{1}{3} \tag{2.33}$$

Una coppia (V, P) che soddisfa completezza e soundness è detta sistema di prova interattivo *per L.*

I sistemi di prova interattivi forniscono un modello valido per verificare la correttezza di affermazioni tramite interazione tra un prover e un verificatore. In tale modello, il verificatore è in grado di decidere se una stringa x appartiene a un linguaggio L con alta probabilità, sulla base dell'interazione con un prover computazionalmente illimitato. Tuttavia non c'è alcuna garanzia che il verificatore non apprenda informazioni aggiuntive oltre alla validità dell'affermazione $x \in L$. Il prover potrebbe, infatti, rivelare informazioni sul witness w associato a x. Per ovviare a questo problema, si introduce la nozione di *zero-knowledge proof*, che rafforza il concetto di sistema di prova interattivo imponendo un ulteriore requisito di sicurezza: il verificatore non deve apprendere alcuna informazione addizionale oltre al fatto che $x \in L$.

Definizione 2.3.6 (Zero-Knowledge Proof)**.** *Sia L un linguaggio in* NP*. Una coppia (P, V) di algoritmi interattivi probabilistici tempo-polinomiale (PPT) è una* perfect (statistical, computational) zero-knowledge proof *per L se valgono le seguenti proprietà.*

- ***Completezza.*** *Per ogni $x \in L$ e per ogni witness w tale che $(x, w) \in R_L$, vale:*

$$\mathbb{P}\left[out_V\langle P(x,w), V(x)\rangle = 1\right] \geq \frac{2}{3} \tag{2.34}$$

- ***Soundness.*** *Per ogni $x \notin L$ e per ogni algoritmo probabilistico (anche non limitato) $\tilde{P}$:*

$$\mathbb{P}\left[out_V\langle \tilde{P}(x), V(x)\rangle = 1\right] \leq \frac{1}{3} \tag{2.35}$$

- ***Zero-Knowledge.*** *Per ogni algoritmo PPT V^* esiste un algoritmo PPT atteso S (simulatore) tale che, per ogni $x \in L$ e witness w:*

$$out_{V^*}\langle P(x,w), V^*(x)\rangle \approx S(x) \tag{2.36}$$

  dove $\approx$ denota uguaglianza perfetta (oppure indistinguibilità statistica o computazionale).

Le proprietà di completezza e soundness sono analoghe a quelle dei sistemi IP. La proprietà di zero-knowledge garantisce che il verificatore non apprenda alcuna informazione addizionale oltre alla validità dell'affermazione.

Definizione 2.3.7 (Honest-Verifier Zero-Knowledge)**.** *Un protocollo soddisfa la proprietà di* Honest-Verifier Zero-Knowledge *se esiste un simulatore PPT atteso S tale che:*

$$out_V\langle P(x,w), V(x)\rangle \approx S(x) \tag{2.37}$$

—

Definizione 2.3.8 (Proof of Knowledge)**.** *Sia $L \in$* NP*. Una coppia (P, V) è una* proof of knowledge *per L se vale la seguente proprietà.*

- ***Knowledge Soundness.*** *Esiste una costante $k > 0$ e un algoritmo PPT atteso E (estrattore) tale che per ogni algoritmo interattivo $\tilde{P}$ e ogni $x \in \{0,1\}$:*

  se

  $$p := \mathbb{P}\left[out_V\langle \tilde{P}, V\rangle = 1\right] > k \tag{2.38}$$

  allora E, dato accesso a $\tilde{P}(x)$, restituisce un witness w per $x \in L$ in un numero di passi limitato da:

  $$\frac{1}{p-k} \cdot \mathrm{poly}(|x|) \tag{2.39}$$

Definizione 2.3.9 (Zero-Knowledge Proof of Knowledge)**.** *Una coppia (P, V) è una* zero-knowledge proof of knowledge (ZKPoK) *se soddisfa:*

- *Completezza*
- *Knowledge soundness*
- *Zero-knowledge*

ovvero combina proprietà di Proof of Knowledge e Zero-knowledge.

Consideriamo relazioni binarie

$$R \subseteq \{0,1\}^* \times \{0,1\}^*.$$

Definiamo il linguaggio associato a R come:

$$L_R := \{x \in \{0,1\}^* \mid \exists w \text{ tale che } (x, w) \in R\}.$$

Definizione 2.3.10 (NIZK)**.** *Un sistema di prova non interattiva zero-knowledge Π_{NIZK} per una relazione R è costituito da tre algoritmi PPT:*

$$(\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify}),$$

definiti come segue:

- $\mathsf{Setup}(1^\lambda) \to \mathsf{crs}$*: algoritmo di inizializzazione che, dato il parametro di sicurezza λ, genera una* common reference string

  $$\mathsf{crs} \in \{0,1\}^{\ell(\lambda)};$$

- $\mathsf{Prove}^H(\mathsf{crs}, x, w) \to \pi$ *oppure* $\perp$*: algoritmo del prover che, dato* crs*, uno statement x e un witness w, produce una prova π oppure restituisce il simbolo di abort $\perp$;*
- $\mathsf{Verify}^H(\mathsf{crs}, x, \pi) \to \{0,1\}$*: algoritmo del verifier che, dato* crs*, uno statement x e una prova π, restituisce un bit $b \in \{0,1\}$, dove $b = 1$ indica accettazione e $b = 0$ rifiuto.*

Definizione 2.3.11 (Correttezza)**.** *Un sistema di prove Π_{NIZK} è corretto se, per ogni $\mathsf{crs} \in \{0,1\}^\ell$ e per ogni $(x, w) \in R$, valgono le seguenti condizioni:*

- *la probabilità che l'algoritmo* $\mathsf{Prove}^H(\mathsf{crs}, x, w)$ *restituisca* $\perp$ *è trascurabile in* λ*;*
- *inoltre, si ha:*

$$\mathbb{P}\left[\begin{array}{c} \pi \leftarrow \mathsf{Prove}^H(\mathsf{crs}, x, w) : \\ \mathsf{Verify}^H(\mathsf{crs}, x, \pi) = 1 \end{array} \;\middle|\; \pi \neq \perp \right] = 1 - \mathrm{negl}(\lambda).$$

Definizione 2.3.12 (Zero-Knowledge)**.** *Un sistema di prove* Π_{NIZK} *è* zero-knowledge *se esiste un simulatore*

$$\mathcal{S} = (\mathcal{S}_0, \mathcal{S}_1),$$

costituito da due algoritmi PPT con stato condiviso, tale che per ogni avversario PPT $\mathcal{A}$ *vale:*

$$\mathrm{Adv}^{\mathrm{ZK}}_{\Pi_{\mathrm{NIZK}}}(\mathcal{A}) := \left|\mathbb{P}[1 \leftarrow \mathcal{A}^{H,\mathsf{Prove}}(\mathsf{crs})] - \mathbb{P}[1 \leftarrow \mathcal{A}^{\mathcal{S}_0,\mathcal{S}_1}(\mathsf{crs})]\right| = \mathrm{negl}(\lambda),$$

dove:

- $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda)$*;*
- Prove *è un oracolo che, su input* (x, w)*, restituisce* $\perp$ *se* $(x, w) \notin R$*, altrimenti restituisce* $\mathsf{Prove}^H(\mathsf{crs}, x, w)$*;*
- $\mathcal{S}_1$ *è un oracolo simulatore che, su input* (crs, x)*, restituisce una prova simulata* $\mathcal{S}_1(\mathsf{crs}, x)$*.*

2.4 Σ-protocolli e Schemi di Identificazione

Definizione 2.4.1 (Σ-protocollo)**.** *Sia* $L \in \mathsf{NP}$*. Un* Σ*-protocollo è un protocollo interattivo a tre mosse tra prover e verifier con la seguente struttura:*

1. *Il prover ha come input*(x, w)
2. *Il verifier ha come input* x
3. *Il prover invia al verifier un commitment* com
4. *Il verifier invia al prover una challenge* $\mathsf{ch} \in \mathcal{C}$
5. *Il prover invia al verifier una risposta* rsp
6. *Il verifier verifica la correttezza della risposta a seconda della challenge e del commitment. Accetta o rifiuta a seconda del caso.*

Il prover consiste di due algoritmi PPT P_1, P_2*, il verifier di due algoritmi PPT* V_1, V_2*. Il transcript è* $(x, \mathsf{com}, \mathsf{ch}, \mathsf{rsp})$*.*

—

Definizione 2.4.2 (Proprietà dei Σ-protocolli)**.** *Un Σ-protocol soddisfa:*

- ***Completezza***

$$\mathbb{P}(\textit{Verifier accetta} | x \in L) = 1 \tag{2.40}$$

- ***Special Soundness***

  Esiste un estrattore PPT E tale che dati due transcript con stesso commitment e challenge diverse:

$$(x, \mathsf{com}, \mathsf{ch}_1, \mathsf{rsp}_1), \quad (x, \mathsf{com}, \mathsf{ch}_2, \mathsf{rsp}_2), \quad \mathsf{ch}_1 \neq \mathsf{ch}_2 \tag{2.41}$$

  restituisce un witness w' di $x \in L$.

- ***Honest-Verifier Zero-Knowledge*** *Esiste un simulatore PPT S tale che:*

$$S(x) \approx (x, \mathsf{com}, \mathsf{ch}, \mathsf{rsp}) \tag{2.42}$$

Il protocollo di Schnorr permette al prover di convincere il verifier di conoscere un segreto $x \in \mathbb{Z}_q$ senza rivelarlo.

Sia $G \subset \mathbb{Z}_p^*$ un sottogruppo ciclico di ordine primo q, generato da g, e sia

$$h := g^x \pmod p$$

il public key associato al segreto x del prover.

Algoritmo 2: Protocollo di Schnorr

Input: Prover: (h, x), Verifier: h
Output: Verifier accetta o rifiuta
Prover: $r \xleftarrow{\$} \mathbb{Z}_q^*$;
Prover: $y := g^r \pmod p$;
Invia y al verifier;
Verificatore: $c \xleftarrow{\$} \mathbb{Z}_q$;
Invia c al prover;
Prover: $z := r + c \cdot x \pmod q$;
Invia z al verifier;
Verifier: se $g^z \stackrel{?}{=} y \cdot h^c \pmod p$ allora accetta, altrimenti rifiuta;

Verifichiamo che il protocollo di Schnorr soddisfa le proprietà di **completezza**, **special soundness** e **honest-verifier zero-knowledge (HVZK)**.

Completezza. Dalla verifica

$$g^z \stackrel{?}{=} y \cdot h^c,$$

se il prover e il verifier seguono il protocollo onestamente, il verifier accetta sempre. Infatti:

$$g^z = g^{r+cx} = g^r \cdot g^{cx} = y \cdot h^c,$$

poiché $y = g^r$ e $h = g^x$.

Special Soundness. Date due trascrizioni valide dello stesso commitment y, con due challenge diverse $c_1 \neq c_2$:

$$(y, c_1, z_1) \quad \text{e} \quad (y, c_2, z_2),$$

possiamo estrarre un witness x' tale che $h = g^{x'}$.

Infatti, dalle verifiche abbiamo

$$g^{z_1} = yh^{c_1}, \quad g^{z_2} = yh^{c_2}.$$

Dividendo la prima equazione per la seconda:

$$g^{z_1 - z_2} = h^{c_1 - c_2}.$$

Poiché $c_1 \neq c_2$, $(c_1 - c_2)$ è invertibile in $\mathbb{Z}_q$, quindi:

$$x' = (z_1 - z_2)(c_1 - c_2)^{-1} \pmod{q}.$$

Questo significa che se il prover riesce a rispondere a due sfide diverse per lo stesso commitment, è anche in grado di calcolare il segreto. In altre parole, barare è difficile quanto calcolare il witness.

Honest-Verifier Zero-Knowledge. L'interazione non rivela alcuna informazione sul segreto x al verifier. Possiamo costruire un simulatore $S(h)$, che conosce solo h, capace di generare trascrizioni indistinguibili da quelle reali.

In un'esecuzione legittima, la trascrizione è (y, c, z), dove:

- y è un elemento casuale di $G \subset \mathbb{Z}_p^*$ di ordine q,
- $c \in \mathbb{Z}_q$ casuale,
- $z \in \mathbb{Z}_q$ soddisfa $g^z = y \cdot h^c$.

Algoritmo 3: Simulatore HVZK per il protocollo di Schnorr

Input: Public key h
Output: Trascrizione (y, c, z) indistinguibile da una reale
$z \xleftarrow{\$} \mathbb{Z}_q$;
$c \xleftarrow{\$} \mathbb{Z}_q$;
$y := g^z \cdot h^{-c} \pmod{p}$;
return (y, c, z);

Definizione 2.4.3 (Schema di Identificazione Canonico)**.** *Uno schema di identificazione a 3-passi consiste in algoritmi PPT* $\mathsf{Gen}, P_1, P_2, V$*, tali che:*

- *L'algoritmo di generazione chiavi* Gen *prende in input* 1^λ*, dove* λ *è il parametro di sicurezza, e restituisce:*

$$(pk, sk)$$

  dove pk è la chiave pubblica e sk è la chiave segreta del Prover.

- *Il Prover e il Verifier eseguono* P_1, P_2, V *come descritto nell'algoritmo sottostante, e se tutte le parti si comportano onestamente, il Verifier restituisce* 1 *(accetta).*

Algoritmo 4: Schema di identificazione canonico a 3-passi

Input: Prover: (pk, sk), Verifier: pk
Output: Verifier accetta o rifiuta
Prover: $(\mathsf{com}, P_{\text{state}}) \leftarrow P_1(pk, sk)$;
Invia com al Verifier;
Verifier: $c \xleftarrow{\$} C$;
Invia c al Prover;
Prover: $\mathsf{rsp} \leftarrow P_2(P_{\text{state}}, c)$;
Invia rsp al Verifier;
Verifier: se $V(pk, \mathsf{com}, c, \mathsf{rsp}) = 1$ allora accetta, altrimenti rifiuta;

Sia $\Pi = (\mathsf{Gen}, P_1, P_2, V)$ uno schema di identificazione. Per ogni avversario $\mathcal{A}$, definiamo l'esperimento :

Algoritmo 5: Esperimento per la sicurezza di uno schema di identificazione

Input: Avversario PPT $\mathcal{A}$, schema di identificazione $\Pi = (\mathsf{Gen}, P_1, P_2, V)$, parametro di sicurezza λ
Output: 1 se $\mathcal{A}$ ha successo, 0 altrimenti
Genera le chiavi del Prover: $(pk, sk) \leftarrow \mathsf{Gen}(1^\lambda)$;
Fornisci all'avversario $\mathcal{A}$ gli input 1^λ, pk e accesso a un oracolo Trans_{sk} che esegue una esecuzione onesta di Π e restituisce il transcript $(\mathsf{com}, \mathsf{ch}, \mathsf{rsp})$;
$\mathcal{A}$ produce un commitment com;
Genera un challenge casuale $\mathsf{ch} \xleftarrow{\$} C$ e invialo ad $\mathcal{A}$;
$\mathcal{A}$ produce una risposta rsp;
if $V(pk, \mathsf{ch}, \mathsf{rsp}) = 1$ **then** output 1 (successo);
else output 0 (fallimento);

Definizione 2.4.4 (Schema di Identificazione Sicuro)**.** *Uno schema di identificazione* Π *si dice* ***sicuro*** *(contro attacchi passivi) se per ogni avversario PPT* $\mathcal{A}$ *esiste una funzione trascurabile* $\text{negl}(\cdot)$ *tale che:*

$$\mathbb{P}[\mathsf{Ident}_{A,\Pi}(\lambda) = 1] \leq \text{negl}(\lambda) \tag{2.43}$$

2.5 Firme digitali

Definizione 2.5.1 (Firma digitale)**.** *Uno schema di firma digitale è una tripla* $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ *di algoritmi PPT tali che:*

- $\mathsf{Gen}(1^\lambda) \to (pk, sk)$
- $\sigma \leftarrow \mathsf{Sign}_{sk}(m)$
- $b \leftarrow \mathsf{Verify}_{pk}(m, \sigma)$

Uno schema di firma digitale corretto soddisfa la proprietà di *correttezza*, ovvero per ogni coppia di chiavi generata da *Gen* e per ogni messaggio m, si ha

$$\mathsf{Verify}(pk, m, \mathsf{Sign}(sk, m)) = 1.$$

Dal punto di vista della sicurezza, le firme digitali devono soddisfare tre proprietà fondamentali:

- **Autenticità**, che garantisce che il messaggio sia stato firmato dal soggetto che possiede la chiave segreta associata alla chiave pubblica utilizzata in verifica;
- **Integrità**, che assicura che il messaggio non sia stato modificato dopo la firma. Qualsiasi alterazione del contenuto comporta l'invalidazione della firma;
- **Non ripudio**, che impedisce al firmatario di negare, in un secondo momento, di aver prodotto la firma sul messaggio.

Sia $\Pi = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ una firma digitale. Per ogni avversario $\mathcal{A}$ definiamo come **esperimento di falsificazione della firma** $\mathsf{SigForg}_{A,\Pi}(\lambda)$ in questo modo:

Algoritmo 6: Esperimento di falsificazione della firma

Input: Parametro di sicurezza λ, schema di firma digitale $\Pi = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$
Output: 1 se l'avversario riesce a falsificare una firma, 0 altrimenti
Genera la coppia di chiavi $(pk, sk) \leftarrow \mathsf{Gen}(1^\lambda)$;
Consegna all'avversario $\mathcal{A}$ la chiave pubblica pk e accesso a un oracolo di firma $\mathsf{Sign}_{sk}(\cdot)$;
L'avversario $\mathcal{A}$ restituisce una coppia messaggio-firma (m, σ);
if $\mathsf{Verify}(pk, m, \sigma) = 1$ ***e*** *m non è mai stato richiesto all'oracolo* $\mathsf{Sign}_{sk}(\cdot)$ **then**
  return 1 `// Successo: falsificazione riuscita`
end
else
  return 0 `// Fallimento: falsificazione non riuscita`
end

Definizione 2.5.2. *Una firma digitale* Π *si dice* ***esistenzialmente non falsificabile sotto attacco a messaggi scelti adattivi*** *(Existentially Unforgeable under Adaptive Chosen-Message Attack, EUF-CMA) se, per ogni avversario PPT* $\mathcal{A}$*, esiste una funzione trascurabile* $\mathrm{negl}(\lambda)$ *tale che:*

$$\mathbb{P}[\mathsf{SigForg}_{\mathcal{A},\Pi}(\lambda) = 1] \leq \mathrm{negl}(\lambda).$$

2.6 Trasformazione di Fiat-Shamir

Definizione 2.6.1 (Random Oracle)**.** *Un* random oracle *è una funzione* $H : \{0,1\}^* \to \{0,1\}^\ell$ *tale che:*

- *su input nuovo restituisce un valore uniforme casuale*
- *su input già visto restituisce lo stesso valore*

Lemma 2.6.1. *Se una funzione di Hash* H *è modellata come random oracle, allora è one-way.*

Lemma 2.6.2. *Se una funzione di Hash* H *è modellata come random oracle, allora è collision-resistant.*

Lemma 2.6.3. *Sia* H *modellata come un random oracle e sia* $x \in \{0,1\}^*$ *un input arbitrario. Allora, per ogni avversario PPT* $\mathcal{A}$*, esiste una funzione trascurabile* $\mathrm{negl}(\lambda)$ *tale che*

$$\mathbb{P}\big[\mathcal{A} \text{ ottiene } H(x) \text{ senza aver effettuato una query su } x\big] \leq \mathrm{negl}(\lambda),$$

dove la probabilità è calcolata sui risultati casuali generati dall'avversario e dal random oracle.

Definizione 2.6.2 (Trasformazione di Fiat-Shamir)**.** *Sia*

$$\Pi = (\mathcal{P}, \mathcal{V})$$

uno schema di identificazione interattivo a tre mosse del tipo:

$$(\mathsf{com}, \mathsf{ch}, \mathsf{rsp}),$$

dove:

- *il prover* $\mathcal{P}$ *genera un commitment* com*;*
- *il verifier* $\mathcal{V}$ *invia una challenge casuale* ch*;*
- *il prover risponde con una risposta* rsp*.*

La trasformazione di Fiat-Shamir converte Π *in uno schema di firma digitale non interattivo*

$$\Pi' = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$$

sostituendo la challenge casuale del verifier con l'output di una funzione hash modellata come random oracle.

Più precisamente, dato un messaggio μ*, il firmatario:*

1. *genera il commitment* com *eseguendo il primo passo del protocollo interattivo;*
2. *calcola la challenge come*
$$\mathsf{ch} = H(\mathsf{com}, \mu);$$
3. *calcola la risposta* rsp *utilizzando la chiave segreta.*

La firma risultante è:

$$\sigma = (\mathsf{com}, \mathsf{rsp}).$$

L'algoritmo di verifica:

- *ricalcola*
$$\mathsf{ch} = H(\mathsf{com}, \mu);$$
- *verifica che* $(\mathsf{com}, \mathsf{ch}, \mathsf{rsp})$ *costituisca una trascrizione valida del protocollo di identificazione originale.*

Teorema 2.6.4 (Sicurezza della trasformazione di Fiat-Shamir)**.** *Sia* Π *uno schema di identificazione sicuro e sia* Π' *lo schema di firma ottenuto tramite Fiat-Shamir con la funzione* H*. Se* H *è un random oracle, allora* Π' *è sicuro.*

2.7 Distanza statistica

Definizione 2.7.1 (Distanza statistica)**.** *Siano* P_0 *e* P_1 *distribuzioni su un insieme finito* $\mathcal{R}$*. La loro distanza statistica è:*

$$\Delta(P_0, P_1) = \frac{1}{2} \sum_{r \in R} |P_0(r) - P_1(r)| \tag{2.44}$$

Definizione 2.7.2 (Indistinguibilità statistica)**.** *Due distribuzioni* P_0 *e* P_1 *sono statisticamente indistinguibili se:*

$$\Delta(P_0, P_1) \leq \mathrm{negl}(\lambda) \tag{2.45}$$

Capitolo 3

Firme digitali basate sui reticoli

3.1 Introduzione

Le firme digitali basate sui reticoli forniscono un metodo sicuro per autenticare messaggi nel contesto post quantistico. Seguendo la tesi di **Roberto Previtali** Previtali [2025], ci si concentra sugli schemi di firma di **Lyubashevsky**, basati sul problema *Short Integer Solution (SIS)* e sulle varianti basate sui reticoli.

Questi schemi utilizzano:

- Distribuzione gaussiana discreta su $\mathbb{Z}^n$;
- Tecnica di *rejection sampling* per garantire l'indipendenza statistica dalle chiavi segrete;
- Trasformazione di Fiat-Shamir con aborti per convertire protocolli interattivi in firme digitali non interattive.

3.2 Schema di firma di Lyubashevsky

Lo schema può essere descritto attraverso tre algoritmi principali: `Gen`, `Sign` e `Verify`.

3.2.1 Algoritmo Gen

Algoritmo 7: Gen

Input: Parametri di sicurezza: dimensione del reticolo n, modulo q, dimensione m, bound β, deviazione σ
Output: Chiave segreta sk e chiave pubblica pk
begin
Scegli una matrice $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ casuale;
scegli il segreto $\mathbf{S} \leftarrow \{-d, ..., d\}^{m \times k}$ Calcola $\mathbf{T} = \mathbf{AS} \bmod q$;
Imposta $sk = \mathbf{S}$, $pk = (\mathbf{A}, \mathbf{T})$;
return (sk, pk);

3.2.2 Algoritmo Sign

Algoritmo 8: Sign

Input: Messaggio μ, chiave segreta $sk = \mathbf{S}$, chiave pubblica $pk = (\mathbf{A}, \mathbf{T})$, deviazione σ, bound β,bound M
Output: Firma $\sigma = (\mathbf{z}, \mathbf{c})$
begin
repeat
Campiona $\mathbf{y} \leftarrow D_{\mathbf{0},\sigma}^m$ Calcola $\mathbf{c} = H(\mathbf{Ay} \bmod q, \mu)$; `// hash del commitment e messaggio`
Calcola $\mathbf{z} = \mathbf{Sc} + \mathbf{y}$;
Accetta $\mathbf{z}$ con probabilità $p = \min\left(1, \frac{D_{\mathbf{0},\sigma}^m(\mathbf{z})}{M \cdot D_{\mathbf{0},\sigma}^m(\mathbf{y})}\right)$; `// rejection sampling`
until *accettazione*;
return $\sigma = (\mathbf{z}, \mathbf{c})$;

3.2.3 Algoritmo Verify

Algoritmo 9: Verify

Input: Messaggio μ, firma $\sigma = (\mathbf{z}, \mathbf{c})$, chiave pubblica $pk = (\mathbf{A}, \mathbf{T})$, bound $\eta\beta$
Output: Accetta o rifiuta
begin
Verifica che $\mathbf{c} = H(\mathbf{Az} - \mathbf{Tc} \bmod q, \mu)$;
Verifica che $\|\mathbf{z}\| \leq \eta\beta$;
return accetta se entrambe le condizioni sono soddisfatte, altrimenti rifiuta;

3.3 Rejection Sampling e distribuzione gaussiana discreta

Il *rejection sampling* garantisce che la firma σ sia statisticamente indipendente dal segreto $\mathbf{S}$, impedendo a un attaccante di ottenere informazioni sul segreto dalle firme multiple. La probabilità di accettazione è definita come:

$$P_{\text{accept}} = \min\left(\frac{D^m_{\mathbf{0},\sigma}(\mathbf{z})}{M \cdot D^m_{\mathbf{0},\sigma}(\mathbf{y})}, 1\right), \tag{3.1}$$

dove $D^m_{\mathbf{0},\sigma}$ è la distribuzione gaussiana discreta centrata in 0.
L'obiettivo è costruire una distribuzione di output f tale che:

- sia indipendente dalla chiave segreta $\mathbf{S}$;
- produca firme di norma piccola.

Nel protocollo di firma, il vettore generato è della forma:

$$\mathbf{z} = \mathbf{y} + \mathbf{Sc}$$

dove $\mathbf{y} \sim D^m_{\mathbf{0},\sigma}$ è distribuito secondo una gaussiana discreta e $\mathbf{Sc}$ dipende dalla chiave segreta.

Il problema principale è che la distribuzione di $\mathbf{z}$ dipende da $\mathbf{S}$, il che può compromettere la sicurezza.

Si definiscono:

- **Distribuzione target**: $f(\mathbf{x}) = D^m_{\mathbf{0},\sigma}$
- **Distribuzione proposta**: $g(\mathbf{x}) = D^m_{\mathbf{Sc},\sigma}$

La distribuzione proposta è una gaussiana discreta centrata in $\mathbf{Sc}$, mentre quella target è centrata in $\mathbf{0}$. Per rendere la distribuzione di $\mathbf{z}$ indipendente da $\mathbf{S}$, si utilizza il rejection sampling.

Dato un campione $\mathbf{z} \sim g$, esso viene accettato con probabilità:

$$\min\left(\frac{D^m_{\mathbf{0},\sigma}(\mathbf{z})}{M \cdot D^m_{\mathbf{Sc},\sigma}(\mathbf{z})}, 1\right)$$

dove M è una costante tale che:

$$D^m_{\mathbf{0},\sigma}(\mathbf{z}) \leq M \cdot D^m_{\mathbf{Sc},\sigma}(\mathbf{z})$$

Affinché il metodo sia efficiente, è necessario che:

$$\sigma = \omega(T\sqrt{\log m})$$

dove $T = \max \|\mathbf{Sc}\|$.

In tal caso, le distribuzioni f e g risultano sufficientemente vicine.

Il seguente risultato garantisce la correttezza della tecnica.

Teorema 3.3.1. *Sia $V \subseteq \mathbb{Z}^m$ un insieme di vettori con norma limitata da T, e sia $\sigma = \omega(T\sqrt{\log m})$. Allora esiste una costante $M = O(1)$ tale che l'algoritmo di rejection sampling produce una distribuzione statisticamente vicina a quella della distribuzione ideale $D^m_{\mathbf{0},\sigma}$.*

In particolare, la distanza statistica tra la distribuzione reale e quella ideale è trascurabile, e la probabilità di accettazione è circa $1/M$. Dati quindi:

- parametro di sicurezza λ bits;
- un algoritmo $\mathcal{A}$ che utilizza il rejection sampling per generare un campione $\mathbf{z}$ da una proposal normale discreta m-dimensionale traslata e avendo una distribuzione target normale discreta m-dimensionale centrata in $\mathbf{0}$;
- un algoritmo $\mathcal{F}$ avente come output coppie $(\mathbf{z}, \mathbf{Sc})$ dove $\mathbf{z} \sim D^m_{\mathbf{0},\sigma}$ indipendentemente da $\mathbf{Sc}$;

avremo che se $\sigma = \alpha T$ per ogni $\alpha > 0$, allora $M = e^{12/\alpha + 0.5/\alpha^2}$, l'output di $\mathcal{A}$ ha una distanza statistica di $\frac{2^{-\lambda}}{M}$ dall'output di $\mathcal{F}$ e la probabilità che $\mathcal{A}$ generi un output è dell'ordine di $\frac{1}{M} - \frac{2^{-\lambda}}{M}$.
Il teorema implica che, dopo l'applicazione del rejection sampling, il vettore $\mathbf{z}$ è distribuito in modo quasi indipendente da $\mathbf{Sc}$, e quindi dalla chiave segreta $\mathbf{S}$.

Il rejection sampling consente di rendere indipendente la distribuzione delle firme dalla chiave segreta, rendendole compatibili con protocolli zero-knowledge e garantendo proprietà fondamentali come anonimato e non collegabilità.

3.4 Analisi dei parametri di sicurezza

Si analizza ora il ruolo dei principali parametri dello schema di firma e i relativi trade-off tra sicurezza ed efficienza, seguendo l'impostazione di Previtali.

Parametri principali. Il parametro principale di sicurezza λ è fissato ed è considerato pari a 100. I parametri rilevanti sono:

- n: dimensione del problema sottostante;
- m: numero di colonne della matrice (dimensione della firma);
- q: modulo;
- σ: deviazione standard della distribuzione gaussiana;
- $T = \max \|\mathbf{Sc}\|$: bound sul termine dipendente dalla chiave segreta.

Lunghezza delle firme. Dalla distribuzione $z \sim D^m_{\mathbf{0},\sigma}$, si ha:

$$\mathbb{E}[\|z\|_2] \approx \sigma\sqrt{m}$$

Pertanto, la lunghezza delle firme cresce principalmente con m.

Ridurre m comporta firme più corte e maggiore efficienza.

Vincoli di sicurezza. La sicurezza dello schema si basa sulla difficoltà del problema $\mathrm{SIS}_{q,n,m}$. Tuttavia, se m è troppo piccolo, il problema diventa più facile, rendendo possibile il recupero della chiave segreta.

Ne segue che m deve essere scelto sufficientemente grande per garantire la sicurezza.

Condizione sul parametro σ. Per garantire la correttezza del rejection sampling, è necessario che $\sigma = \omega(T\sqrt{\log m})$. Questo vincolo implica che σ non può essere troppo piccolo.

Trade-off. I parametri dello schema devono soddisfare un delicato equilibrio:

- valori grandi di σ migliorano la correttezza statistica, ma aumentano la lunghezza delle firme;
- valori piccoli di m migliorano l'efficienza, ma riducono la sicurezza;
- valori grandi di m aumentano la sicurezza, ma peggiorano le prestazioni.

Implicazioni. L'efficienza complessiva dello schema è dominata dal parametro m, mentre la sicurezza impone un limite inferiore su tale parametro.

3.4.1 Equivalenza tra LWE e SIS a bassa densità

Si considera una matrice:

$$\mathbf{A} = [\bar{\mathbf{A}} \mid \mathbf{I}] \in \mathbb{Z}_q^{n \times 2n} \tag{3.2}$$

dove $\bar{\mathbf{A}} \xleftarrow{\$} \mathbb{Z}_q^{n \times n}$ è campionata uniformemente e $\mathbf{I}$ è la matrice identità.

Allora distinguere coppie $(\mathbf{A}, \mathbf{As})$ da distribuzioni uniformi è computazionalmente equivalente al problema LWE, a meno della distribuzione del segreto $\mathbf{s}$.

Questo mostra che il problema LWE può essere interpretato come una variante del problema SIS a bassa densità.

Gli algoritmi fondamentali dello schema rimangono `Gen`, `Sign` e `Verify`, ma vengono adattati al nuovo modello.

3.4.2 Algoritmo Gen

Algoritmo 10: Gen

Input: Parametri di sicurezza: dimensione n, modulo q, dimensione m, deviazione ψ
Output: Chiave segreta sk e chiave pubblica pk
begin
- Scegli una matrice $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ casuale;
- campiona il segreto $\mathbf{S} \leftarrow D_{\mathbf{0},\psi}^{m \times k}$;
- campiona l'errore $\mathbf{E} \leftarrow D_{\mathbf{0},\psi}^{n \times k}$;
- Calcola $\mathbf{T} = \mathbf{AS} + \mathbf{E} \bmod q$;
- Imposta $sk = \mathbf{S}$, $pk = (\mathbf{A}, \mathbf{T})$;
- **return** (sk, pk);

3.4.3 Algoritmo Sign

Algoritmo 11: Sign

Input: Messaggio μ, chiave segreta $sk = \mathbf{S}$, chiave pubblica $pk = (\mathbf{A}, \mathbf{T})$, deviazione σ, bound β, bound M
Output: Firma $\sigma = (\mathbf{z}, \mathbf{c})$
begin
- **repeat**
  - Campiona $\mathbf{y} \leftarrow D_{\mathbf{0},\sigma}^{m}$;
  - Calcola $\mathbf{u} = \mathbf{Ay} \bmod q$;
  - Calcola $\mathbf{c} = H(\mathbf{u}, \mu)$; `// hash del commitment e messaggio`
  - Calcola $\mathbf{z} = \mathbf{Sc} + \mathbf{y}$;
  - Accetta $\mathbf{z}$ con probabilità

$$p = \min\left(1, \frac{D_{\mathbf{0},\sigma}^{m}(\mathbf{z})}{M \cdot D_{\mathbf{Sc},\sigma}^{m}(\mathbf{z})}\right)$$

  - ; `// rejection sampling`
- **until** *accettazione*;
- **return** $\sigma = (\mathbf{z}, \mathbf{c})$;

3.4.4 Algoritmo Verify

Algoritmo 12: Verify

Input: Messaggio μ, firma $\sigma = (\mathbf{z}, \mathbf{c})$, chiave pubblica $pk = (\mathbf{A}, \mathbf{T})$, bound $\eta\beta$
Output: Accetta o rifiuta
begin
Verifica che

$$\mathbf{c} = H(\mathbf{Az} - \mathbf{Tc} \bmod q, \mu)$$

;
Verifica che $\|\mathbf{z}\|_2 \leq \eta\beta$;
return accetta se entrambe le condizioni sono soddisfatte, altrimenti rifiuta;

Motivazione dell'uso di LWE. Negli schemi basati su SIS, la sicurezza dipende dalla difficoltà di trovare vettori $\mathbf{v}$ tali che:

$$\mathbf{Av} = \mathbf{0} \mod q \tag{3.3}$$

con norma limitata. Tuttavia, nel contesto delle firme, tali vettori dipendono dalla firma stessa:

$$\mathbf{v} = \mathbf{z} - \mathbf{z}' + \mathbf{Sc} - \mathbf{Sc}' \tag{3.4}$$

e quindi la difficoltà del problema dipende dalla norma di $\mathbf{z}$.

Poiché:

$$\|\mathbf{z}\|_2 \approx \sigma\sqrt{m}, \tag{3.5}$$

firme più lunghe possono rendere il problema più facile.

Vantaggi dell'approccio basato su LWE. Basando la sicurezza sul problema LWE:

- la sicurezza della chiave segreta dipende dall'indistinguibilità della distribuzione;
- la sicurezza contro la falsificazione non dipende direttamente dalla norma della firma;

Interpretazione tramite knapsack. Il problema può essere interpretato come una generalizzazione del problema knapsack su reticoli. La difficoltà dipende dalla norma del segreto:

- segreti troppo piccoli o troppo grandi rendono il problema facile;
- la difficoltà massima si ha in una regione intermedia.

Nel caso di schemi basati su SIS, la costruzione può spostarsi fuori da questa regione ottimale, mentre l'approccio basato su LWE mantiene la difficoltà vicino al punto di massima complessità.

Scelta dei parametri. Nel caso LWE, il segreto $\mathbf{s}$ non è scelto uniformemente in un insieme limitato, ma viene campionato secondo una distribuzione gaussiana discreta:

$$\mathbf{s} \sim D^m_{\mathbf{0},\psi} \tag{3.6}$$

dove ψ rappresenta la deviazione standard.

Di conseguenza, la norma di $\mathbf{s}$ è concentrata attorno a:

$$\|\mathbf{s}\|_2 \approx \psi\sqrt{m}. \tag{3.7}$$

Per rendere il modello LWE compatibile con il modello SIS a bassa densità, in cui il segreto è scelto da $\{-d, \dots, d\}^m$, si sceglie ψ in modo che le norme dei due segreti siano comparabili. In particolare:

$$\psi \approx \sqrt{\frac{d(d+1)}{3}}. \tag{3.8}$$

Questa scelta è giustificata dal fatto che la norma di un vettore uniforme in $\{-d, \dots, d\}^m$ è concentrata attorno a $\sqrt{\frac{d(d+1)}{3}}\, m$, mentre la norma del vettore gaussiano è concentrata attorno a $\psi\sqrt{m}$.

Infatti sia quindi:

$$\mathbf{S} \leftarrow \{-d, \dots, d\}^{m\times k}.$$

Ogni coefficiente della matrice segue quindi una distribuzione uniforme discreta in $\{-d, \dots, d\}$.

Sia:

$$\mathbf{S}_{ij} \sim \mathrm{U}(\{-d, \dots, d\}).$$

Poiché la distribuzione è simmetrica attorno a 0, si ha:

$$\mathbb{E}[\mathbf{S}_{ij}] = 0.$$

La varianza coincide dunque con:

$$\mathrm{Var}(\mathbf{S}_{ij}) = \mathbb{E}[\mathbf{S}^2_{ij}].$$

Si calcola:

$$\mathbb{E}[\mathbf{S}^2_{ij}] = \frac{1}{2d+1}\sum_{k=-d}^{d} k^2.$$

Essendo:

$$\sum_{k=-d}^{d} k^2 = 2\sum_{k=1}^{d} k^2,$$

segue:

$$\mathbb{E}[\mathbf{S}^2_{ij}] = \frac{2}{2d+1}\sum_{k=1}^{d} k^2.$$

Utilizzando:

$$\sum_{k=1}^{d} k^2 = \frac{d(d+1)(2d+1)}{6},$$

si ottiene:

$$\mathbb{E}[\mathbf{S}_{ij}^2] = \frac{2}{2d+1} \cdot \frac{d(d+1)(2d+1)}{6}.$$

Semplificando:

$$\mathbb{E}[\mathbf{S}_{ij}^2] = \frac{d(d+1)}{3}.$$

Pertanto:

$$\mathrm{Var}(\mathbf{S}_{ij}) = \frac{d(d+1)}{3}.$$

Nel modello LWE, invece, i coefficienti della matrice segreta vengono campionati secondo una distribuzione gaussiana discreta:

$$\mathbf{S}_{ij} \sim D_{0,\psi},$$

la cui varianza è approssimativamente:

$$\mathrm{Var}(\mathbf{S}_{ij}) \approx \psi^2.$$

Per rendere confrontabili i due modelli, si sceglie ψ imponendo che le due varianze coincidano:

$$\psi^2 \approx \frac{d(d+1)}{3}.$$

Conclusioni. L'utilizzo del problema LWE al posto del problema SIS consente di ottenere schemi di firma più efficienti e con una struttura di sicurezza più solida. In particolare, il passaggio a segreti gaussiani, parametrizzati da ψ, permette di controllare la distribuzione del segreto e di mantenere l'equilibrio tra sicurezza ed efficienza, rendendo tali schemi adatti per costruzioni avanzate come i sistemi di credenziali anonime.

Capitolo 4

ABDLOP

4.1 Il protocollo di identificazione di Lyubashevsky

All'interno di questa sottosezione si riprendono i concetti già presentati da Previtali [2025] (Appendix B). Si consideri il seguente schema di identificazione basato su reticoli, definito sull'anello $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^n + 1)$.

Setup. Sia $\mathbf{A} \in \mathcal{R}_q^{k \times \ell}$ una matrice pubblica campionata uniformemente e sia $\mathbf{s} \in \mathcal{R}_q^{\ell}$ un vettore segreto tale che $\|\mathbf{s}\|$ è piccolo. Si definisce:

$$\mathbf{t} = \mathbf{A}\mathbf{s} \pmod{q}.$$

La chiave pubblica è $(\mathbf{A}, \mathbf{t})$, mentre la chiave segreta è $\mathbf{s}$.

Protocollo. Il protocollo tra prover e verifier è definito come segue:

1. Il prover campiona $\mathbf{y} \leftarrow D_{\mathbf{0},\sigma}^{\ell}$ e calcola

   $$\mathbf{w} = \mathbf{A}\mathbf{y} \pmod{q},$$

   inviando $\mathbf{w}$ al verifier.

2. Il verifier sceglie una challenge $c \leftarrow \mathcal{C}$ e la invia al prover.

3. Il prover calcola

   $$\mathbf{z} = \mathbf{y} + c\mathbf{s},$$

   e applica rejection sampling; se il campione è accettato, invia $\mathbf{z}$ al verifier, altrimenti riavvia il protocollo.

4. Il verifier accetta se:

   $$\mathbf{A}\mathbf{z} = \mathbf{w} + c\mathbf{t} \mod q \quad \text{e} \quad \|\mathbf{z}\|_2 \leq B,$$

   per un opportuno bound B.

4.2 Analisi come Σ-protocollo

Si analizza ora il protocollo rispetto alle proprietà dei Σ-protocolli definite nella sezione 2.4.

4.2.1 Completezza

Proposizione 4.2.1. *Il protocollo soddisfa la proprietà di completezza 2.4.2.*

Dimostrazione. Se il prover e il verifier seguono il protocollo onestamente, allora:

$$\mathbf{z} = \mathbf{y} + c\mathbf{s}.$$

Applicando $\mathbf{A}$ si ottiene:

$$\mathbf{Az} = \mathbf{Ay} + c\mathbf{As} = \mathbf{w} + c\mathbf{t} \pmod q,$$

dove si è utilizzato che $\mathbf{w} = \mathbf{Ay}$ e $\mathbf{t} = \mathbf{As}$. Inoltre, poiché sia $\mathbf{y}$ che $\mathbf{s}$ sono vettori di norma piccola, anche $\mathbf{z}$ risulta essere piccolo con alta probabilità. Pertanto, il verifier accetta sempre. □

4.2.2 Honest-Verifier Zero-Knowledge

Proposizione 4.2.2. *Il protocollo soddisfa la proprietà di honest-verifier zero-knowledge 2.4.2.*

Dimostrazione. Si dimostra l'esistenza di un simulatore efficiente $\mathcal{S}$ che, dato in input $\mathbf{A}$, $\mathbf{t}$ e una challenge $c \in \mathcal{C}$ scelta onestamente, è in grado di produrre una trascrizione $(\mathbf{w}, c, \mathbf{z})$ indistinguibile da una trascrizione reale del protocollo.

Il simulatore $\mathcal{S}$ opera come segue:

1. Campiona $\mathbf{z} \leftarrow D^m_{\mathbf{0},\sigma}$ soggetto al vincolo $\|\mathbf{z}\|_2 \leq B$.
2. Calcola
$$\mathbf{w} = \mathbf{Az} - c\mathbf{t} \pmod q.$$
3. Restituisce la trascrizione $(\mathbf{w}, c, \mathbf{z})$.

Si osserva innanzitutto che la trascrizione simulata soddisfa sempre la condizione di verifica:

$$\mathbf{Az} = \mathbf{w} + c\mathbf{t} \pmod q,$$

per costruzione.

Resta da dimostrare che la distribuzione delle trascrizioni simulate è indistinguibile da quella delle trascrizioni reali.

Nel protocollo reale, il prover campiona $\mathbf{y} \leftarrow D^m_{\mathbf{0},\sigma}$ e calcola:

$$\mathbf{z} = \mathbf{y} + c\mathbf{s},$$

accettando il campione $\mathbf{z}$ tramite rejection sampling in modo che la distribuzione finale di $\mathbf{z}$ sia statisticamente vicina a $D^m_{\mathbf{0},\sigma}$ indipendentemente dal segreto $\mathbf{s}$.

Pertanto, per il verifier:

- $\mathbf{z}$ è distribuito (statisticamente vicino) come $D^m_{\mathbf{0},\sigma}$ condizionato a $\|\mathbf{z}\|_2 \leq B$;
- $\mathbf{w}$ è determinato come
$$\mathbf{w} = \mathbf{A}\mathbf{z} - c\mathbf{t} \pmod q.$$

Questa è, infatti, la stessa distribuzione prodotta dal simulatore $\mathcal{S}$.

Ne segue che le distribuzioni delle trascrizioni simulate e reali sono statisticamente indistinguibili. Pertanto, il protocollo soddisfa la proprietà di honest-verifier zero-knowledge. □

4.2.3 Limiti della special soundness

Si mostra ora che il protocollo non soddisfa pienamente la proprietà di special soundness 2.4.2.

Proposizione 4.2.3. *Il protocollo non è una proof of knowledge nel senso standard, poiché non è garantita l'estrazione di un segreto* $\mathbf{s}$ *avente norma limitata.*

Dimostrazione. Siano dati due transcript validi con lo stesso commitment $\mathbf{w}$ e challenge distinte $c \neq c'$:
$$(\mathbf{w}, c, \mathbf{z}), \quad (\mathbf{w}, c', \mathbf{z}').$$
Poiché il verifier accetta entrambe le trascrizioni, si ha:
$$\mathbf{A}\mathbf{z} = \mathbf{w} + c\mathbf{t} \pmod q,$$
$$\mathbf{A}\mathbf{z}' = \mathbf{w} + c'\mathbf{t} \pmod q.$$
Sottraendo membro a membro:
$$\mathbf{A}(\mathbf{z}' - \mathbf{z}) = (c' - c)\mathbf{t} \pmod q.$$
Poiché $\mathbf{t} = \mathbf{A}\mathbf{s}$, segue che:
$$\mathbf{A}(\mathbf{z}' - \mathbf{z}) = \mathbf{A}((c' - c)\mathbf{s}) \pmod q.$$
Pertanto:
$$\mathbf{A}(\mathbf{z}' - \mathbf{z} - (c' - c)\mathbf{s}) = 0 \pmod q.$$

Idealmente, si vorrebbe definire
$$\mathbf{s} = (c' - c)^{-1}(\mathbf{z}' - \mathbf{z}).$$

Tuttavia, nel contesto degli anelli su reticoli:

- l'elemento $(c' - c)$ potrebbe non essere invertibile in $\mathcal{R}_q$;
- anche quando è invertibile, l'inverso può amplificare significativamente la norma dei coefficienti.

Di conseguenza, non è garantita l'estrazione efficiente di un vettore $\mathbf{s}$ di norma limitata tale che $\mathbf{A}\mathbf{s} = \mathbf{t}$. □

4.2.4 Conseguenze

Il risultato precedente implica che il protocollo non costituisce una proof of knowledge del segreto $\mathbf{s}$, bensì una proof of knowledge di una relazione derivata. In particolare, è possibile dimostrare la conoscenza di una coppia $(\mathbf{z}, c)$ tale che:

$$\mathbf{Az} = c\mathbf{t} \pmod{q},$$

ma non la conoscenza di un preimage corto $\mathbf{s}$ di $\mathbf{t}$.

Questa distinzione è cruciale in applicazioni crittografiche avanzate. Ad esempio, in scenari come la cifratura vera e propria, è necessario dimostrare la conoscenza di un vettore corto $\mathbf{s}$ tale che $\mathbf{As} = \mathbf{t}$. Il protocollo precedente non garantisce tale proprietà, poiché il vettore corto è associato a $c\mathbf{t}$ e non a $\mathbf{t}$ stesso.

Per superare questo limite, è necessario introdurre uno schema di commitment al prover di provare determinare determinate proprietà in zero knowledge.

In particolare, si desidera un meccanismo che consenta di:

- impegnarsi a un valore $\mathbf{s}$;
- dimostrare che $\mathbf{s}$ soddisfa determinate proprietà (ad esempio, $\|\mathbf{s}\|_2 \leq \beta$);
- garantire la proprietà di knowledge soundness, ovvero la possibilità di estrarre un preimage corto $\mathbf{s}$ del valore pubblico $\mathbf{t}$, superando il limite del protocollo di identificazione di Lyubashevsky in cui è possibile estrarre soltanto soluzioni della forma $\mathbf{z}$ tali che $\mathbf{Az} = c\mathbf{t}$.

Questo porta alla costruzione di schemi di commitment strutturati su reticoli. In particolare, Lyubashevsky introduce lo schema di commitment ABDLOP.

4.3 Definizione dello schema

4.3.1 Ajtai commitment scheme

Nel commitment scheme di Ajtai, introdotto in Ajtai [1996], si vuole impegnare (commit) un messaggio $\mathbf{s}_1$ utilizzando un vettore random $\mathbf{s}_2$, entrambi aventi norma piccola. Il commitment è definito come:

$$\mathbf{A}_1\mathbf{s}_1 + \mathbf{A}_2\mathbf{s}_2 = \mathbf{t} \pmod{q}. \tag{4.1}$$

La sicurezza di questo schema si basa sul problema SIS 2.12: infatti, trovare due coppie distinte $(\mathbf{s}_1, \mathbf{s}_2)$ e $(\mathbf{s}_1', \mathbf{s}_2')$ che producono lo stesso valore $\mathbf{t}$ equivale a risolvere SIS su $\mathcal{R}_q$.

La proprietà di *hiding* deriva invece dall'indistinguibilità della distribuzione $(\mathbf{A}_2, \mathbf{A}_2\mathbf{s}_2)$ da una distribuzione uniforme.

Un vantaggio importante di questo schema è che la dimensione del messaggio $\mathbf{s}_1$ non influisce sulla dimensione del commitment. Inoltre, la sicurezza non dipende in modo significativo dalla dimensione della soluzione, ma è necessario che i coefficienti di $\mathbf{s}_1$ siano piccoli.

4.3.2 BDLOP commitment scheme

Il commitment scheme BDLOP Benhamouda et al. [2013] consente di impegnare un messaggio $\mathbf{m}$ utilizzando una variabile casuale $\mathbf{s}$, come segue:

$$\begin{bmatrix}\mathbf{A}\\ \mathbf{B}\end{bmatrix}\mathbf{s}+\begin{bmatrix}\mathbf{0}\\ \mathbf{m}\end{bmatrix}=\begin{bmatrix}\mathbf{t}_A\\ \mathbf{t}_B\end{bmatrix} \pmod q. \tag{4.2}$$

In questo caso, solo $\mathbf{s}$ deve avere norma piccola, mentre il messaggio $\mathbf{m}$ può avere coefficienti arbitrari modulo q. L'apertura del commitment consiste unicamente in $\mathbf{s}$, che determina univocamente $\mathbf{m}$.

Anche qui, la sicurezza (binding) si riconduce al problema SIS: due aperture diverse implicano una soluzione non banale di SIS per la matrice $\mathbf{A}$. La proprietà di hiding si basa sull'indistinguibilità dalla distribuzione uniforme della coppia:

$$\left(\begin{bmatrix}\mathbf{A}\\ \mathbf{B}\end{bmatrix},\begin{bmatrix}\mathbf{A}\\ \mathbf{B}\end{bmatrix}\mathbf{s}\right).$$

Questo schema presenta due vantaggi principali:

- i coefficienti del messaggio $\mathbf{m}$ possono essere arbitrari modulo q;
- è possibile estendere efficientemente il commitment per includere nuovi messaggi, aumentando solo linearmente la dimensione del commitment.

Lo svantaggio principale è che sia la dimensione del commitment che quella dell'apertura crescono linearmente con la dimensione di $\mathbf{m}$.

4.3.3 ABDLOP commitment scheme

Per ottenere uno schema più flessibile, è possibile combinare i vantaggi dei due approcci precedenti. Nasce così il commitment scheme ABDLOP in Lyubashevsky et al. [2022], che consente di impegnare simultaneamente:

- un vettore $\mathbf{s}_1$ a norma piccola;
- un messaggio $\mathbf{m}$ con coefficienti arbitrari modulo q.

Il commitment è definito come:

$$\begin{bmatrix}\mathbf{A}_1\\ \mathbf{0}\end{bmatrix}\mathbf{s}_1+\begin{bmatrix}\mathbf{A}_2\\ \mathbf{B}\end{bmatrix}\mathbf{s}_2+\begin{bmatrix}\mathbf{0}\\ \mathbf{m}\end{bmatrix}=\begin{bmatrix}\mathbf{t}_A\\ \mathbf{t}_B\end{bmatrix} \pmod q. \tag{4.3}$$

In questo schema, la casualità è data da $\mathbf{s}_2$, mentre $\mathbf{s}_1$ rappresenta un segreto a norma piccola.

ABDLOP combina quindi:

- la compattezza e l'efficienza dell'Ajtai commitment;
- la flessibilità del BDLOP commitment rispetto a messaggi di grande ampiezza.

Schema	**Messaggi grandi**	**Segreti piccoli**	**Proof of Knowledge**
Ajtai	No	Sì	No
BDLOP	Sì	No	No
ABDLOP	Sì	Sì	In senso rilassato

Tabella 4.1. Confronto tra diversi schemi di commitment basati su reticoli

La Tabella evidenzia le differenze fondamentali tra i principali schemi di commitment basati su reticoli.

Nota 5. *Una proof of knowledge in senso rilassato è una variante della proof of knowledge classica nella quale la proprietà di estrazione è indebolita. In particolare, se un prover convince il verifier con probabilità sufficientemente elevata, esiste un estrattore efficiente che è in grado di ricavare non necessariamente il witness esatto utilizzato dal prover, ma un witness che soddisfa la relazione richiesta entro un errore limitato.*

Lo schema di commitment completo viene presentato in Lyubashevsky et al. [2022] come segue:

Protocollo ABDLOP

Informazioni private: $(\mathbf{s_1}, \mathbf{m}, \mathbf{s_2}) \in \mathcal{R}_q^{m_1+m_2+l} : \|\mathbf{s_1}\| \leq \alpha_1, \|\mathbf{s_2}\| \leq \alpha_2$

Informazioni pubbliche: $\mathbf{A_1} \in \mathcal{R}_q^{n \times m_1}, \mathbf{A_2} \in \mathcal{R}_q^{n \times m_2}, \mathbf{B} \in \mathcal{R}_q^{l \times m_2}, \mathbf{R_1} \in \mathcal{R}_q^{N \times m_1}, \mathbf{R_m} \in \mathcal{R}_q^{N \times l}$

$$\begin{bmatrix} \mathbf{A}_1 \\ \mathbf{0} \end{bmatrix} \mathbf{s}_1 + \begin{bmatrix} \mathbf{A}_2 \\ \mathbf{B} \end{bmatrix} \mathbf{s}_2 + \begin{bmatrix} \mathbf{0} \\ \mathbf{m} \end{bmatrix} = \begin{bmatrix} \mathbf{t}_A \\ \mathbf{t}_B \end{bmatrix}, \ \mathbf{u} = \mathbf{R_1 s_1} + \mathbf{R_m m}$$

Prover		**Verifier**
$\mathbf{y_1} \leftarrow D^{m_1}_{\mathbf{0},\sigma_1}$		
$\mathbf{y_2} \leftarrow D^{m_2}_{\mathbf{0},\sigma_2}$		
$\mathbf{w} = \mathbf{A_1 y_1} + \mathbf{A_2 y_2}$		
$\mathbf{v} = \mathbf{R_1 y_1} - \mathbf{R_m B y_2}$		
	$\xrightarrow{(\mathbf{w},\mathbf{v})}$	
		$c \leftarrow \mathcal{C}$
	$\xleftarrow{c}$	
$\mathbf{z_1} = \mathbf{y_1} + c\,\mathbf{s_1}$		
$\mathbf{z_2} = \mathbf{y_2} + c\,\mathbf{s_2}$		
if $Rej(\mathbf{z}_i, c\mathbf{s}_i, \sigma_i) = 1$ then abort		
	$\xrightarrow{(\mathbf{z_1},\mathbf{z_2})}$	

Verifier accepts if:

$$\mathbf{A_1 z_1} + \mathbf{A_2 z_2} = \mathbf{w} + c\,\mathbf{t_A},$$
$$\mathbf{R_1 z_1} + \mathbf{R_m} c\,\mathbf{t_B} - \mathbf{R_m B z_2} = \mathbf{v} + c\,\mathbf{u},$$
$$\|\mathbf{z_1}\|_2 \leq B_1, \qquad \|\mathbf{z_2}\|_2 \leq B_2.$$

Nota 6. *Nel protocollo ABDLOP, il challenge space $\mathcal{C} \subseteq \mathcal{R}_q$ consiste di elementi scalari scelti dal verifier e utilizzati per mascherare i vettori segreti del prover. In particolare, ogni challenge $c \in \mathcal{C}$ viene combinato con un vettore segreto $\mathbf{r} \in \mathcal{R}_q^\ell$, L'insieme*

$$V = \{c\mathbf{r} \mid c \in \mathcal{C} \subseteq \mathcal{R}_q, \mathbf{r} \in \mathcal{R}_q^\ell\}$$

Per garantire la correttezza del protocollo e l'efficacia del rejection sampling, è necessario imporre un bound sulla norma di tali vettori. Si definisce

$$\|c\mathbf{r}\|_2 \leq \beta \cdot \|\mathbf{r}\|_2,$$

dove in Lyubashevsky et al. [2022] $\beta = (\|\sigma_{-1}(c^k)c^k\|_1)^{1/(2k)}$ e $\sigma_i \in Aut(\mathcal{R}_q)$ è tale che $\sigma_i(X) = X^i$ (pertanto $\sigma_{-i}(\sigma_i(X)) = X = I(X)$).

4.4 Proprietà di sicurezza

In questa sezione si presentano le principali proprietà di sicurezza di ABDLOP. Si mostra che il protocollo è corretto e soddisfa le proprietà di hiding, binding, honest-verifier zero-knowledge e knowledge extractability. Complessivamente, queste caratteristiche confermano che ABDLOP costituisce un protocollo sicuro basato sulle difficoltà computazionali di Module-SIS e LWE $\mathcal{R}_q^m$.

4.4.1 Correttezza

Proposizione 4.4.1 (Correttezza)**.** *Se il prover e il verifier eseguono il protocollo ABDLOP onestamente, il verifier accetta sempre.*

Dimostrazione. Il prover calcola:

$$\mathbf{z}_1 = \mathbf{y}_1 + c\mathbf{s}_1, \quad \mathbf{z}_2 = \mathbf{y}_2 + c\mathbf{s}_2.$$

Allora:

$$\mathbf{A}_1\mathbf{z}_1 + \mathbf{A}_2\mathbf{z}_2 = \mathbf{A}_1(\mathbf{y}_1 + c\mathbf{s}_1) + \mathbf{A}_2(\mathbf{y}_2 + c\mathbf{s}_2) = (\mathbf{A}_1\mathbf{y}_1 + \mathbf{A}_2\mathbf{y}_2) + c(\mathbf{A}_1\mathbf{s}_1 + \mathbf{A}_2\mathbf{s}_2) = \mathbf{w} + c\mathbf{t}_A.$$

Analogamente, per la seconda equazione:

$$\mathbf{R}_1\mathbf{z}_1 + \mathbf{R}_m c\mathbf{t}_B - \mathbf{R}_m\mathbf{B}\mathbf{z}_2 = \mathbf{v} + c\mathbf{u}.$$

Infine, si considera la norma di $\mathbf{z}_i$:

$$\|\mathbf{z}_i\|_2 = \|\mathbf{y}_i + c\mathbf{s}_i\|_2 \leq \|\mathbf{y}_i\|_2 + \|c\mathbf{s}_i\|_2 \leq \sigma\sqrt{md} + \beta\|\mathbf{s}\|_2$$

poichè $\mathbf{y}_i$ è campionato da una gaussiana centrata in $\mathbf{0}$ e β è definito in 6. Nella sezione 4.6 il bound viene ulteriormente raffinato e si dimostra la validità del bound utilizzato in Lyubashevsky et al. [2022] e Bootle et al. [2023] $B_i = \sigma_i\sqrt{2m_i d}$. □

4.4.2 Hiding

Proposizione 4.4.2 (Hiding)**.** *Il commitment ABDLOP è computazionalmente hiding sotto l'ipotesi della difficoltà del problema Module-LWE (o, equivalentemente, sotto l'indistinguibilità della distribuzione* $(\mathbf{A}_2, \mathbf{B}, \mathbf{A}_2\mathbf{s}_2, \mathbf{B}\mathbf{s}_2)$ *da uniforme).*

Dimostrazione. Si considerano due messaggi distinti $\mathbf{m}, \mathbf{m}' \in \mathcal{R}_q^l$ e si fissano $\mathbf{s}_1$ con $\|\mathbf{s}_1\|_2 \leq \alpha_1$. Si vuole mostrare che i rispettivi commitment sono computazionalmente indistinguibili.

Il commitment ABDLOP è definito come:

$$\begin{bmatrix}\mathbf{A}_1 \\ \mathbf{0}\end{bmatrix}\mathbf{s}_1 + \begin{bmatrix}\mathbf{A}_2 \\ \mathbf{B}\end{bmatrix}\mathbf{s}_2 + \begin{bmatrix}\mathbf{0} \\ \mathbf{m}\end{bmatrix} = \begin{bmatrix}\mathbf{t}_A \\ \mathbf{t}_B\end{bmatrix} \pmod q,$$

dove $\mathbf{s}_2$ è campionato da una distribuzione gaussiana discreta.

Si osserva che:

$$\mathbf{t}_A = \mathbf{A}_1\mathbf{s}_1 + \mathbf{A}_2\mathbf{s}_2, \quad \mathbf{t}_B = \mathbf{B}\mathbf{s}_2 + \mathbf{m}.$$

La componente $\mathbf{t}_A$ non dipende dal messaggio $\mathbf{m}$, mentre tutta l'informazione su $\mathbf{m}$ è contenuta in $\mathbf{t}_B$.

Sotto l'ipotesi Module-LWE, la distribuzione

$$(\mathbf{A}_2, \mathbf{B}, \mathbf{A}_2\mathbf{s}_2, \mathbf{B}\mathbf{s}_2)$$

è computazionalmente indistinguibile da una distribuzione uniforme su

$$(\mathbf{A}_2, \mathbf{B}, \mathbf{u}_1, \mathbf{u}_2),$$

dove $\mathbf{u}_1, \mathbf{u}_2$ sono uniformi in $\mathcal{R}_q$.

Di conseguenza, anche $\mathbf{B}\mathbf{s}_2$ è indistinguibile da un vettore uniforme in $\mathcal{R}_q^l$. Pertanto, $\mathbf{t}_B = \mathbf{B}\mathbf{s}_2 + \mathbf{m}$ è indistinguibile da una distribuzione uniforme traslata, e quindi indipendente da $\mathbf{m}$.

Ne segue che, per ogni coppia $\mathbf{m}, \mathbf{m}'$, le distribuzioni dei commitment corrispondenti sono computazionalmente indistinguibili.

Pertanto, il commitment ABDLOP soddisfa la proprietà di hiding. □

4.4.3 Binding

Proposizione 4.4.3 (Binding)**.** *Il commitment ABDLOP è binding: non è possibile aprire lo stesso commitment con due coppie distinte* $(\mathbf{s}_1, \mathbf{s}_2, \mathbf{m}) \neq (\mathbf{s}_1', \mathbf{s}_2', \mathbf{m}')$ *modulo* q*, tranne che risolvendo il problema SIS.*

Dimostrazione. Si supponga che esistano due aperture distinte:

$$\begin{bmatrix}\mathbf{A}_1 \\ \mathbf{0}\end{bmatrix}\mathbf{s}_1 + \begin{bmatrix}\mathbf{A}_2 \\ \mathbf{B}\end{bmatrix}\mathbf{s}_2 + \begin{bmatrix}\mathbf{0} \\ \mathbf{m}\end{bmatrix} = \begin{bmatrix}\mathbf{A}_1 \\ \mathbf{0}\end{bmatrix}\mathbf{s}_1' + \begin{bmatrix}\mathbf{A}_2 \\ \mathbf{B}\end{bmatrix}\mathbf{s}_2' + \begin{bmatrix}\mathbf{0} \\ \mathbf{m}'\end{bmatrix} \pmod q.$$

Sottraendo membro a membro si ottiene:

$$\begin{bmatrix}\mathbf{A}_1 \\ \mathbf{0}\end{bmatrix}(\mathbf{s}_1 - \mathbf{s}_1') + \begin{bmatrix}\mathbf{A}_2 \\ \mathbf{B}\end{bmatrix}(\mathbf{s}_2 - \mathbf{s}_2') + \begin{bmatrix}\mathbf{0} \\ \mathbf{m} - \mathbf{m}'\end{bmatrix} = 0 \pmod q.$$

Si considerano separatamente i due blocchi.

Primo blocco:

$$\mathbf{A}_1(\mathbf{s}_1 - \mathbf{s}_1') + \mathbf{A}_2(\mathbf{s}_2 - \mathbf{s}_2') = 0 \pmod q.$$

Se $(\mathbf{s}_1, \mathbf{s}_2) \neq (\mathbf{s}_1', \mathbf{s}_2')$, allora il vettore

$$(\mathbf{s}_1 - \mathbf{s}_1', \mathbf{s}_2 - \mathbf{s}_2')$$

è una soluzione non banale dell'istanza Module-SIS definita dalla matrice $(\mathbf{A}_1 \mid \mathbf{A}_2)$.

Poiché i vettori coinvolti sono di norma limitata, ciò contraddice l'ipotesi di difficoltà del problema Module-SIS.

Secondo blocco:

$$\mathbf{B}(\mathbf{s}_2 - \mathbf{s}_2') + (\mathbf{m} - \mathbf{m}') = 0 \pmod q.$$

Se invece $\mathbf{s}_2 = \mathbf{s}_2'$, dalla relazione sopra segue immediatamente:

$$\mathbf{m} = \mathbf{m}' \pmod q.$$

Pertanto, in questo caso deve necessariamente valere anche $\mathbf{s}_1 = \mathbf{s}_1'$, altrimenti si otterrebbe nuovamente una soluzione non banale di Module-SIS dal primo blocco. In conclusione, qualsiasi coppia di aperture distinte dello stesso commitment implica l'esistenza di una soluzione a norma limitata non banale del problema Module-SIS, che è computazionalmente difficile da trovare.

Pertanto, ABDLOP è binding. □

4.4.4 Honest-Verifier Zero-Knowledge

Proposizione 4.4.4 (HVZK)**.** *Il protocollo ABDLOP è honest-verifier zero-knowledge.*

Dimostrazione. Si costruisce un simulatore $\mathcal{S}$ che, dato un input pubblico $(\mathbf{A}_1, \mathbf{A}_2, \mathbf{B}, \mathbf{t}_A, \mathbf{t}_B)$ e una challenge $c \leftarrow \mathcal{C}$, produce transcript non distinguibili da quelli reali.

Il simulatore agisce come segue:

1. Campiona:
$$\mathbf{z}_1 \leftarrow D^{m_1}_{\mathbf{0},\sigma_1}, \quad \mathbf{z}_2 \leftarrow D^{m_2}_{\mathbf{0},\sigma_2}$$
e applica il rejection sampling.

2. Calcola:
$$\mathbf{w} = \mathbf{A}_1\mathbf{z}_1 + \mathbf{A}_2\mathbf{z}_2 - c\,\mathbf{t}_A \pmod q,$$
$$\mathbf{v} = \mathbf{R}_1\mathbf{z}_1 + \mathbf{R}_m c\,\mathbf{t}_B - \mathbf{R}_m\mathbf{B}\mathbf{z}_2 - c\,\mathbf{u} \pmod q.$$

3. Restituisce:
$$(\mathbf{w}, \mathbf{v}, c, \mathbf{z}_1, \mathbf{z}_2).$$

Si considera ora la distribuzione del transcript reale.
Nel protocollo reale:

- il prover campiona $\mathbf{y}_i \leftarrow D^{m_i}_{\mathbf{0},\sigma_i}$;
- calcola $\mathbf{z}_i = \mathbf{y}_i + c\mathbf{s}_i$;
- applica rejection sampling per garantire che $\mathbf{z}_i$ sia distribuito secondo una Gaussiana centrata in $\mathbf{0}$ e soddisfa $\|\mathbf{z}_i\|_2 \leq B_i$.

Dal risultato del rejection sampling nei protocolli di Lyubashevsky Lyubashevsky et al. [2022], la distribuzione di $\mathbf{z}_i$ è statisticamente vicina a:

$$D^{m_i}_{\mathbf{0},\sigma_i} \text{ condizionata a } \|\mathbf{z}_i\|_2 \leq B_i,$$

ed è inoltre indipendente dal segreto $\mathbf{s}_i$.

Dato che:

$$\mathbf{A}_1\mathbf{z}_1 + \mathbf{A}_2\mathbf{z}_2 = \mathbf{w} + c\mathbf{t}_A,$$

per costruzione di $\mathbf{w}$.

Allo stesso modo,

$$\mathbf{R}_1\mathbf{z}_1 + \mathbf{R}_m c\mathbf{t}_B - \mathbf{R}_m\mathbf{B}\mathbf{z}_2 = \mathbf{v} + c\mathbf{u},$$

per costruzione di $\mathbf{v}$. Si noti che le quantità $(\mathbf{w}, \mathbf{v})$ sono totalmente determinate da $(\mathbf{z}_1, \mathbf{z}_2)$ e dalla challenge c.

Poiché il simulatore campiona $\mathbf{z}_1, \mathbf{z}_2$ dalla stessa distribuzione del prover, anche l'intera trascrizione simulata è statisticamente indistinguibile da quella reale.

Ne segue che esiste un simulatore efficiente $\mathcal{S}$ che produce transcript indistinguibili da quelli reali dati $(\mathbf{A}_1, \mathbf{A}_2, \mathbf{B}, \mathbf{t}_A, \mathbf{t}_B)$ e una challenge $c \leftarrow \mathcal{C}$ e quindi il protocollo ABDLOP è honest-verifier zero-knowledge. □

4.4.5 Knowledge Extractability

Proposizione 4.4.5 (Knowledge Extractability)**.** *Siano dati due transcript validi*

$$(\mathbf{w}, \mathbf{v}, c, \mathbf{z}_1, \mathbf{z}_2)$$

e

$$(\mathbf{w}, \mathbf{v}, c', \mathbf{z}'_1, \mathbf{z}'_2)$$

con lo stesso commitment iniziale $(\mathbf{w}, \mathbf{v})$ *e con* $c \neq c'$*.*
Allora esiste un estrattore efficiente che ricava un witness valido

$$(\hat{\mathbf{s}}_1, \hat{\mathbf{s}}_2, \hat{\mathbf{m}})$$

che soddisfa la relazione ABDLOP.

Dimostrazione. Dalle due equazioni di verifica

$$\mathbf{A}_1\mathbf{z}_1 + \mathbf{A}_2\mathbf{z}_2 = \mathbf{w} + c\mathbf{t}_A,$$

$$\mathbf{A}_1\mathbf{z}'_1 + \mathbf{A}_2\mathbf{z}'_2 = \mathbf{w} + c'\mathbf{t}_A,$$

sottraendo membro a membro si ottiene

$$\mathbf{A}_1(\mathbf{z}_1 - \mathbf{z}'_1) + \mathbf{A}_2(\mathbf{z}_2 - \mathbf{z}'_2) = (c - c')\mathbf{t}_A.$$

Poiché $c - c'$ è invertibile per un'opportuna selezione del challenge space, si definiscono

$$\hat{\mathbf{s}}_1 = (c - c')^{-1}(\mathbf{z}_1 - \mathbf{z}'_1),$$

$$\hat{\mathbf{s}}_2 = (c - c')^{-1}(\mathbf{z}_2 - \mathbf{z}'_2).$$

Segue immediatamente

$$\mathbf{A}_1\hat{\mathbf{s}}_1 + \mathbf{A}_2\hat{\mathbf{s}}_2 = \mathbf{t}_A.$$

Si considera ora la seconda equazione di verifica:

$$\mathbf{R}_1\mathbf{z}_1 + \mathbf{R}_m c\mathbf{t}_B - \mathbf{R}_m\mathbf{B}\mathbf{z}_2 = \mathbf{v} + c\mathbf{u},$$

$$\mathbf{R}_1\mathbf{z}_1' + \mathbf{R}_m c'\mathbf{t}_B - \mathbf{R}_m\mathbf{B}\mathbf{z}_2' = \mathbf{v} + c'\mathbf{u}.$$

Sottraendo e moltiplicando per $(c - c')^{-1}$ si ottiene

$$\mathbf{R}_1\hat{\mathbf{s}}_1 + \mathbf{R}_m\mathbf{t}_B - \mathbf{R}_m\mathbf{B}\hat{\mathbf{s}}_2 = \mathbf{u}.$$

Si definisce quindi

$$\hat{\mathbf{m}} = \mathbf{t}_B - \mathbf{B}\hat{\mathbf{s}}_2.$$

Allora

$$\mathbf{R}_1\hat{\mathbf{s}}_1 + \mathbf{R}_m\hat{\mathbf{m}} = \mathbf{u}.$$

Pertanto

$$\begin{bmatrix}\mathbf{A}_1\\ \mathbf{0}\end{bmatrix}\hat{\mathbf{s}}_1 + \begin{bmatrix}\mathbf{A}_2\\ \mathbf{B}\end{bmatrix}\hat{\mathbf{s}}_2 + \begin{bmatrix}\mathbf{0}\\ \hat{\mathbf{m}}\end{bmatrix} = \begin{bmatrix}\mathbf{t}_A\\ \mathbf{t}_B\end{bmatrix},$$

ossia $(\hat{\mathbf{s}}_1, \hat{\mathbf{s}}_2, \hat{\mathbf{m}})$ è un witness valido.
Quindi il protocollo soddisfa la proprietà di knowledge extractability. □

4.5 ABDLOP come Proof of Knowledge

Teorema 4.5.1 (Lyubashevsky, 2022)**.** *Sia* $(\bar{\mathbf{s}}_1, \bar{\mathbf{s}}_2, \bar{\mathbf{c}}) \in \mathcal{R}_q^{m_1} \times \mathcal{R}_q^{m_2} \times \bar{\mathcal{C}}$ *una tripletta tale che:*

$$\mathbf{A}_1\bar{\mathbf{s}}_1 + \mathbf{A}_2\bar{\mathbf{s}}_2 = \mathbf{t}_A \pmod q, \tag{4.4}$$

$$\|\bar{\mathbf{s}}_i\bar{\mathbf{c}}\|_2 \le 2\sigma_i\sqrt{m_i} \quad \textit{per } i = 1,2, \tag{4.5}$$

$$\mathbf{R}_1\bar{\mathbf{s}}_1 + \mathbf{R}_m(\mathbf{t}_B - \mathbf{B}\bar{\mathbf{s}}_2) = \mathbf{u} \pmod q. \tag{4.6}$$

Allora, assumendo che il problema $MSIS_{n,m_1+m_2,B}$ *sia difficile per*

$$B = 8\eta\sqrt{2m_1 d\,\sigma_1^2 + 2m_2 d\,\sigma_2^2},$$

vale che:

1. *la coppia* $(\bar{\mathbf{s}}_1, \bar{\mathbf{s}}_2)$ *è unica;*

2. *per qualsiasi due trascrizioni valide* $(\mathbf{w}, \mathbf{v}, c, \mathbf{z}_1, \mathbf{z}_2)$ *e* $(\mathbf{w}, \mathbf{v}, c', \mathbf{z}_1', \mathbf{z}_2')$ *con* $c \neq c'$*, si ha*

$$\mathbf{z}_i - c\bar{\mathbf{s}}_i = \mathbf{z}_i' - c'\bar{\mathbf{s}}_i, \quad i = 1{,}2.$$

Dimostrazione. Siano date due trascrizioni accettate:

$$(\mathbf{w}, \mathbf{v}, c, \mathbf{z}_1, \mathbf{z}_2), \quad (\mathbf{w}, \mathbf{v}, c', \mathbf{z}_1', \mathbf{z}_2')$$

ottenute riavvolgendo il prover sulla stessa prima mossa $(\mathbf{w}, \mathbf{v})$, con $c \neq c'$.

Si definisce:

$$\bar{\mathbf{z}}_i = \mathbf{z}_i - \mathbf{z}_i', \quad \bar{c} = c - c'.$$

Poiché entrambe le trascrizioni sono accettate, sottraendo le equazioni di verifica si ottiene:

$$\mathbf{A}_1\bar{\mathbf{z}}_1 + \mathbf{A}_2\bar{\mathbf{z}}_2 = \bar{c}\,\mathbf{t}_A.$$

Dividendo per $\bar{c}$, si ottiene:

$$\mathbf{A}_1\bar{\mathbf{s}}_1 + \mathbf{A}_2\bar{\mathbf{s}}_2 = \mathbf{t}_A, \quad \text{dove} \quad \bar{\mathbf{s}}_i = \bar{\mathbf{z}}_i\bar{c}^{-1}.$$

Poiché $\|\mathbf{z}_i\|_2 \leq B_i, \|\mathbf{z}_i'\|_2 \leq B_i$, si ha:

$$\|\bar{\mathbf{z}}_i\|_2 \leq 2B_i,$$

quindi $\bar{\mathbf{s}}_i\bar{c}$ sono vettori di norma piccola.

Si considera ora la seconda equazione di verifica. Sottraendo le due istanze si ottiene:

$$\mathbf{R}_1\bar{\mathbf{z}}_1 + \mathbf{R}_m(\bar{c}\,\mathbf{t}_B - \mathbf{B}\bar{\mathbf{z}}_2) - \bar{c}\,\mathbf{u} = 0.$$

Moltiplicando per $\bar{c}^{-1}$ e sostituendo $\bar{\mathbf{s}}_i = \bar{\mathbf{z}}_i\bar{c}^{-1}$, si ottiene:

$$\mathbf{R}_1\bar{\mathbf{s}}_1 + \mathbf{R}_m(\mathbf{t}_B - \mathbf{B}\bar{\mathbf{s}}_2) = \mathbf{u}.$$

Questo mostra che $(\bar{\mathbf{s}}_1, \bar{\mathbf{s}}_2)$ soddisfa tutte le condizioni del teorema.

Si supponga ora che esista un'altra soluzione distinta $(\bar{\mathbf{s}}_1', \bar{\mathbf{s}}_2') \neq (\bar{\mathbf{s}}_1, \bar{\mathbf{s}}_2)$.

Allora:

$$\mathbf{A}_1\bar{\mathbf{s}}_1 + \mathbf{A}_2\bar{\mathbf{s}}_2 = \mathbf{A}_1\bar{\mathbf{s}}_1' + \mathbf{A}_2\bar{\mathbf{s}}_2'.$$

Moltiplicando per opportuni scalari $\bar{c}, \bar{c}'$ e riordinando, si ottiene una relazione della forma:

$$\mathbf{A}_1\mathbf{x}_1 + \mathbf{A}_2\mathbf{x}_2 = 0,$$

dove $\mathbf{x}_i$ sono vettori non nulli e di norma limitata.

Più precisamente, utilizzando il fatto che la moltiplicazione per elementi del challenge space aumenta la norma al più di un fattore η, si ottiene:

$$\|\mathbf{x}_i\|_2 \leq 8\eta B.$$

Questo fornisce una soluzione non banale al problema Module-SIS con bound piccolo, ma Module-SIS è difficile.

Pertanto, la soluzione $(\bar{\mathbf{s}}_1, \bar{\mathbf{s}}_2)$ è unica.

Infine, per mostrare che

$$\mathbf{z}_i - c\bar{\mathbf{s}}_i = \mathbf{z}'_i - c'\bar{\mathbf{s}}_i,$$

si supponga per assurdo che:

$$\mathbf{z}_i - c\bar{\mathbf{s}}_i = \mathbf{z}'_i - c'\bar{\mathbf{s}}_i + \mathbf{r}.$$

Allora si avrebbe:

$$\bar{\mathbf{s}}_i = \bar{\mathbf{z}}_i\bar{c}^{-1} = \bar{\mathbf{s}}_i + \mathbf{r}\bar{c}^{-1},$$

da cui segue $\mathbf{r} = 0$. □

4.6 Analisi dei parametri

La sicurezza e l'efficienza del protocollo ABDLOP dipendono in modo cruciale dalla scelta dei parametri. In questa sezione si analizzano i principali parametri coinvolti e i vincoli che devono soddisfare. Sia $\lambda \in \mathbb{N}$ il parametro di sicurezza. Tutti i parametri sono solitamente scelti come funzione di λ.

4.6.1 Parametri principali

I parametri fondamentali del sistema sono:

- q: modulo dell'anello $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^d + 1)$ con $q = \text{poly}(\lambda)$;
- d: grado del polinomio (dimensione dell'anello) scelto come $d = \Theta(\lambda)$;
- m_1, m_2: dimensioni dei vettori segreti $\mathbf{s}_1, \mathbf{s}_2$, tipicamente $m_i = \Theta(\lambda)$;
- σ_1, σ_2: deviazioni standard delle distribuzioni gaussiane discrete, scelte come $\text{poly}(\lambda)$;
- B_1, B_2: bound sulle norme $\|\mathbf{z}_1\|, \|\mathbf{z}_2\|$ accettate dal verifier;
- $\mathcal{C}$: challenge space con cardinalità esponenziale in λ;
- η: fattore di espansione associato agli elementi di $c \in \mathcal{C}$.

4.6.2 Il modulo q

Il parametro q definisce l'anello $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^d+1)$ e determina lo spazio in cui vengono effettuati i calcoli.

Affinché le operazioni siano corrette, q deve essere sufficientemente grande da evitare overflow nelle operazioni lineari e da garantire che le distribuzioni coinvolte siano ben definite modulo q. Inoltre, q influisce direttamente sulla sicurezza dei problemi Module-SIS e LWE: valori più grandi aumentano lo spazio delle soluzioni, ma possono ridurre la difficoltà relativa di tali problemi. In particolare, è necessario che q cresca almeno polinomialmente in λ per mantenere la sicurezza del sistema.

4.6.3 B_1, B_2

Affinché il protocollo sia corretto, è necessario che i vettori $\mathbf{z}_i = \mathbf{y}_i + c\mathbf{s}_i$ soddisfino i bound richiesti con alta probabilità. Il bound imposto in Lyubashevsky et al. [2022] è $\|\mathbf{z}_i\|_2 \leq \sigma_i\sqrt{2m_id}$ ed è un risultato ottenuto mediante il seguente lemma di Micciancio and Regev [2007]:

Lemma 4.6.1. *Sia* $\mathbf{y} \leftarrow D^m_{\mathbf{0},\sigma}$. *Allora* $\mathbb{P}(\|\mathbf{y}\| > t\sigma\sqrt{m}) < (te^{0.5-t^2/2})^m$.

Siccome $\mathbf{z}_i = \mathbf{y}_i + c\mathbf{s}_i \sim D^m_{c\mathbf{s}_i,\sigma}$ viene campionato mediante rejection sampling si ha che $\mathbf{z_i}$ si può considerare proveniente da una distribuzione normale centrata in $\mathbf{0}$. Pertanto il risultato 4.6.1 vale anche per $\mathbf{z}_i$ con alta probabilità. Siccome solitamente si scelgono $d = \Theta(\lambda)$, $m_i = \Theta(\lambda)$ e $\sigma_i = \text{poly}(\lambda)$ avremo che $B_i = \sigma_i\sqrt{2m_id} = \text{poly}(\lambda)$.

Nota 7. *Si noti che* $\mathbf{z}_i \in \mathcal{R}_q^{m_i}$ *pertanto si ha* $m = m_id$ *poichè ogni elemento del vettore* $\mathbf{z}_i$ *è un polinomio di grado al più* $d-1$.

4.6.4 σ_1, σ_2 in relazione con il challenge space $\mathcal{C}$ ed η

I parametri σ_1 e σ_2 determinano la deviazione standard delle distribuzioni gaussiane discrete utilizzate per campionare i vettori $\mathbf{y}_1$ e $\mathbf{y}_2$.

Essi controllano:

- la probabilità di accettazione nel rejection sampling;
- la norma dei vettori $\mathbf{z}_i$;
- la sicurezza rispetto al problema SIS.

In particolare, si ha che con alta probabilità:

$$\|\mathbf{z}_i\|_2 \leq \sigma_i\sqrt{2m_id}.$$

Valori troppo piccoli di σ_i aumentano il tasso di rigetto, mentre valori troppo grandi possono compromettere la sicurezza producendo vettori troppo lunghi. Inoltre in Lyubashevsky et al. [2022] il challenge space $\mathcal{C}$ viene così definito:

$$\mathcal{C} := \{c : \|c\|_\infty \leq \kappa, \sigma(c) = c, (\|\sigma_{-1}(c^k)c^k\|_1)^{1/2k} \leq \eta\}$$

dove $\sigma \in \{\sigma_1, \sigma_{-1}\}$ viene definito a seconda del protocollo. In particolare $\sigma_1(P(X)) = P(X)$ e $\sigma_{-1}(P(X)) = P(X^{-1})$. Si noti che nel caso $\sigma = \sigma_{-1}$ la richiesta che $\sigma(c) = c$ non restringe in modo ecessivo lo spazio delle challenge $\mathcal{C}$, poichè se si sceglie $c = c_0 + \sum_{i=1}^{d/2-1} c_i(X^i - X^{d-i})$ si ha $\sigma(c) = c$. Infatti:

$$\sigma_{-1}(c) = c_0 + \sum_{i=1}^{d/2-1} c_i(X^{-i} - X^{-d+i}) = c_0 + \sum_{i=1}^{d/2-1} c_i(-X^{d-i} + X^i) = c$$

dato che $X^d + 1 = 0$. I coefficienti da settare sarebbero dunque $d/2$ invece che d.

η è scelto sperimentalmente dimodochè la disequazione $(\|\sigma_{-1}(c^k)c^k\|_1)^{1/2k} \leq \eta$ sia soddisfatta con alta probabilità (si fa riferimento ad una probabilità ≈ 0.99).

κ è scelto tale che $\kappa < 2^{-3/2}\sqrt{q_1}$ (q_1 è il più piccolo fattore primo di q) per non solo assicurare l'invertibilità degli elementi $c \in \mathcal{C}$ su $\mathcal{R}_q$, ma anche della differenza $c_1 - c_2 = \bar{c} \in \mathcal{C}$. Per non infierire sulla difficoltà del problema Module-SIS, $|\mathcal{C}|$ deve essere esponenzialmente grande. La dipendenza da λ e solitamente dell'ordine di $2^{\Omega(\lambda)}$.

A questo punto è possibile settare le deviazioni standard σ_i: sapendo che $\mathbf{s}_i \in \mathcal{R}_q^{m_i}$ e avendo $\|\mathbf{s}_i\|_2 \leq \alpha_i$ allora $\sigma_i := \gamma\eta\alpha_i$ dove $\gamma > 0$ è il repetition rate. Se si ha anche che $\|\mathbf{s}_i\|_\infty \leq \nu$ si può settare $\sigma_i := \gamma\nu\eta\sqrt{m_i d}$.

4.6.5 Vincoli di sicurezza

La sicurezza del protocollo si basa sulla difficoltà del problema Module-SIS. In particolare, il bound

$$B = 8\eta\sqrt{2m_1 d\,\sigma_1^2 + 2m_2 d\,\sigma_2^2}$$

deve essere sufficientemente piccolo da rendere il problema Module-SIS intrattabile. Questo vincolo garantisce che non esistano soluzioni corte che possano essere facilmente trovate da un eventuale avversario.

Ne consegue che:

- le deviazioni standard σ_i non devono essere troppo grandi, altrimenti la distribuzione dei vettori $\mathbf{z}_i$ diventerebbe troppo larga, permettendo soluzioni più corte del previsto per Module-SIS, da cui $\sigma_i = \text{poly}(\lambda)$;
- il fattore η, che dipende dallo spazio delle challenge $\mathcal{C}$, deve essere scelto per controllare l'aumento della norma dei vettori durante le operazioni del protocollo;
- le dimensioni m_1 e m_2 devono essere sufficientemente grandi da garantire la hardness del problema Module-SIS, per cui m_i deve crescere almeno linearmente in λ.

4.6.6 Compromesso tra sicurezza ed efficienza

La scelta dei parametri richiede un compromesso tra sicurezza ed efficienza. In particolare:

- **Sicurezza:** parametri più grandi (ad esempio d, q, σ_i) aumentano la difficoltà dei problemi SIS/LWE, rendendo più difficile un attacco;
- **Efficienza:** parametri più piccoli riducono il costo computazionale e la dimensione delle prove da trasmettere;
- **Probabilità di accettazione:** valori di σ_i troppo piccoli aumentano il tasso di rigetto durante il rejection sampling, riducendo l'efficienza del protocollo.

In generale, il compromesso può essere sintetizzato come segue:

- aumentare d e q migliora la sicurezza senza influire direttamente sulla correttezza;

- aumentare σ_i migliora la probabilità che i vettori $\mathbf{z}_i$ rispettino i bound richiesti, ma riduce la sicurezza;

- ridurre η tramite la scelta dello spazio $\mathcal{C}$ aiuta a controllare la norma dei vettori e garantire correttezza, senza compromettere eccessivamente la sicurezza.

Parametro	**Ruolo**	**Impatto sulla sicurezza**	**Impatto su efficienza/correttezza**
d	Grado massimo dei polinomi	Maggiore d aumenta la sicurezza	Maggiore costo computazionale
q	Modulo dei polinomi	Maggiore q aumenta la sicurezza	Maggiore dimensione dei vettori
m_1, m_2	Dimensione dei vettori $\mathbf{y}_i$ e $\mathbf{s}_i$	Maggiore dimensione aumenta la difficoltà di Module-SIS	Maggiore costo computazionale
σ_1, σ_2	Deviazioni standard delle gaussiane discrete	Troppo grandi riducono la sicurezza, generando vettori più lunghi	Troppo piccoli aumentano il tasso di rifiuto
η	Fattore di incremento della norma tramite challenge	Troppo grande aumenta le norme e può ridurre sicurezza	Troppo piccolo migliora i bound ma aumenta i rifiuti
$\mathcal{C}$	Spazio delle challenge	Deve essere esponenzialmente grande per garantire sicurezza	Influenza il bound dei vettori e la probabilità di accettazione
γ	Repetition rate nel sampling	Maggiore γ aumenta la correttezza	Influenza la probabilità di accettazione e i costi computazionali

Tabella 4.2. Sintesi dei parametri principali e dei loro effetti su sicurezza ed efficienza.

4.7 Implementazione del protocollo ABDLOP

Per verificare sperimentalmente il protocollo ABDLOP 4.3.3 sono state realizzate due implementazioni Python del protocollo.

Gli esperimenti sono stati condotti su una macchina equipaggiata con un processore Intel Core 2 Duo E7600 a 3.06 GHz, dotato di 2 core e 2 thread, e di 4 GB di memoria RAM. Il sistema operativo utilizzato è Ubuntu 22.04.3 LTS, con kernel Linux $6.8.0-138-$generic e architettura $x86_64$.

Una prima implementazione verifica la stabilità dei parametri selezionati con dimensioni del problema uguali, pertanto sono stati scelti (m_1, m_2) dimodochè $m_1 = m_2 = m$.

Nella seconda implementazione, invece, una volta verificata la correttezza della parametrizzazione precedente verranno scelti $m_1 \neq m_2$ in modo da poter apprezzare al meglio la struttura a blocchi di dimensione differente del protocollo.

Entrambe le implementazioni comprendono tutte le componenti principali del protocollo: generazione delle matrici pubbliche, campionamento dei segreti, generazione delle challenge, rejection sampling e verifica della proof ottenuta.

4.7.1 Parametrizzazione

In questa sezione si descrive la scelta dei parametri utilizzati per l'implementazione del protocollo ABDLOP, al fine di garantire sicurezza a livello di $\lambda = 128$ bit e consentire un'efficienza computazionale accettabile. La selezione dei parametri è stata effettuata seguendo le linee guida stabilite in Lyubashevsky [2012] e Lyubashevsky et al. [2022], tenendo conto delle caratteristiche del protocollo, come la dimensione dei polinomi, la norma dei vettori segreti e la struttura del challenge space.

Il modulo q è scelto come $q = 12289$. La dimensione di q è sufficientemente grande da evitare overflow nei calcoli intermedi quando si eseguono operazioni su polinomi con coefficienti interi, garantendo così la correttezza delle riduzioni modulo q senza perdita di informazione. Esso rappresenta inoltre un connubio tra sicurezza ed efficienza: è sufficientemente grande da fornire uno spazio adeguato per i protocolli basati su reticoli, ma al tempo stesso abbastanza piccolo da consentire un'implementazione efficiente e non eccessivamente dispendiosa dal punto di vista computazionale.

Le dimensioni dei polinomi d sono fissate a λ e 2λ, coerentemente con quanto affermato nella sezione 4.6.

Nel contesto del protocollo originale, lo spazio delle challenge è definito:

$$\mathcal{C} := \left\{ c : \|c\|_\infty \leq \kappa,\ \sigma(c) = c,\ \left(\|\sigma_{-1}(c^k)c^k\|_1 \right)^{1/2k} \leq \eta \right\},$$

per le motivazionji già discusse in 4.6.4. Per motivi implementativi, si utilizza una semplificazione dello spazio delle challenge che sfrutta polinomi ternari sparsi. In particolare, ogni challenge $c \in \mathcal{C}$ è generata scegliendo un insieme di indici $\mathcal{S} \subseteq \{1, \ldots, d\}$ di cardinalità τ, e assegnando coefficienti in $\{-1, +1\}$ su tali posizioni, mentre tutti gli altri coefficienti sono posti a zero. Formalmente:

$$c_i = \begin{cases} \pm 1 & \text{se } i \in \mathcal{S},\ |\mathcal{S}| = \tau, \\ 0 & \text{altrimenti.} \end{cases}$$

In questo modo, la cardinalità dello spazio delle challenge diventa $|\mathcal{C}| = \binom{d}{\tau} \cdot 2^\tau$, poiché si sceglie prima il supporto e successivamente il segno di ciascun coefficiente non nullo.

Il parametro τ viene fissato in modo da garantire la sicurezza del protocollo, imponendo $|\mathcal{C}| \geq 2^\lambda$. Equivalentemente:

$$\binom{d}{\tau} \cdot 2^\tau \geq 2^\lambda.$$

Per ottenere una stima analitica di τ, si utilizza il bound standard sul coefficiente binomiale: $\binom{d}{\tau} \leq \left(\frac{ed}{\tau}\right)^\tau$.

Da cui segue la condizione sufficiente: $\left(\frac{2ed}{\tau}\right)^{\tau} \geq 2^{\lambda}$, che, prendendo il logaritmo in base 2, diventa:

$$\tau \cdot \log_2\left(\frac{2ed}{\tau}\right) \geq \lambda.$$

Nota 8. *Si supponga di avere un challenge space $\mathcal{C}$ generico e si cerchi un bound dell'operatore che, preso in input un generico vettore c, restituisce $\sigma_{-1}(c^k)c^k$. Si identifichi il polinomio c con il vettore dei coefficienti $c = \begin{pmatrix} c_0 \\ c_1 \\ \vdots \\ c_{d-1} \end{pmatrix} \in \mathbb{Z}^d$. La moltiplicazione per un polinomio $P(X)$ induce un operatore lineare*

$$T_P : \mathbb{Z}^d \to \mathbb{Z}^d, \qquad T_P(c) = M_P c,$$

dove M_P è la matrice definita da

$$(M_P)_{i,j} = \begin{cases} c_{i-j} & i \geq j, \\ -c_{i-j+d} & i < j, \end{cases} \qquad (\text{mod } d),$$

equivalentemente dalla relazione $X^d = -1$ in $\mathcal{R}$.

In particolare,

$$c^k = P(X)^k \iff M_P^k c^0, \quad c^0 = (1,0,\ldots,0)^T.$$

Siccome l'operatore σ_{-1} è così definito: $\sigma_{-1}(P(X)) = P(X^{-1})$, sia $\zeta = e^{2\pi i/(2d)}$ la radice $2d$-esima dell'unità, si ha:

$$\widehat{P}(k) = P(\zeta^k), \qquad \widehat{\sigma_{-1}(P(k))} = P(\zeta^{-k}).$$

Poiché $P(X)^k$ corrisponde a M_P^k e σ_{-1} commuta con le potenze:

$$\sigma_{-1}(P(X)^k) = \sigma_{-1}(P(X))^k,$$

si ha che

$$\sigma_{-1}(P(X)^k)P(X)^k = \left(P(X)P(X^{-1})\right)^k.$$

Si definisca ora l'operatore simmetrico:

$$Q(X) := P(X)P(X^{-1}), \qquad Q \longleftrightarrow M_Q.$$

Allora:

$$\sigma_{-1}(P(X)^k)P(X)^k = Q(X)^k.$$

Trasformando secondo Fourier si ha che $\widehat{Q}(k) = P(\zeta^k)P(\zeta^{-k})$.

Quindi:

$$\widehat{Q^k}(k) = \left(P(\zeta^k)P(\zeta^{-k})\right)^k.$$

Ritornando nel dominio dei coefficienti:

$$\sigma_{-1}(P(X)^k)P(X)^k = \mathcal{F}^{-1}\left(\left(P(\zeta^k)P(\zeta^{-k})\right)^k\right).$$

La condizione

$$\left(\|\sigma_{-1}(c^k)c^k\|_1\right)^{1/2k} \leq \eta$$

corrisponde alla stima:

$$\left(\|Q(X)^k\|_1\right)^{2k} \leq \eta, \qquad Q(X) = P(X)P(X^{-1}),$$

cioè un vincolo sulla crescita della norma ℓ_1 della potenza dell'operatore simmetrico associato a P. Siccome però si ha che per $c \in \mathcal{C}$ vale $\sigma_{-1}(c) = c$ ($M_P = M_P^T$) ed è possibile semplificare la condizione $(\|\sigma_{-1}(c^k)c^k\|_1)^{1/2k} \leq \eta$. Infatti:

$$\|\sigma_{-1}(c^k)c^k\|_1 = \|\sigma_{-1}(c)^k c^k\|_1 = \|c^{2k}\|_1 \leq \|c\|_1^{2k},$$

pertanto $\|\sigma_{-1}(c^k)c^k\|_1)^{1/2k} \leq \|c\|_1 \approx \eta$.

Nella definizione del challenge space originale in Lyubashevsky et al. [2022] si seleziona per $d = 128$ e $\kappa = 1$, $\eta = 27$. Nell'implementazione proposta il primo τ che soddisfa la relazione 4.7.1 è proprio $\tau = 28 = \|c\|_1$, che conferma l'idoneità di un challenge space ternario sparso.

Per $d = 128$ il primo τ che soddisfa 4.7.1 è $\tau = 28$, mentre per $d = 256$ si ha che $\tau = 22$.

E' stata impostata una limitazione sulla norma dei segreti $\mathbf{s}_i$ come in 4.6.4, in questo caso, $\|\mathbf{s}_i\|_\infty \leq \nu$ e ν è stato scelto pari a 2. Allora vale il bound sulla norma 2

$$\|\mathbf{s}_i\|_2 \leq \nu\sqrt{m_i d}.$$

Nel contesto del teorema 3.3.1, si identifica quindi il parametro $T := \nu\sqrt{m_i d}$.

La deviazione standard del rejection sampling viene scelta come

$$\sigma_i := \gamma\nu\rho\sqrt{m_i d} = \gamma\rho\, T_i = \alpha T_i$$

in accordo con Lyubashevsky et al. [2022] e con quanto riportato in 4.6.4. Pertanto il rapporto del teorema è dato da $\frac{\sigma_i}{T_i} = \gamma\rho$. Il teorema 3.3.1 richiede che $\sigma_i = \omega(T_i\sqrt{\log m_i})$, ovvero che esista una costante $k > 0$ tale che

$$\sigma_i \geq k\, T\sqrt{\log m_i}.$$

Sostituendo la definizione di σ_i, si ottiene la condizione equivalente

$$\gamma\rho\, T_i \geq k\, T_i\sqrt{\log m_i},$$

da cui segue $\gamma\rho \geq k\sqrt{\log m_i}$. Ne segue che una scelta naturale e asintoticamente valida è

$$\alpha = \Theta(\sqrt{\log m_i}).$$

In particolare, sotto tale condizione, il teorema garantisce l'esistenza di una costante $M = O(1)$ tale che l'algoritmo di rejection sampling produce campioni distribuiti con distanza statistica trascurabile rispetto alla Gaussiana discreta ideale $D^m_{\mathbf{0},\sigma}$, e con probabilità di accettazione circa $1/M$. Sostituendo $\alpha = \Theta(\sqrt{\log m})$ nella formula

$$M = \exp\left(\frac{12}{\alpha} + \frac{0.5}{\alpha^2}\right),$$

si ottiene

$$M = \exp\left(\frac{12}{\Theta(\sqrt{\log m})} + \frac{0.5}{\Theta(\log m)}\right)$$

da cui chiaramente $M = O(1)$. In particolare, si considerano valori

$$\rho \in \{1,2\}, \qquad \gamma \in \{4,6,8\},$$

in modo da esplorare significativo del parametro composto

$$\alpha \in \{4,6,8,12,16\}.$$

Questa scelta consente di osservare empiricamente il comportamento del protocollo sia nel regime più conservativo (α piccolo, maggiore tasso di rejection), sia nel regime più rilassato (α grande, maggiore vicinanza alla distribuzione ideale e $M \approx 1$), mantenendo comunque α in un intervallo compatibile con la condizione $\alpha = \omega(\sqrt{\log m})$ per il parametro m considerato.

4.7.2 Prima implementazione simmetrica: m_1=m_2

Questa prima implementazione prevede che $m_1 = m_2 = m$.

Per ciascuna configurazione dei parametri (d, m, γ, ρ) sono state eseguite 20 simulazioni indipendenti. In ogni simulazione vengono misurate:

- il tempo medio di esecuzione del prover;
- il tempo medio di esecuzione del verifier;
- il numero medio di tentativi richiesti dal rejection sampling;
- il tasso di accettazione della prova;
- la dimensione teorica della chiave pubblica;
- la dimensione teorica della chiave segreta;
- la dimensione del transcript;

Le matrici pubbliche $\mathbf{A}_1$ e $\mathbf{A}_2$,$\mathbf{R}_1$ e $\mathbf{R}_m$ sono generate uniformemente in $\mathcal{R}_q^{m\times m}$, mentre i vettori segreti $\mathbf{s}_1$ e $\mathbf{s}_2$ sono campionati m-dimensionalmente in modo uniforme con coefficienti interi limitati da ν. L'obiettivo dell'analisi sperimentale è individuare le configurazioni che garantiscono un elevato tasso di accettazione mantenendo contenute le dimensioni delle chiavi e del transcript.

Analisi dei risultati

In questa sezione si svolge un'analisi critica dei risultati ottenuti per le diverse configurazioni dei parametri (d, m, γ, ρ). In tutte le simulazioni il protocollo è stato verificato correttamente, ottenendo un tasso di accettazione pari al 100% per tutte le prove generate. Ciò conferma la correttezza dell'implementazione delle equazioni di verifica e della procedura di rejection sampling.

	$\alpha = 4$	$\alpha = 6$	$\alpha = 8$	$\alpha = 16$
$(d, m) = (128,8)$	170.00	47.00	12.50	19.75
$(d, m) = (128,12)$	384.00	55.75	9.75	16.50
$(d, m) = (256,8)$	262.75	100.00	11.00	33.50
$(d, m) = (256,12)$	1168.75	17.75	20.75	18.75

Tabella 4.3. Numero medio di tentativi del rejection sampling in funzione dei parametri (d, m) e $\alpha = \gamma\rho$.

Effetto dei parametri sul rejection sampling Come mostrato in tabella 4.7.2 si osserva una decrescita rapida del numero di tentativi al crescere di α, in accordo con la definizione della costante M:

$$M = \exp\left(\frac{12}{\alpha} + \frac{0.5}{\alpha^2}\right).$$

Infatti, all'aumentare di α, il valore di M tende rapidamente a 1, aumentando la probabilità di accettazione nel rejection sampling. Questo comportamento è dunque coerente con la parametrizzazione proposta. Un ruolo cruciale è svolto anche dal parametro $\sigma = \alpha\nu\sqrt{md}$. In particolare, al crescere di m e d, la norma dei vettori $\mathbf{z}_i$ scala come $\sqrt{md}$, rendendo più stringenti i vincoli di accettazione del rejection sampling. Tuttavia, tale crescita è compensata dal fattore α che, entrando sia in σ che in M, controlla simultaneamente ampiezza della distribuzione gaussiana e probabilità di accettazione.

I risultati sperimentali confermano dunque la coerenza tra il parametro α, la costante M e la stabilità del processo di rejection sampling.

Tempi di esecuzione I tempi di esecuzione del protocollo mostrano una dipendenza considerevole dalle dimensioni del reticolo e dalla procedura di rejection sampling. In particolare, la complessità computazionale è dominata da due fattori principali:

- la dimensione del reticolo, determinata dai parametri (d, m);
- il costo del rejection sampling, che introduce un numero variabile di tentativi prima dell'accettazione, influenzato dai parametri α (e quindi da M) e σ

Dal punto di vista computazionale, le operazioni dominanti sono le moltiplicazioni polinomiali di complessità $O(d^2)$ e le moltiplicazioni matrice-vettore in $\mathcal{R}_q$, che hanno

(d, m)	α	Tempi medi del prover (s)
(128,8)	4	11.27
(128,8)	6	3.18
(128,8)	8	0.84
(128,12)	4	54.86
(128,12)	6	7.81
(128,12)	8	1.37
(256,8)	4	33.51
(256,8)	6	12.74
(256,8)	8	1.39
(256,12)	4	329.17
(256,12)	6	5.15
(256,12)	8	5.96

Tabella 4.4. Tempo medio del prover in funzione di (d, m) e α.

complessità $O(m^2d^2)$. Tuttavia, il fattore più critico in termini di variabilità dei tempi è il rejection sampling.

In particolare, si osserva che l'aumento di d e m amplifica sia il costo delle operazioni, sia la varianza dei vettori gaussiani, rendendo più probabile il rifiuto dei campioni. Di conseguenza, il tempo di esecuzione del prover cresce più che linearmente rispetto alla dimensione del sistema, mentre il verifier resta molto efficiente.

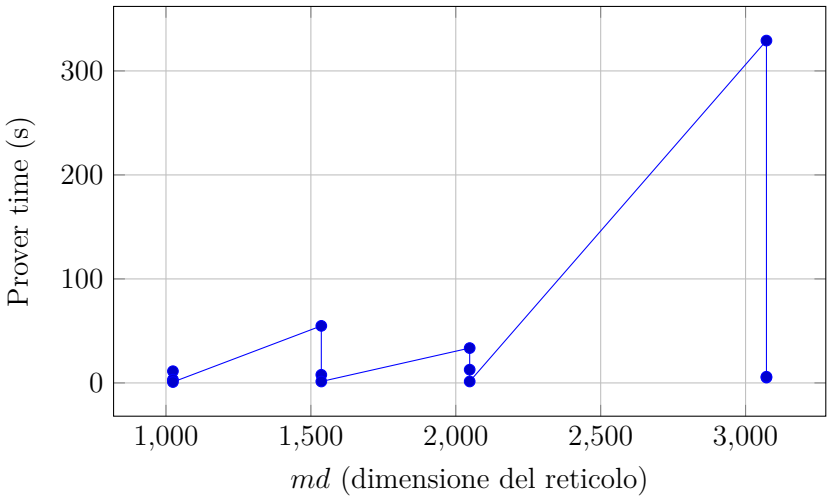


Figura 4.1. Scaling del tempo del prover rispetto alla dimensione del reticolo md.

Norme di $\mathbf{z}_i$ Le norme osservate dei vettori $\mathbf{z}_1$ e $\mathbf{z}_2$ risultano sempre significativamente inferiori ai bound teorici utilizzati durante la verifica. Questo comportamento è coerente

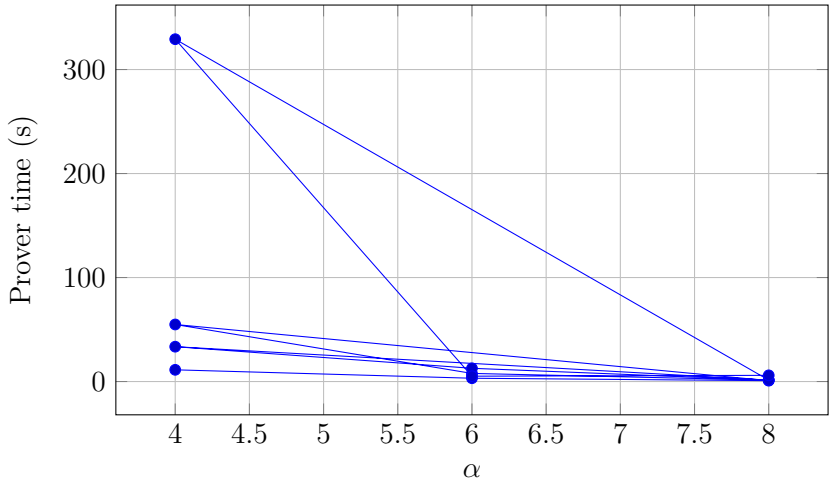


Figura 4.2. Riduzione del tempo del prover al crescere di α, dovuta alla diminuzione del costo medio del rejection sampling.

con l'analisi svolta in 4.6, poiché:

$$\|\mathbf{z}_i\|_2 = \|\mathbf{y}_i + c\mathbf{s}_i\|_2 \leq \|\mathbf{y}_i\|_2 + \|c\mathbf{s}_i\|_2 \leq \sigma_i\sqrt{m_i d} + \|c\|_2\|\mathbf{s}_i\|_2 \leq \sigma_i\sqrt{m_i d} + \sqrt{\tau}\nu\sqrt{m_i d}$$

. Siccome $\tau \in \{22{,}28\}$ e $\nu = 2$ si ha che $\sigma\sqrt{m_i d} + \sqrt{\tau}\nu\sqrt{m_i d} < \sigma\sqrt{2m_i d}$ per ogni $\sigma_i > \frac{\nu\sqrt{\tau}}{\sqrt{2}-1} \in \{22.65{,}25.55\}$. La disequazione è verificata per ogni σ_i proposto nelle simulazioni.

Pertanto certamente

$$\|\mathbf{z}_i\| \leq \sigma\sqrt{m_i d} + \sqrt{\tau}\nu \leq \sigma\sqrt{m_i d} + \sqrt{\tau}\nu\sqrt{m_i d} < \sigma\sqrt{2m_i d}.$$

I bound adottati sono dunque sufficientemente conservativi e il tasso di rifiuto è principalmente dovuto al rejection sampling .

Considerazioni sulla sicurezza Dal punto di vista della sicurezza, le configurazioni considerate mantengono invariati i parametri fondamentali che determinano la difficoltà del problema 2.1.14. I valori utilizzati nell'implementazione sono stati scelti in modo da soddisfare la condizione

$$|\mathcal{C}| \geq 2^{\lambda},$$

garantendo quindi che la probabilità di successo di un avversario che tenti di indovinare la challenge sia trascurabile.

Per $d = 128$ è stato utilizzato $\tau = 28$, mentre per $d = 256$ è stato utilizzato $\tau = 22$ secondo il criterio discusso in 4.7.1. Come si mostra con il grafico 4.3, in entrambi i casi si ottiene una cardinalità dello spazio delle challenge superiore a 2^{128}, mantenendo quindi il livello di sicurezza desiderato.

È importante osservare che i parametri γ ed ρ non modificano direttamente la sicurezza computazionale del protocollo. Essi influenzano principalmente i parametri σ e M

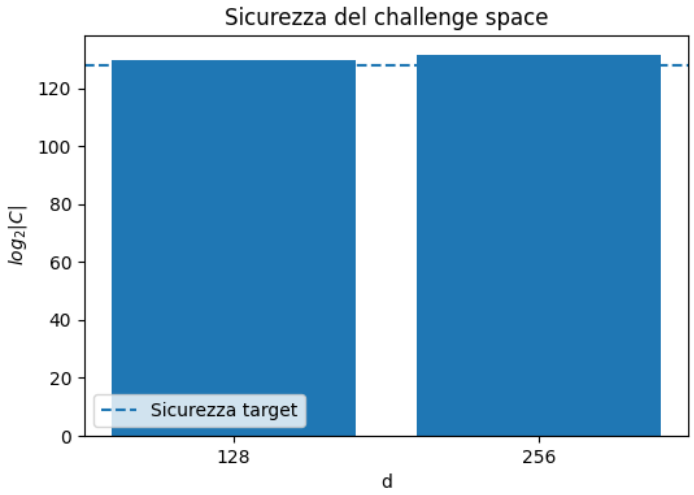


Figura 4.3. Sicurezza del challenge space $\mathcal{C}$ per $(d, \tau) = (128{,}28)$ e $(d, \tau) = (256{,}22)$.

che controllano l'efficienza del rejection sampling. L'aumento del grado dei polinomi da $d = 128$ a $d = 256$ comporta invece un incremento della dimensione del reticolo e, di conseguenza, una maggiore resistenza agli attacchi. Questo incremento di sicurezza è però accompagnato da un aumento dei costi computazionali e delle dimensioni delle chiavi.

Analisi della memoria Un aspetto fondamentale del protocollo riguarda la dimensione della proof prodotta.

La proof finale è composta dai seguenti elementi:

$$\pi = (\mathbf{z}_1, \mathbf{z}_2, \mathbf{w}, \mathbf{v}, c),$$

La dimensione totale della proof è quindi dominata dalla rappresentazione dei vettori $\mathbf{z}_1$ e $\mathbf{z}_2$ entrambi gaussiani con deviazione standard σ.

Di conseguenza, il numero di bit necessari per rappresentare ciascun coefficiente cresce come

$$O(\log_2 \sigma) = O\Big(\log(\sqrt{md})\Big),$$

mentre la lunghezza totale dei vettori è proporzionale a md.

Considerazioni analoghe possono essere svolte per $\mathbf{w}$ e $\mathbf{v}$.

c, invece, avendo supporto τ ha dimensione $\log_2 \binom{d}{\tau} + \tau \approx \tau \log_2(d/\tau) + \tau$ Ne risulta che la dimensione della proof può essere stimata complessivamente come

$$\text{size}(\pi) = O(md \log \sigma + \tau \log_2(d/\tau) + \tau) = O(md \log_2 \sqrt{md} + \tau \log_2(d/\tau) + \tau) \approx O(md \log_2(\sqrt{md}) + \tau).$$

Per i valori proposti si ha però che $\tau << md$, quindi l'aumento dei parametri (m, d) incrementa la dimensione della proof come $(md)^{1+\varepsilon}$.

Si riporta la dimensione della proof π ottenuta per ogni configurazione dei parametri nella tabella 4.7.2.

	Dimensione della proof π (KB)
$(d, m) = (128,8)$	83.18
$(d, m) = (128,12)$	177.06
$(d, m) = (256,8)$	166.89
$(d, m) = (256,12)$	355.07

Tabella 4.5. Media della dimensione delle proof per le configurazioni (d, m) con $m_1 = m_2 = m$.

4.7.3 Seconda implementazione asimmetrica: $m_1 \neq m_2$

La seconda implementazione del protocollo ABDLOP prevede una generalizzazione della struttura a blocchi del sistema, eliminando l'ipotesi di simmetria tra le dimensioni dei segreti. In particolare, si considera il caso in cui i parametri soddisfano $m_1 \neq m_2$, mantenendo invariati tutti gli altri parametri del protocollo rispetto alla prima implementazione.

Questa scelta consente di esplorare più accuratamente la natura a due componenti del protocollo, distinguendo esplicitamente il contributo dei due blocchi di segreti $\mathbf{s}_1 \in R_q^{m_1}$ e $\mathbf{s}_2 \in R_q^{m_2}$, che nella prima implementazione erano forzati a condividere la stessa dimensione.

$$m_1 \in \{8,12\} \quad m_2 \in \{20,24\}$$

La struttura delle matrici viene quindi generalizzata come segue:

$$\mathbf{A}_1 \in \mathcal{R}_q^{n \times m_1}, \qquad \mathbf{A}_2 \in \mathcal{R}_q^{n \times m_2}, \qquad \mathbf{B} \in \mathcal{R}_q^{l \times m_2},$$

$$\mathbf{R}_1 \in \mathcal{R}_q^{m_2 \times m_1}, \qquad \mathbf{R}_m \in \mathcal{R}_q^{m_2 \times l},$$

mentre i vettori segreti e il messaggio risultano definiti come:

$$\mathbf{s}_1 \in \mathcal{R}_q^{m_1}, \qquad \mathbf{s}_2 \in \mathcal{R}_q^{m_2}, \qquad \mathbf{m} \in \mathcal{R}_q^{l}.$$

La distribuzione dei parametri gaussiani e dei bound di norma viene adattata alla nuova struttura, mantenendo la stessa forma funzionale della prima implementazione:

$$\sigma_i = \gamma_i \rho \nu \sqrt{m_i d}, \qquad \|\mathbf{s}_i\|_2 \leq \nu \sqrt{m_i d}, \quad i \in \{1,2\}.$$

Dal punto di vista algoritmico, il prover e il verifier rimangono invariati nella loro struttura, ma tutte le operazioni di campionamento e combinazione lineare vengono ora effettuate su dimensioni differenti, introducendo una reale asimmetria nei costi computazionali delle due componenti del protocollo.

Questa modifica consente di analizzare in modo più fine l'impatto della disuguaglianza tra le dimensioni dei blocchi sui seguenti aspetti:

- tasso di accettazione del rejection sampling nei due rami separati;
- distribuzione delle norme dei vettori $\mathbf{z}_1$ e $\mathbf{z}_2$;

- costo computazionale sbilanciato tra le due componenti del prover;
- variazione della dimensione complessiva del transcript.

In questa seconda configurazione, tutte le altre grandezze del protocollo rimangono invariate rispetto alla prima implementazione, inclusi il modulo q, la dimensione del polinomio d, lo spazio delle challenge $\mathcal{C}$ e i parametri γ_i,ρ e τ.

Nota 9. *Si noti che avendo adesso diversi valori di m_i e quindi σ_i si ha la necessità di generare due differenti costanti di rejection sampling $M_i = \exp\left(\frac{12}{\alpha_i} + \frac{0.5}{\alpha_i^2}\right)$*

Alla luce dei risultati ottenuti durante la prima sperimentazione si ha che il giusto compromesso tra efficienza del rejection sampling e controllo di σ sia $\alpha \in \{6,8\}$. Si cerca quindi di ricavare la costante $C > 0$ ottimale per $\alpha_i = C\sqrt{\log m_i}$. Dai casi considerati si ha $\sqrt{\log 8} \approx 1.442$, $\sqrt{\log 12} \approx 1.58$, da cui segue una stima $C \approx 4.5$.

Sostituendo tale valore si ottiene la seguente regola pratica per la scelta di α_i:

$$\alpha_i = \gamma_i \rho \approx 4.5\sqrt{\log m_i}.$$

Per $m_1 \in \{8,12\}$ si ottiene $\alpha_1 \in \{6.5,7.1\}$ da cui una scelta naturale ricade in $\alpha_1 = 6.8$, mentre, Per $m_2 \in \{20,24\}$ si ottiene $\alpha_2 \in \{7.8,8.0\}$ da cui $\alpha_2 = 7.9$.

L'obiettivo dell'analisi sperimentale è quindi isolare esclusivamente l'effetto della disuguaglianza $m_1 \neq m_2$.

Analisi dei risultati

Anche in questo caso tutte le esecuzioni hanno prodotto una verifica corretta, con un tasso di accettazione pari al 100%, confermando la coerenza dell'implementazione.

Effetto dei parametri sul rejection sampling La Tabella 4.7.3 riporta il numero medio di tentativi del rejection sampling nelle diverse configurazioni asimmetriche.

(d, m_1, m_2)	σ_1	σ_2	Tentativi medi
(128,8,20)	435.20	799.42	27.05
(128,8,24)	435.20	875.72	31.40
(128,12,20)	533.01	799.42	23.05
(128,12,24)	533.01	875.72	28.60
(256,8,20)	615.47	1130.56	24.15
(256,8,24)	615.47	1238.46	40.65
(256,12,20)	753.79	1130.56	31.40
(256,12,24)	753.79	1238.46	25.50

Tabella 4.6. Numero medio di tentativi del rejection sampling nella configurazione asimmetrica.

Dai risultati si osserva che il numero di tentativi del rejection sampling rimane complessivamente stabile e contenuto.

La variazione dei parametri m_1 e m_2 non induce effetti drastici sulla distribuzione dei tentativi, suggerendo che la scelta dei parametri σ_1,σ_2 e che la selezione dei parametri α_1 e α_2 proposta mantiene il sistema in una regione di accettazione bilanciata.

Si osserva inoltre che l'effetto di m_2 tende a influenzare maggiormente la variabilità dei tentativi rispetto a m_1, coerentemente con il fatto che $m_2 \in [5/3m_1,3m_1]$.

Tempi di esecuzione La Tabella 4.7.3 riporta i tempi medi del prover nelle diverse configurazioni asimmetriche.

(d, m_1, m_2)	Tempi prover (s)	Tentativi medi
(128,8,20)	4.27	27.05
(128,8,24)	5.86	31.40
(128,12,20)	4.32	23.05
(128,12,24)	6.55	28.60
(256,8,20)	7.90	24.15
(256,8,24)	15.68	40.65
(256,12,20)	11.52	31.40
(256,12,24)	11.17	25.50

Tabella 4.7. Tempi medi del prover nella configurazione asimmetrica.

Dal punto di vista computazionale il tempo del prover rimane chiaramente correlato al numero medio di tentativi del rejection sampling.

L'aumento delle dimensioni del problema comporta un incremento ovvio generale del costo computazionale.

Nel complesso, i risultati confermano che il costo del prover è dominato dalla procedura di sampling, mentre le altre operazioni contribuiscono in modo secondario.

Dimensione della proof nel caso asimmetrico Nel caso della configurazione asimmetrica, la dimensione della proof risente direttamente della differenziazione tra i parametri m_1 e m_2, che introducono una struttura non uniforme nei vettori del transcript.

La proof continua ad essere definita come:

$$\pi = (\mathbf{z}_1, \mathbf{z}_2, \mathbf{w}, \mathbf{v}, c),$$

ma la sua dimensione complessiva dipende ora in modo distinto dalle due componenti del segreto.

Più nel dettaglio:

- la dimensione di $\mathbf{z}_1$ scala con $m_1 d$ e con il parametro σ_1;
- la dimensione di $\mathbf{z}_2$ scala con $m_2 d$ e con il parametro σ_2;

- i contributi di $\mathbf{w}$ e $\mathbf{v}$ rimangono proporzionali alla dimensione del sistema;
- il challenge c contribuisce in modo trascurabile rispetto ai vettori gaussiani.

Nel complesso, la dimensione della proof può essere approssimata come:

$$\text{size}(\pi) = O\big(m_1 d \log \sigma_1 + m_2 d \log \sigma_2 + \tau \log(d/\tau)\big),$$

e quindi la crescita è dominata dalle due componenti asimmetriche del sistema. La Tabella 4.8 riporta la dimensione media della proof nelle diverse configurazioni testate.

(d, m_1, m_2)	Chiave pubblica (KB)	Transcript (KB)	Dimensione della proof π (KB)
(128,8,20)	192.50	12.59	206.88
(128,8,24)	226.62	14.34	242.94
(128,12,20)	218.75	13.41	234.12
(128,12,24)	256.38	15.16	273.69
(256,8,20)	385.00	25.19	413.75
(256,8,24)	453.25	28.69	485.88
(256,12,20)	437.50	27.19	468.62
(256,12,24)	512.75	30.69	547.75

Tabella 4.8. Dimensione della proof nella configurazione asimmetrica.

Dai risultati si osserva che la dimensione della proof cresce principalmente con il parametro d, mentre l'effetto dell'asimmetria tra m_1 e m_2 introduce una variazione secondaria non dominante.

Il contributo di m_2 risulta ovviamente leggermente più rilevante rispetto a quello di m_1, poiché influenza direttamente la dimensione del vettore $\mathbf{z}_2$, che rappresenta la componente più pesante del transcript.

La configurazione asimmetrica non altera l'ordine di grandezza della proof rispetto al caso simmetrico, ma consente una maggiore flessibilità nella distribuzione dei costi tra le diverse componenti.

4.7.4 Conclusioni e scelta proposta dei parametri

Sulla base dei risultati sperimentali ottenuti nelle due implementazioni e sulle considerazioni svolte in 4.7.1, si propone la seguente configurazione dei parametri, che bilancia efficienza del rejection sampling, tempi del prover e dimensione della proof.

Parametro	Valore scelto	Valore derivato	Motivazione
d	128	–	Compromesso proposto tra sicurezza ed efficienza
m_1	12	–	Dimensione scelta per $\mathbf{s}_1$
m_2	20	–	Dimensione scelta per $\mathbf{s}_2$
ℓ	10	–	Dimensione scelta per $\mathbf{m}$
α_1	6.8	$\approx 4.5\sqrt{\log m_1}$	Calibrazione empirica del sampling
α_2	7.9	$\approx 4.5\sqrt{\log m_2}$	Calibrazione empirica del sampling
τ	28	$\min_{\tau\in\mathbb{N}}\left\{\tau \log_2\left(\frac{2ed}{\tau}\right) \geq \lambda\right\}$	Sicurezza del challenge space
σ_1	267	$\sigma_1 = \alpha_1\sqrt{m_1 d}$	4.6.4
σ_2	400	$\sigma_2 = \alpha_2\sqrt{m_2 d}$	4.6.4
M_1	5.9	$\exp\left(\frac{12}{\alpha_1} + \frac{0.5}{\alpha_1^2}\right)$	3.3.1
M_2	4.6	$\exp\left(\frac{12}{\alpha_2} + \frac{0.5}{\alpha_2^2}\right)$	3.3.1

Tabella 4.9. Scelta dei parametri. La configurazione è ottimizzata per flessibilità ed efficienza del rejection sampling.

Capitolo 5

Dalle prove non interattive alle credenziali anonime

Nel capitolo precedente è stato analizzato il protocollo di identificazione di Lyubashevsky e la sua evoluzione nello schema ABDLOP, mostrando come quest'ultimo consenta di ottenere una proof of knowledge per relazioni su reticoli con segreti di norma limitata.

Tuttavia, il protocollo ABDLOP, così come presentato, appartiene alla classe dei protocolli Σ ed è quindi interattivo: richiede infatti uno scambio di messaggi tra prover e verifier. In molte applicazioni crittografiche avanzate l'interattività rappresenta un limite pratico.

Pertanto, è necessario trasformare tali protocolli in versioni non interattive, mantenendo allo stesso tempo le proprietà di sicurezza fondamentali, quali:

- completezza;
- zero-knowledge;

Questo obiettivo può essere raggiunto attraverso la trasformazione di Fiat-Shamir 2.6.2, che consente di sostituire la challenge c del verifier con il risultato di una funzione hash modellata come oracolo random. In questo modo si ottengono le *Non-Interactive Zero-Knowledge Proofs* (NIZK), in cui il prover può generare autonomamente una prova verificabile pubblicamente.

Nel contesto dei reticoli, la costruzione di NIZK presenta ulteriori difficoltà rispetto al caso classico, dovute alla necessità di preservare la distribuzione dei campioni (ad esempio tramite rejection sampling) e di garantire che le prove risultanti siano efficienti.

In questo capitolo si formalizzano le principali NIZK proofs:

- la prova $\Pi_{\text{NIZK}}^{\text{ISIS}}$, che consente di dimostrare la conoscenza di soluzioni corte di istanze del problema ISIS;
- la prova $\Pi_{\text{NIZK}}^{\text{Com}}$, che permette di dimostrare in zero knowledge la corretta apertura di commitment ABDLOP e proprietà sui valori impegnati.

Queste costruzioni rappresentano i mattoni fondamentali per la realizzazione dello schemi di credenziali anonime ideato nel paper Bootle et al. [2023]. I dettagli costruttivi delle relative prove NIZK e delle dimostrazioni delle proprietà di sicurezza sono riportati nell'appendice A.

5.1 Proprietà richieste ai sistemi di credenziali anonime

Un sistema di credenziali anonime ha l'obiettivo di consentire ad un utente di dimostrare il possesso di determinati attributi senza rivelare la propria identità né informazioni ulteriori rispetto a quelle strettamente necessarie alla verifica.

A differenza dei tradizionali sistemi di autenticazione, nei quali il verificatore apprende direttamente l'identità dell'utente, le credenziali anonime richiedono il soddisfacimento simultaneo di diverse proprietà di sicurezza.

Correttezza. Un utente onesto in possesso di credenziali valide deve essere sempre in grado di produrre prove accettate dal verificatore.

Anonimato. L'esecuzione del protocollo non deve rivelare informazioni che consentano di identificare l'utente o ricostruire i suoi attributi privati.

Unlinkability. Due esecuzioni del protocollo effettuate dallo stesso utente non devono poter essere collegate tra loro da un osservatore esterno.

Non trasferibilità. Una credenziale emessa per un determinato utente non deve poter essere ceduta o utilizzata da terze parti.

Resistenza a falsificazioni. Un avversario non deve essere in grado di produrre credenziali valide senza aver partecipato correttamente alla fase di emissione.

Le proprietà sopra elencate richiedono strumenti crittografici avanzati in grado di combinare autenticità e privacy. In particolare, le prove zero-knowledge svolgono un ruolo centrale poiché consentono di dimostrare il possesso di informazioni segrete senza rivelarle.

Nel contesto post-quantistico, tali requisiti devono inoltre essere garantiti mediante assunzioni computazionali basate sui problemi 2.23 2.1.14 e 2.1.17.

5.2 Il problema ISIS_{bin}

Le proprietà di sicurezza delle prove $\Pi^{\mathrm{ISIS}}_{\mathrm{NIZK}}$ e $\Pi^{\mathrm{Com}}_{\mathrm{NIZK}}$ si basano su una variante del problema ISIS_f 2.1.17.

Definizione 5.2.1 (Il problema ISIS_{bin})**.** *Sia $t \in \mathbb{N}$ e sia $B \in \mathbb{Z}_q^{nd \times t}$ una matrice pubblica. Si definisce la funzione*

$$f(x) := \mathrm{Coeffs}^{-1}(B \cdot enc(x)) \in \mathcal{R}_q^n,$$

dove $\mathrm{enc}(x) \in \{0,1\}^t$ *rappresenta la decomposizione binaria di* $x-1$ *e la funzione* $Coeffs(x) : \mathcal{R}_q \to \mathbb{Z}_q$ *restituisce un vettore* $\vec{u}$ *dei coefficienti del polinomio* $x \in \mathcal{R}_q$ *. Ne segue naturalmente che* $N = 2^t$*.*

Si osserva che dimostrare la conoscenza di una preimmagine x *tale che*

$$f(x) = \mathbf{y}$$

è equivalente a dimostrare la conoscenza di un vettore binario $\vec{u} \in \{0,1\}^t$ *tale che:*

$$B\vec{u} = \mathrm{Coeffs}(\mathbf{y}).$$

Il problema si riduce a dimostrare la conoscenza di una soluzione binaria di un sistema lineare su $\mathbb{Z}_q$*.*

La matrice B *ha il ruolo di fornire flessibilità nella scelta dello spazio delle preimmagini, in particolare nella determinazione della dimensione* $N = 2^t$*. Tuttavia, essa non influisce in modo significativo sulla sicurezza del sistema. Si denota il problema ISIS associato a questa funzione con il nome di* $\mathrm{ISIS}_{\mathrm{bin}}$*.*

Nel caso del problema $\mathrm{ISIS}_{\mathrm{bin}}$, la Proof of Knowledge si riduce a provare l'esistenza di vettori che soddisfano specifici vincoli.

In particolare, si vuole dimostrare la conoscenza di:

- un vettore binario $\vec{u} \in \{0,1\}^t$;
- un vettore corto $\vec{s} \in \mathbb{Z}_{\hat{q}}^m$ tale che $\|\vec{s}\| \leq B_s$;
- un vettore corto $\vec{r} \in \mathbb{Z}_{\hat{q}}^{\ell_r}$ tale che $\|\vec{r}\| \leq B_r$,

tali che valga la relazione:

$$P\vec{s} = B\vec{u} + C \begin{bmatrix} \vec{m} \\ \vec{r} \end{bmatrix} \pmod{\hat{q}} \tag{5.1}$$

dove:

- $P \in \mathbb{Z}_{\hat{q}}^{n \times m}$;
- $B \in \mathbb{Z}_{\hat{q}}^{n \times t}$;
- $C \in \mathbb{Z}_{\hat{q}}^{n \times (\ell_m + \ell_r)}$;
- $\vec{m} \in \mathbb{Z}_{\hat{q}}^{\ell_m}$ è parte dello statement;
- i bound B_s, B_r fanno parte dello statement insieme a eventuali informazioni ausiliarie aux.

La relazione (5.1) costituisce esattamente il linguaggio per cui è definita la proof $\Pi_{\mathrm{NIZK}}^{\mathrm{ISIS}}$. Tale protocollo consente di dimostrare in modo non interattivo la conoscenza dei vettori $(\vec{u}, \vec{s}, \vec{r})$ che soddisfano:

- vincoli lineari modulo $\hat{q}$;
- vincoli di smallness sui vettori $\vec{s}$ e $\vec{r}$;
- vincolo combinatorio $\vec{u} \in \{0,1\}^t$.

Nota 10 (passaggio da $\mathcal{R}_q$ a $\mathbb{Z}_{\hat{q}}$)**.** *Se $q \mid \hat{q}$, è possibile ricondurre equazioni definite sull'anello $\mathcal{R}_q$ alla forma (5.1) tramite trasformazioni standard:*

- *rappresentando gli elementi di $\mathcal{R}_q$ come vettori in $\mathbb{Z}_q^d$ mediante matrici di moltiplicazione;*
- *effettuando un passaggio da $\mathbb{Z}_q$ a $\mathbb{Z}_{\hat{q}}$.*

Nota 11 (Ruolo del modulo $\hat{q}$)**.** *L'introduzione di un modulo esteso $\hat{q}$ è un passaggio tecnico fondamentale nel framework. Intuitivamente, $\hat{q}$ viene scelto più grande di q per i seguenti motivi:*

- ***stabilità delle distribuzioni:*** *consente di gestire correttamente il rejection sampling e preservare la distribuzione dei vettori anche dopo operazioni lineari;*
- ***controllo delle norme:*** *evita effetti che renderebbero difficile garantire i bound sui vettori corti;*
- ***riduzioni di sicurezza:*** *permette di collegare le istanze considerate ai problemi standard (come ISIS e Module-SIS) nelle dimostrazioni di sicurezza;*
- ***uniformità del framework:*** *consente di esprimere diverse relazioni (anche provenienti da anelli $\mathcal{R}_q$) come sistemi lineari su $\mathbb{Z}_{\hat{q}}$.*

In sintesi, il passaggio a $\mathbb{Z}_{\hat{q}}$ non modifica la natura del problema, ma semplifica l'analisi e la costruzione delle prove zero-knowledge.

5.3 Approximate Range Proofs

Le prove $\Pi_{\text{NIZK}}^{\text{Com}}$ e $\Pi_{\text{NIZK}}^{\text{ISIS}}$ consentono di dimostrare in zero knowledge la conoscenza di witness che soddisfano determinate relazioni lineari sui reticoli.

Tuttavia, la sicurezza dei protocolli basati su Module-SIS richiede che tali witness abbiano norma limitata. Ovvero, se $\mathbf{s} \in \mathcal{R}_q^m$ è un vettore segreto impegnato tramite ABDLOP, è spesso necessario dimostrare che $\|\mathbf{s}\|_2 \leq B$ per un opportuno bound pubblico B.

Il commitment ABDLOP da solo non fornisce alcuna garanzia su questa proprietà: il verificatore osserva solamente il commitment pubblico e non può verificare direttamente la norma del witness.

Per questo motivo il framework di Lyubashevsky Lyubashevsky et al. [2022] introduce le *Approximate Range Proofs*, che permettono di dimostrare in zero knowledge che il witness impegnato appartenga ad una regione limitata dello spazio. Le Approximate Range Proofs costituiscono un passaggio essenziale tra il commitment ABDLOP e le prove $\Pi_{\text{NIZK}}^{\text{ISIS}}$.

Infatti, l'estrattore associato alla proof of knowledge deve restituire un witness che sia non soltanto valido, ma anche sufficientemente corto da poter essere considerato una soluzione del problema Module-SIS.

Lo schema generale è quindi il seguente:

$$\text{Witness} \longrightarrow \text{ABDLOP Commitment} \longrightarrow \text{Range Proof} \longrightarrow \Pi_{\text{NIZK}}.$$

Nota 12. *Le proof* Π_{NIZK} *presentate non costituiscono un contributo originale di questa tesi e non vengono qui sviluppate o progettate ex novo. Esse sono richiamate esclusivamente come applicazione dello schema di commitment ABDLOP, al fine di mostrare come quest'ultimo possa essere impiegato nella costruzione di protocolli non interattivi per la dimostrazione di proprietà sui witness, quali ad esempio le range proof.*

La range proof certifica che il witness impegnato soddisfa i bound di norma richiesti dal protocollo, senza rivelarne il valore. Come mostrato nell'appendice A, grazie al teorema dei quattro quadrati ogni intero non negativo può essere rappresentato come somma di quattro quadrati.

Pertanto la condizione $B^2 - \|\mathbf{s}\|_2^2 \geq 0$ è equivalente all'esistenza di interi a_1, a_2, a_3, a_4 tali che $B^2 - \|\mathbf{s}\|_2^2 = a_1^2 + a_2^2 + a_3^2 + a_4^2$. Equivalentemente,

$$\|\mathbf{s}\|_2^2 + a_1^2 + a_2^2 + a_3^2 + a_4^2 = B^2.$$

Introducendo le variabili ausiliarie (a_1, a_2, a_3, a_4), il vincolo sulla norma viene trasformato nell'esistenza di un witness esteso

$$(\mathbf{s}, a_1, a_2, a_3, a_4)$$

che soddisfa l'equazione

$$\|\mathbf{s}\|_2^2 + a_1^2 + a_2^2 + a_3^2 + a_4^2 = B^2.$$

La verifica del bound sulla norma viene quindi ricondotta alla verifica dell'esistenza di una decomposizione quadratica, che può essere incorporata all'interno della relazione dimostrata dal protocollo zero-knowledge. Il termine "approximate" deriva dal fatto che la prova non certifica necessariamente il bound esatto.

Per ragioni di efficienza e per semplificare la costruzione delle prove, si accetta un piccolo fattore di rilassamento $\gamma > 1$.

In particolare il protocollo garantisce che se $\|\mathbf{s}\|_2 \leq B$ allora il verifier accetta, mentre se il verifier accetta, si può dedurre

$$\|\mathbf{s}\|_2 \leq \gamma B.$$

La differenza tra i due bound è controllata dai parametri del sistema e risulta trascurabile nelle applicazioni pratiche.

Capitolo 6

Framework per Credenziali Anonime

Questo capitolo è dedicato ad un'applicazione pratica delle zero-knowledge proofs $\Pi^{\mathsf{ISIS}}_{\mathsf{NIZK}}$ e $\Pi^{\mathsf{Com}}_{\mathsf{NIZK}}$ del capitolo A. Nel lavoro Bootle et al. [2023] viene proposto un framework per la costruzione di sistemi di credenziali anonime.

Nel framework, un utente ottiene una credenziale da un'autorità emittente e successivamente è in grado di dimostrarne il possesso senza rivelarne il contenuto.

In particolare, il framework utilizza:

- una prova $\Pi^{\mathsf{ISIS}}_{\mathsf{NIZK}}$, che consente di dimostrare la conoscenza di una soluzione corta a un'istanza del problema ISIS_f;
- una prova $\Pi^{\mathsf{Com}}_{\mathsf{NIZK}}$, che permette di dimostrare la correttezza di aperture di commitment costruiti mediante ABDLOP. Questa componente è utilizzata per assicurare la coerenza tra i valori impegnati e quelli utilizzati nelle altre parti del protocollo.

Le due prove vengono combinate per ottenere protocolli più complessi. In particolare, $\Pi^{\mathsf{Com}}_{\mathsf{NIZK}}$ viene utilizzata per legare i valori segreti a commitment pubblici, mentre $\Pi^{\mathsf{ISIS}}_{\mathsf{NIZK}}$ garantisce che tali valori soddisfino specifiche relazioni richieste dal sistema di credenziali.

6.1 Credenziali Anonime

Prima di analizzare il framework ideato in Bootle et al. [2023] si riportano le definizioni relative ad un sistema a credenziali anonime. Le entità coinvolte nello schema sono:

- **Issuer** $\mathcal{I}$, che certifica gli attributi degli holders (utenti);
- **Holder** $\mathcal{H}$, che riceve credenziali a partire dai suoi attributi e le utilizza per autenticarsi;
- **Verifier** $\mathcal{V}$, che esegue un protocollo con un holder e apprende un sottoinsieme dei suoi attributi.

Definizione 6.1.1 (**Credenziali anonime**). *Uno schema a* credenziali anonime *è una tripla di algoritmi*

$$(\mathsf{Init}, \mathsf{Issue}_{\mathcal{I},\mathcal{H}}, \mathsf{Verify}_{\mathcal{I},\mathcal{H}})$$

definiti come segue:

- $\mathsf{Init}(1^\lambda, 1^\ell) \to (ipk, isk)$*: l'algoritmo di inizializzazione prende in input il parametro di sicurezza* 1^λ *e la dimensione* $\ell \geq 1$ *(in unario) dell'array di attributi che l'issuer certificherà, e restituisce la chiave pubblica* ipk *e la chiave segreta* isk *dell'issuer.*

- $\mathsf{Issue}_{I,H}\langle(isk, \mathbf{attrs}', \mathbf{idx}), (ipk, \mathbf{attrs}, \mathbf{idx})\rangle \to \mathsf{cred}$ *oppure* $\perp$*: questo è il protocollo di emissione tra l'issuer* $\mathcal{I}$ *e l'utente* $\mathcal{H}$*.*

  L'utente ha in input:

  - *la chiave pubblica* ipk*;*
  - *un array di attributi* $\mathbf{attrs} := [a_i]_{i\in[\ell]}$*, con* $a_i \in \{0,1\}^*$*;*
  - *un array di indici* $\mathbf{idx} \in [\ell]^{\ell'}$*, con* $1 \leq \ell' \leq \ell$*, costituito da* ℓ' *indici distinti che individuano gli attributi da rivelare all'issuer.*

  L'issuer ha in input:

  - *la chiave segreta* isk*;*
  - *un array di attributi rivelati* $\mathbf{attrs}' := [a'_i]_{i\in\mathbf{idx}}$*;*
  - *gli indici* $\mathbf{idx}$*.*

  L'utente restituisce una credenziale cred *sugli attributi* $\mathbf{attrs}$ *se*

  $$\forall i \in \mathbf{idx}, \quad a'_i = a_i,$$

  e $\perp$ *altrimenti.*

- $\mathsf{Verify}_{H,V}\langle(\mathsf{cred}, \mathbf{attrs}, \mathbf{idx}), (ipk, \mathbf{attrs}', \mathbf{idx})\rangle \to \{0,1\}$*: questo è un protocollo tra un utente* $\mathcal{H}$ *e un verifier* $\mathcal{V}$*.*

  L'utente ha in input:

  - *una credenziale* cred*;*
  - *un array di attributi* $\mathbf{attrs} := [a_i]_{i\in[\ell]}$*;*
  - *un array di indici* $\mathbf{idx} \in [\ell]^{\ell'}$*, con* $1 \leq \ell' \leq \ell$*, che identifica gli attributi da rivelare.*

  Il verifier ha in input:

  - *la chiave pubblica* ipk *dell'autorità emittente;*
  - *un array di attributi rivelati* $\mathbf{attrs}' := [a'_i]_{i\in\mathbf{idx}}$*;*
  - *gli indici* $\mathbf{idx}$*.*

Il verificatore restituisce 1 *se* cred *è una credenziale valida emessa da* I *sugli attributi* $\mathbf{attrs}$ *tale che*

$$\forall i \in \mathbf{idx}, \quad a'_i = a_i,$$

e 0 *altrimenti.*

Definizione 6.1.2 (Correttezza)**.** *Uno schema di* credenziali anonime *si dice corretto se, per ogni* $\lambda \in \mathbb{N}$*,* $\ell \in \mathbb{N}$*,* $\ell', \ell'' \in [\ell]$*, per ogni* $\mathbf{idx}' \in [\ell]^{\ell'}$ *e* $\mathbf{idx}'' \in [\ell]^{\ell''}$ *con elementi a due a due distinti, e per ogni array di attributi*

$$\mathbf{attrs} := [a_0, \ldots, a_{\ell-1}], \quad \mathbf{attrs}' := [a'_0, \ldots, a'_{\ell'-1}], \quad \mathbf{attrs}'' := [a''_0, \ldots, a''_{\ell''-1}],$$

tali che

$$\forall i \in \mathbf{idx}',\ a_i = a'_i \quad e \quad \forall i \in \mathbf{idx}'',\ a_i = a''_i,$$

valgono le seguenti condizioni:

$$\mathbb{P}\Big[(ipk, isk) \leftarrow \mathsf{Init}(1^\lambda, \ell) : \bot \leftarrow \mathsf{Issue}_{I,H}\langle (isk, \mathbf{attrs}', \mathbf{idx}'), (ipk, \mathbf{attrs}, \mathbf{idx}')\rangle\Big] \leq negl(\lambda), \tag{6.1}$$

$$\mathbb{P}\Big[\begin{array}{l}(ipk, isk) \leftarrow \mathsf{Init}(1^\lambda, \ell),\\ \mathsf{cred} \leftarrow \mathsf{Issue}_{I,H}\langle (isk, \mathbf{attrs}', \mathbf{idx}'), (ipk, \mathbf{attrs}, \mathbf{idx}')\rangle : \\ \langle \bot, 0\rangle \leftarrow \mathsf{Verify}_{H,V}\langle (\mathsf{cred}, \mathbf{attrs}, \mathbf{idx}''), (ipk, \mathbf{attrs}'', \mathbf{idx}'')\rangle\end{array}\Big] \leq \mathsf{negl}(\lambda). \tag{6.2}$$

Mentre la correttezza garantisce il funzionamento del sistema in presenza di parti oneste, la costruzione di schemi di credenziali anonime richiede anche garanzie di privacy.

Tali proprietà si basano sulla possibilità di dimostrare il possesso di una credenziale senza rivelarla, problema che richiede la costruzione di protocolli zero-knowledge efficienti su reticoli.

Definizione 6.1.3 (Gioco di anonimato)**.** *Il gioco di anonimato si svolge tra un avversario* $\mathcal{A}$ *e un challenger* $\mathcal{C}$*.*

1. ***Setup.*** *L'avversario riceve il parametro di sicurezza* 1^λ *e produce la chiave pubblica dell'issuer* ipk*.*

2. ***Challenge.*** $\mathcal{A}$ *sceglie due array di attributi* $\mathbf{attrs_0}$ *e* $\mathbf{attrs_1}$ *tali che* $|\mathbf{attrs_0}| = l$ *e* $|\mathbf{attrs_1}| = l$*, e un array di indici* $\mathbf{idx}^* \in [l]^{l'}$*, con* $1 \leq l' \leq l$*, tale che*

$$\forall i \in idx^*,\ \mathbf{attrs}_{0,i} = \mathbf{attrs}_{1,i}.$$

Il challenger $\mathcal{C}$ *estrae* $b \in \{0{,}1\}$*.*

Successivamente, $\mathcal{A}$ *e* $\mathcal{C}$ *eseguono il protocollo*

$$\mathsf{Issue}_{I,H}\langle (\cdot,\cdot,\cdot,\cdot), (ipk, \mathbf{attrs}_b, idx^*)\rangle,$$

dove il challenger interpreta il ruolo dell' holder *con input l'array di attributi* $\mathbf{attrs}_b$ *e gli indici rivelati* $\mathbf{idx}^*$*, mentre l'avversario interpreta il ruolo dell'*issuer*.*

Se il protocollo ha successo, il challenger memorizza la credenziale generata cred_b*.*

Successivamente, l'avversario e il challenger eseguono il protocollo

$$\mathsf{Verify}_{H,V}\langle(\mathsf{cred}_b, \mathbf{attrs}_b, \mathbf{idx}^*), (\cdot,\cdot,\cdot)\rangle,$$

dove il challenger interpreta il ruolo del holder *con input la credenziale* cred_b*, l'array di attributi* $\mathbf{attrs}_b$ *e gli indici rivelati* $\mathbf{idx}^*$*, mentre l'avversario interpreta il ruolo del* verifier*.*

3. ***Risposta.*** *$\mathcal{A}$ restituisce b' e vince il gioco se $b' = b$.*

Definizione 6.1.4 (Anonimato)**.** *Uno schema di credenziali possiede la proprietà di anonimato se nessun avversario PPT può vincere il gioco di anonimato con probabilità maggiore di* $\frac{1}{2} + \mathrm{negl}(\lambda)$*.*

6.2 Il problema NTRU-ISIS$_f$

Nel capitolo 2 è stato introdotto il problema 2.1.17, che costituisce la base per la costruzione della proof $\Pi^{\mathsf{ISIS}}_{\mathsf{NIZK}}$. Tuttavia, tale problema presenta alcune limitazioni dal punto di vista della sicurezza.

In particolare, la generazione delle istanze nel problema ISIS_f è legata a procedure di *preimage sampling* che utilizzano trapdoor. Questo rende difficile formalizzare un esperimento di sicurezza in cui il challenger possa verificare in modo efficiente le soluzioni prodotte da un avversario.

Un'ipotesi crittografica si dice *falsificabile* se esiste un esperimento efficiente che consente di generare istanze e verificare le soluzioni.

Nel caso di ISIS_f, tale proprietà non è soddisfatta, poiché la distribuzione delle istanze dipende dalla presenza di una trapdoor utilizzata nel processo di campionamento delle preimmagini.

Per superare questa limitazione, viene introdotta una variante del problema, denominata $\mathsf{NTRU} - \mathsf{ISIS}_f$, in cui il challenger dispone esplicitamente di una trapdoor che consente di generare preimmagini corte.

Si definisce ora un reticolo NTRU.

Definizione 6.2.1 (Reticolo NTRU)**.** *Sia $d \in \mathbb{N}$ una potenza di due e sia $q \geq 2$ un intero. Si considerino gli anelli $\mathcal{R}$ e $\mathcal{R}_q$ Siano $f, g \in \mathcal{R}$ tali che f sia invertibile in $\mathcal{R}_q$. Si definisca*

$$h := g \cdot f^{-1} \in \mathcal{R}_q.$$

Il reticolo NTRU *associato a h e q è definito come*

$$\Lambda_{h,q} := \{(u, v) \in \mathcal{R}^2 \ : \ u + v \cdot h \equiv 0 \pmod{q}\}.$$

Si osserva che $\Lambda_{h,q}$ è un reticolo a rango pieno di dimensione $2d$ su $\mathbb{Z}$. Una base per $\Lambda_{h,q}$ può essere descritta tramite la matrice

$$\mathbf{A}_{h,q} := \begin{pmatrix} -\mathrm{rot}(h) & I_d \\ qI_d & 0 \end{pmatrix} \in \mathbb{Z}^{2d \times 2d},$$

dove

$$\mathrm{rot}(h) := \begin{pmatrix} h_0 & h_{d-1} & h_{d-2} & \cdots & h_1 \\ h_1 & h_0 & h_{d-1} & \cdots & h_2 \\ h_2 & h_1 & h_0 & \cdots & h_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ h_{d-1} & h_{d-2} & h_{d-3} & \cdots & h_0 \end{pmatrix}.$$

è la matrice di rotazione associata al polinomio h.

Nota 13. *La matrice* $\mathrm{rot}(h)$ *rappresenta l'operazione di moltiplicazione per* h *nell'anello* $\mathcal{R}_q$*. Più precisamente, per ogni* $v \in \mathcal{R}_q$ *vale*

$$\mathrm{rot}(h) \cdot \mathrm{coeff}(v) = \mathrm{coeff}(h \cdot v),$$

dove $\mathrm{coeff}(\cdot)$ *denota il vettore dei coefficienti rispetto alla base* $\{1, X, \ldots, X^{d-1}\}$*.*

Nota 14. *Si noti inoltre che, definita la matrice* $\mathbf{A} = \begin{pmatrix} I_d \\ \mathrm{rot}(h) \end{pmatrix}$*, si ha che* $\mathbf{A}_{h,q} \cdot \mathbf{A} = -\mathrm{rot}(h) + \mathrm{rot}(h) = \mathbf{0} \in \mathbb{Z}_q^d$*. Dunque denotando con* $\mathbf{a}_{i,h,q} \in \mathbb{Z}^{2d} \quad \forall i \in [2d]$ *le righe di* $\mathbf{A}_{h,q}$ *si ha che* $\mathbf{a}_{i,h,q} \subseteq \mathrm{Ker}(\mathbf{A}^{\mathrm{T}})$*.*

Si richiama ora il seguente risultato fondamentale.

Lemma 6.2.1. *Esiste un algoritmo efficiente*

$$\mathsf{NTRU.TrapGen}(q, d)$$

che restituisce una coppia $(h, \mathbf{B})$ *tale che:*

- $h \in \mathcal{R}_q$*;*
- $\mathbf{B}$ *è una base di* $\Lambda_{h,q}$*;*
- *la base* $\mathbf{B}$ *è* corta*, ovvero soddisfa*

$$\|\widetilde{\mathbf{B}}\| \leq 1.17\sqrt{q},$$

dove $\widetilde{B}$ *denota l'ortogonalizzazione di Gram-Schmidt di* $\mathbf{B}$*.*

La base corta $\mathbf{B}$ funge da *trapdoor* per il reticolo $\Lambda_{h,q}$. La disponibilità di una trapdoor consente di campionare efficientemente preimmagini corte rispetto a vincoli lineari modulo q, utilizzando il framework di *Gaussian Preimage Sampling* (GPV).

Lemma 6.2.2 (Campionamento con trapdoor NTRU)**.** *Sia $\lambda \in \mathbb{N}$ il parametro di sicurezza e sia $\varepsilon = 2^{-\lambda/(4d)}$. Esiste un algoritmo PPT*

$$\mathsf{GSampler}$$

che, dati in input:

- $(h, \mathbf{B}) \leftarrow \mathsf{NTRU.TrapGen}(q, d)$*;*
- *un vettore $a \in \mathcal{R}_q^m$;*
- *una deviazione standard $\sigma > 0$;*
- *un target $c \in \mathcal{R}_q$;*

restituisce un vettore $(u, v, \mathbf{w}) \in \mathcal{R}^{m+2}$ tale che:

$$u + v \cdot h + \mathbf{a}^T \mathbf{w} \equiv c \pmod{q},$$

e la distribuzione di $(u, v, \mathbf{w})$ è statisticamente vicina a una gaussiana discreta centrata in $\mathbf{0}$, ossia

$$\Delta\big(D_{\mathbf{0},\sigma}^{m+2}, \mathsf{GSampler}(h, a, \mathbf{B}, \sigma, c)\big) \leq 2^{-\lambda},$$

a condizione che

$$\sigma \geq 1.17\sqrt{q} \cdot \eta'_\varepsilon(\mathbb{Z}),$$

dove $\eta'_\varepsilon(\mathbb{Z}) \approx \frac{1}{\pi}\sqrt{\frac{1}{2}\ln(2 + 2/\varepsilon)}$.

Ora è possibile definire il problema $\mathsf{NTRU} - \mathsf{ISIS}_f$.

Definizione 6.2.2 (Problema $\mathsf{NTRU} - \mathsf{ISIS}_f$)**.** *Sia $pp := (q, d, m, N, k, B)$ un insieme di parametri dipendenti dal parametro di sicurezza λ. Sia $f : [N] \to \mathcal{R}_q$ una funzione computazionalmente efficiente.*

L'esperimento $\mathsf{Exp}_{pp}^{\mathsf{NTRU\text{-}ISIS}_f}(A)$ è definito come segue:

Algoritmo 13: $\mathsf{Exp}_{pp}^{\mathsf{NTRU\text{-}ISIS}_f}(\mathcal{A})$

Input: *Parametri pubblici $pp = (q, d, m, N, k, f, B)$*
Output: 1 *se l'avversario vince,* 0 *altrimenti*
$(a_1, \mathbf{B}) \leftarrow \mathsf{NTRU.TrapGen}(q, d)$*;*
$a_2 \xleftarrow{\$} \mathcal{R}_q^m$*;*
$\mathbf{A} \leftarrow [a_1 \,\|\, a_2^T \,\|\, 1]$*;*
for $i \in [k]$ **do**
  $x_i \xleftarrow{\$} [N]$*;*
  $\mathbf{s}_i \leftarrow \mathsf{GSampler}(a_1, a_2, \mathbf{B}, \sigma, f(x_i))$*;*
$(x^*, \mathbf{s}^*) \leftarrow \mathcal{A}\big(\mathbf{A}, (x_i, \mathbf{s}_i)_{i\in[k]}\big)$*;*
if $\mathbf{A}\mathbf{s}^* = f(x^*) \ \wedge \ \|\mathbf{s}^*\| \leq B \ \wedge \ (x^*, \mathbf{s}^*) \notin \{(x_i, \mathbf{s}_i)\}_{i\in[k]}$ **then**
  return 1*;*
else
  return 0*;*

6.3 Introduzione al framework

All'interno di questa sezione si introducono le componenti principali degli algoritmi dello schema vero e proprio.

- $f(x) = \text{Coeffs}^{-1}(B \cdot \text{enc}(x)) \in \mathcal{R}_q^n$ la funzione definita all'inizio del capitolo A, dove $N = 2^t$, $x \in [1, N+1) \cap \mathbb{Z}$ e $B \in \mathbb{Z}_q^{d \times m}$;
- $\Pi_{\mathsf{NIZK}}^{\mathsf{ISIS}} := (\mathsf{Prove}_H^{\mathsf{ISIS}}, \mathsf{Verify}_H^{\mathsf{ISIS}})$ presentata nella sezione A.1;
- $\Pi_{\mathsf{NIZK}}^{\mathsf{Com}} := (\mathsf{Prove}_H^{\mathsf{Com}}, \mathsf{Verify}_H^{\mathsf{Com}})$ presentata nella sezione A.2;
- quattro funzioni hash $\mathsf{H}_{\mathsf{crs}}, \mathsf{H}_{\mathsf{Com}}, \mathsf{H}_{\mathsf{ISIS}}, \mathsf{H}_M$ modellate come oracoli random nella dimostrazione di sicurezza. In particolare:
  - La funzione $\mathsf{H}_{\mathsf{crs}}$ è una funzione hash utilizzata esclusivamente per generare i crs. In particolare, si assume che
    $$\mathsf{H}_{\mathsf{crs}}(0) = (\mathsf{crs}_{\mathsf{ISIS}}, \mathsf{crs}_{\mathsf{Com}}),$$
    dove:
    * $\mathsf{crs}_{\mathsf{ISIS}}$ è utilizzato dal sistema $\Pi_{\mathsf{NIZK}}^{\mathsf{ISIS}}$;
    * $\mathsf{crs}_{\mathsf{Com}}$ è utilizzato dal sistema $\Pi_{\mathsf{NIZK}}^{\mathsf{Com}}$.
  - Le funzioni $\mathsf{H}_{\mathsf{Com}}$ e $\mathsf{H}_{\mathsf{ISIS}}$ sono impiegate internamente nei rispettivi sistemi
    $$\Pi_{\mathsf{NIZK}}^{\mathsf{Com}} \quad \text{e} \quad \Pi_{\mathsf{NIZK}}^{\mathsf{ISIS}}.$$
  - La funzione H_M è utilizzata per comprimere gli attributi in un vettore di lunghezza h. Formalmente,
    $$\mathsf{H}_M : \{0,1\}^* \to \{-\psi_\infty, \dots, \psi_\infty\}^h.$$
- $\mathsf{NTRU.TrapGen}(q, d)$: algoritmo PPT definito in 6.2.1 che genera una coppia $(h, \mathbf{B})$, dove $h \in \mathcal{R}_q$ e $\mathbf{B}$ è una base corta del reticolo $\Lambda_{h,q}$;
- $\mathsf{GSampler}$: algoritmo di campionamento di preimmagini definito in 6.2.2 che, dato $(h, \mathbf{B})$, consente di estrarre vettori corti distribuiti secondo una gaussiana discreta e che soddisfano vincoli lineari modulo q.

Si riporta inoltre la tabella dei parametri principali coinvolti.

6.4 Il sistema a credenziali anonime

Generazione delle chiavi. L'algoritmo di generazione delle chiavi procede come segue:

- esegue
  $$(a_1, \mathbf{B}) \leftarrow \mathsf{NTRU.TrapGen}(q, d);$$

Parametro	Descrizione
$\hat{d}$	Grado dell'anello $\hat{\mathcal{R}}$ utilizzato nei protocolli $\Pi^{\mathsf{ISIS}}_{\mathsf{NIZK}}$ e $\Pi^{\mathsf{Com}}_{\mathsf{NIZK}}$
q	Modulo principale
$\hat{q}_1$	Modulo per $\Pi^{\mathsf{Com}}_{\mathsf{NIZK}}$, tale che $q \mid \hat{q}_1$
$\hat{q}_2$	Modulo per $\Pi^{\mathsf{ISIS}}_{\mathsf{NIZK}}$, tale che $q \mid \hat{q}_2$
$N = 2^t$	Dimensione del dominio della funzione f
$t = \log(N)$	lunghezza del vettore $\vec{u} \in \{0,1\}^t$
h	Lunghezza dell'output della funzione hash H_M
l	numero di attributi, tale che $d \mid lh$
m	lunghezza di a_2
ℓ_r	Lunghezza del vettore casuale $\mathbf{r}$ generato dall' holder
ℓ_m	lh/d
ψ_∞	Norma infinito di $\mathbf{r}$ e dell'output della funzione hash H_M
σ	le preimmagini vengono campionate da $D^{md+2d}_{\mathbf{0},\sigma}$

Tabella 6.1. Parametri del sistema di credenziali anonime

- campiona uniformemente
$$\mathbf{a}_2 \leftarrow \mathcal{R}_q^m;$$
- campiona
$$\mathbf{c}_0 \leftarrow \mathcal{R}_q^{\ell_m}, \qquad \mathbf{c}_1 \leftarrow \mathcal{R}_q^{\ell_r}.$$

La chiave pubblica e la chiave segreta sono definite rispettivamente come

$$\mathsf{ipk} := (a_1, \mathbf{a}_2, \mathbf{c}_0, \mathbf{c}_1), \qquad \mathsf{isk} := \mathbf{B}.$$

Credential Issuing. Sia $\mathcal{H}$ un holder con input $(\mathsf{ipk}, \mathbf{attrs}, \mathsf{idx})$, dove

$$\mathbf{attrs} = (a_1, \ldots, a_l).$$

L'holder costruisce il vettore polinomiale

$$\mathbf{m} := \mathrm{Coeffs}^{-1}\big(\mathsf{H}_M(a_1), \ldots, \mathsf{H}_M(a_l)\big) \in \mathcal{R}_q^{\ell_m}.$$

Successivamente:

- campiona
$$\mathbf{r} \leftarrow S_{\psi_\infty}^{\ell_r};$$
- calcola il commitment
$$u := \mathbf{c}_0^T \mathbf{m} + \mathbf{c}_1^T \mathbf{r} \in \mathcal{R}_q.$$

L'holder dimostra quindi la conoscenza di un'apertura valida del commitment u. Per utilizzare il sistema $\Pi^{\mathsf{ISIS}}_{\mathsf{NIZK}}$, si traduce la relazione in $\mathbb{Z}_q$.

Siano:

$$P := [\mathrm{rot}(\mathbf{c}_0^T)_{\overline{\mathsf{idx}}} \parallel \mathrm{rot}(\mathbf{c}_1)], \quad \vec{s} := (\mathrm{Coeffs}(\mathbf{m})_{\overline{\mathsf{idx}}}, \mathrm{Coeffs}(\mathbf{r})),$$
$$\vec{u} := \mathrm{Coeffs}(u) - \mathrm{rot}(\mathbf{c}_0^T)_{\mathsf{idx}} \cdot \mathrm{Coeffs}(\mathbf{m})_{\mathsf{idx}}.$$

Si verifica che:

$$P\vec{s} = \vec{u}.$$

Elevando modulo $\hat{q}_1$, si ottiene la relazione:

$$\frac{\hat{q}_1}{q} P\vec{s} = \frac{\hat{q}_1}{q}\vec{u} \pmod{\hat{q}_1}.$$

L'holder invia quindi (u, π) all'issuer, dove π è una prova $\Pi^{\mathsf{Com}}_{\mathsf{NIZK}}$.

L'issuer $\mathcal{I}$, ricevuti (u, π) e il set di indici idx:

- verifica la prova π;
- campiona
$$x \leftarrow [N];$$
- calcola
$$s \leftarrow \mathsf{GSampler}(a_1, \mathbf{a}_2, \mathbf{B}, \sigma, f(x) + u),$$
tale che
$$[a_1 \mid \mathbf{a}_2^T \mid 1]\mathbf{s} = f(x) + u.$$
per la definizione 6.2.2.

L'issuer restituisce $(x, \mathbf{s})$ all'holder.
L'holder verifica che $\mathbf{s}$ sia corto e soddisfi la relazione sopra; in tal caso, la credenziale è

$$\mathsf{cred} := (\mathbf{s}, \mathbf{r}, x).$$

Verifica. Nel protocollo di verifica tra $\mathcal{H}$ e $\mathcal{V}$, l'holder vuole dimostrare la conoscenza di:

- un vettore corto $\mathbf{s} \in \mathcal{R}_q^{m+2}$;
- un elemento $x \in [N]$;
- un vettore corto $\mathbf{r} \in \mathcal{R}_q^{\ell_r}$;
- le componenti segrete di $\mathbf{m}$;

tali che

$$[a_1 \mid \mathbf{a}_2^T \mid 1]\mathbf{s} = f(x) + \mathbf{c}_0^T\mathbf{m} + \mathbf{c}_1^T\mathbf{r}.$$

Passando alla rappresentazione matriciale, si ottiene:

$$\mathrm{rot}([a_1 \mid \mathbf{a}_2^T \mid 1]) \cdot \mathrm{Coeffs}(\mathbf{s}) = \mathrm{Coeffs}(f(x)) + \mathrm{rot}(\mathbf{c}_0^T)\mathrm{Coeffs}(\mathbf{m}) + \mathrm{rot}(\mathbf{c}_1)\mathrm{Coeffs}(\mathbf{r}).$$

Si definiscono:

$$P := \mathrm{rot}([a_1 \mid \mathbf{a}_2^T \mid 1]), \quad C := \left[\mathrm{rot}(\mathbf{c}_0^T)_{\mathsf{idx}} \mid \mathrm{rot}(\mathbf{c}_0^T)_{\overline{\mathsf{idx}}} \mid \mathrm{rot}(\mathbf{c}_1)\right],$$

$$\vec{s} := \mathrm{Coeffs}(s), \quad \vec{u} := \mathrm{enc}(x), \quad \vec{r} := \left(\mathrm{Coeffs}(\mathbf{m})_{\overline{\mathsf{idx}}}, \mathrm{Coeffs}(\mathbf{r})\right).$$

Si ottiene quindi la relazione:

$$P\vec{s} = B\vec{u} + C\begin{bmatrix}\vec{m}\\ \vec{r}\end{bmatrix}.$$

Con alta probabilità valgono i bound:

$$\|\vec{s}\| \le \sigma\sqrt{(m+2)d}, \qquad \|\vec{r}\| \le \psi_\infty\sqrt{(h\cdot|\overline{\mathsf{idx}}| + \ell_r)d}, \qquad \vec{u} \in \{0,1\}^t.$$

Elevando modulo $\hat{q}_2$, si considera:

$$\frac{\hat{q}_2}{q}P\vec{s} = \frac{\hat{q}_2}{q}B\vec{u} + \frac{\hat{q}_2}{q}C\begin{bmatrix}\vec{m}\\ \vec{r}\end{bmatrix} \pmod{\hat{q}_2}.$$

L'holder genera quindi una prova

$$\pi \leftarrow \Pi^{\mathsf{ISIS}}_{\mathsf{NIZK}}$$

per tale relazione, rispetto all'istanza pubblica

$$\left(\frac{\hat{q}_2}{q}P, \frac{\hat{q}_2}{q}C, \vec{m}, \frac{\hat{q}_2}{q}B, (\sigma\sqrt{(m+2)d}, \psi_\infty\sqrt{(h\cdot|\overline{\mathsf{idx}}| + \ell_r)d}), \mathsf{idx}\right).$$

Infine, il verificatore $\mathcal{V}$:

- ricostruisce l'istanza pubblica;
- verifica la prova π.

Capitolo 7

Conclusioni e sviluppi futuri

L'obiettivo di questa tesi è stato lo studio di protocolli zero-knowledge post-quantistici basati sui reticoli per la realizzazione di sistemi di credenziali anonime resistenti ad avversari quantistici.

Dopo aver introdotto i principali strumenti teorici della crittografia su reticoli, tra cui i problemi SIS, LWE e le relative varianti, sono state richiamate le nozioni fondamentali di proof of knowledge, protocolli Σ e trasformazione di Fiat-Shamir, necessarie per la comprensione delle costruzioni successive. Nel corso della tesi è stato analizzato il legame tra tali costruzioni e la progettazione di sistemi orientati alla privacy.

Il contributo principale della tesi è rappresentato dallo studio del commitment scheme ABDLOP. Tale schema costituisce uno dei più importanti sviluppi recenti nell'ambito delle prove zero-knowledge sui reticoli, poiché consente di costruire proof of knowledge efficienti per relazioni lineari e vincoli di norma definiti sull'anello $\mathcal{R}_q$. Le proprietà di hiding, binding, honest-verifier zero-knowledge e proof of knowledge rendono tale schema un elemento centrale nella costruzione delle prove Π^{ISIS}NIZK e Π^{Com}NIZK, le cui proprietà sono state analizzate approfonditamente nell'appendice A e successivamente utilizzate nel framework per credenziali anonime di Bootle et al. [2023].

Una parte importante del lavoro è stata dedicata all'analisi teorica della parametrizzazione del protocollo. La selezione dei parametri nei protocolli basati su reticoli costituisce infatti un problema particolarmente delicato, poiché richiede di bilanciare requisiti spesso contrastanti di sicurezza, correttezza ed efficienza computazionale. Sono stati studiati in dettaglio il ruolo dei bound B_i, delle deviazioni standard σ_i, delle costanti M_i del rejection sampling e il ruolo cruciale del challenge space $\mathcal{C}$ sulle proprietà del protocollo.

Accanto allo studio teorico, la tesi ha incluso una componente sperimentale mediante lo sviluppo di un'implementazione in Python del protocollo ABDLOP. L'analisi sperimentale ha consentito di confrontare differenti configurazioni dei parametri e di valutare empiricamente il loro impatto sui tempi di esecuzione e sulla dimensione delle proof generate.

È opportuno sottolineare che i risultati sperimentali ottenuti non hanno la pretesa di individuare una parametrizzazione ottimale o definitiva del protocollo. La scelta dei parametri nei sistemi crittografici basati su reticoli dipende infatti dal livello di sicurezza

desiderato, dai requisiti applicativi e dai vincoli implementativi. L'obiettivo dell'implementazione sviluppata in questa tesi è stato quello di proporre una metodologia di parametrizzazione e di verificarne sperimentalmente il comportamento, mostrando come sia possibile ottenere configurazioni che risultino utilizzabili in pratica e che rappresentino un valido punto di partenza per un impiego del protocollo.

I risultati ottenuti mostrano che il parametro che influenza maggiormente la dimensione complessiva del protocollo è il grado del reticolo d. In particolare, il raddoppio di d comporta una crescita pressoché lineare sia della dimensione delle chiavi pubbliche sia della dimensione delle proof.

L'analisi del rejection sampling ha evidenziato il ruolo centrale dei parametri α_i. Valori troppo piccoli comportano frequenti rigetti e quindi un aumento del tempo medio di generazione delle prove, mentre valori eccessivamente elevati determinano un incremento non necessario delle deviazioni standard gaussiane e delle dimensioni dei transcript.

Infine, è stato analizzato il passaggio dai protocolli interattivi alle prove non interattive mediante la trasformazione di Fiat-Shamir e il loro impiego nella costruzione delle prove $\Pi^{\text{ISIS}}_{\text{NIZK}}$ e $\Pi^{\text{Com}}_{\text{NIZK}}$. Tali strumenti sono stati successivamente inseriti nel framework di credenziali anonime introdotto nel capitolo 6, mostrando come le moderne tecniche di zero-knowledge basate sui reticoli possano costituire una valida alternativa alle costruzioni fondate su ipotesi crittografiche classiche.

Appendice A

Non Interactive Zero Knowledge Proofs

Questa appendice riporta, adattandole alla notazione della tesi, le costruzioni $\Pi^{\text{ISIS}}_{\text{NIZK}}$ e $\Pi^{\text{Com}}_{\text{NIZK}}$ presentate in Lyubashevsky et al. [2022] e Bootle et al. [2023] per fornire un raccordo tra il commitment ABDLOP e il framework per le credenziali anonime di Bootle et al. [2023].

A.1 Non Interactive Zero Knowledge Proof $\Pi^{\text{ISIS}}_{\text{NIZK}}$

Sia H un *random oracle*, che verrà utilizzato nella costruzione delle prove NIZK.

Nota 15. *Assumiamo che la funzione hash H sia modellata come un* random oracle*, ovvero una funzione che, per ogni nuovo input, restituisce un output uniformemente casuale e indipendente, mantenendo la consistenza sui valori già interrogati.*

Sia $\hat{d}$ una potenza di due e consideriamo l'anello:

$$\hat{\mathcal{R}} := \mathbb{Z}[X]/(X^{\hat{d}} + 1).$$

Dato un modulo $\hat{q} \in \mathbb{N}$, definiamo:

$$\hat{\mathcal{R}}_{\hat{q}} := \hat{\mathcal{R}}/(\hat{q}).$$

Nel seguito, assumiamo che $\hat{q} = \hat{p}_1\hat{p}_2$ sia il prodotto di due numeri primi tali che:

$$\hat{p}_i \equiv 5 \pmod 8, \quad \hat{p}_1 < \hat{p}_2.$$

Per un polinomio $x \in \hat{\mathcal{R}}_{\hat{q}}$, indichiamo con $\tilde{x}$ il suo coefficiente costante.

Assumiamo inoltre che $\hat{d}$ divida il parametro di sicurezza λ. Per il vettore di polinomi $\mathbf{x} \in \hat{\mathcal{R}}^n_{\hat{q}}$, definiamo:

$$\sigma(\mathbf{x}) := (\sigma(x_1), \ldots, \sigma(x_n)).$$

La definizione si estende anche a matrici.

Sia $\tau \in \mathbb{N}$ tale che:

$$\frac{1}{\hat{p}_1^{\tau}} = \text{negl}(\lambda).$$

Questo parametro verrà utilizzato per amplificare la *soundness* delle costruzioni, riducendo la probabilità di successo di avversari.

Assumiamo che le variabili n, m, t, ℓ_r siano divisibili per $\hat{d}$. Per il vettore $\vec{s}$, osserviamo che, per il teorema della somma di quattro quadrati, la condizione $\|\vec{s}\| \leq B_s$ è equivalente all'esistenza di quattro interi $a_1, a_2, a_3, a_4 \in \mathbb{Z}$ tali che

$$B_s^2 - \|\vec{s}\|^2 = a_1^2 + a_2^2 + a_3^2 + a_4^2.$$

Possiamo quindi definire il vettore

$$\vec{a} := (a_1, a_2, a_3, a_4, 0, \ldots, 0) \in \mathbb{Z}_{\hat{q}}^{\hat{d}},$$

e osservare che

$$\left\| \begin{bmatrix} \vec{s} \\ \vec{a} \end{bmatrix} \right\| = B_s \quad \text{e} \quad \begin{bmatrix} P & 0 \end{bmatrix} \begin{bmatrix} \vec{s} \\ \vec{a} \end{bmatrix} = P\vec{s}.$$

Definiamo

$$\hat{m} := m + \hat{d}, \quad \mathbf{s} := (\vec{s}, \vec{a}) \in \mathbb{Z}_{\hat{q}}^{m}, \quad P := \begin{bmatrix} P & 0 \end{bmatrix} \in \hat{\mathcal{R}}_{\hat{q}}^{n \times m}.$$

Analogamente procediamo per il vettore $\vec{r}$. Pertanto, l'obiettivo della prova è dimostrare che

$$\|\vec{s}\| = B_s, \quad \|\mathbf{r}\| = B_r, \quad \vec{u} \in \{0,1\}^t, \quad P\vec{s} = B\vec{u} + C \begin{bmatrix} \vec{m} \\ \vec{r} \end{bmatrix}. \tag{A.1}$$

La *common random string* consiste in matrici scelte uniformemente a caso e viene utilizzata per lo schema di commitment ABDLOP:

$$\text{crs} := (\mathbf{A}_1, \mathbf{A}_2, \mathbf{B}_y, \mathbf{B}_g, \mathbf{b}) \in \hat{\mathcal{R}}_{\hat{q}}^{n \times m_1} \times \hat{\mathcal{R}}_{\hat{q}}^{n \times m_2} \times \hat{\mathcal{R}}_{\hat{q}}^{256/\hat{d} \times m_2} \times \hat{\mathcal{R}}_{\hat{q}}^{\tau \times m_2} \times \hat{\mathcal{R}}_{\hat{q}}^{m_2}.$$

Definiamo lo *statement* come

$$x := (P, C, \vec{m}, B, \text{Bounds} := (B_s, B_r), \text{aux}).$$

Primo round

Sia $\mathbf{s} := \text{Coeffs}^{-1}(\mathbf{s}) \in \hat{\mathcal{R}}_{\hat{q}}^{\hat{m}/\hat{d}}$, e analogamente per $\mathbf{r}$ e $\mathbf{u}$. Definiamo $\mathbf{s}_1 := (\mathbf{s}, \mathbf{r}, \mathbf{u})$ e $m_1 := \hat{m} + \ell_r + t$. Si calcola il commitment di $\mathbf{s}_1$ utilizzando lo schema ABDLOP (usando solo la parte Ajtai), campionando i vettori casuali

$$\mathbf{s}_2 \leftarrow \chi^{m_2},$$

dove χ è una distribuzione di polinomi in $\hat{\mathcal{R}}_{\hat{q}}$, e calcolando

$$\mathbf{t}_A := \mathbf{A}_1\mathbf{s}_1 + \mathbf{A}_2\mathbf{s}_2,$$

con $(\mathbf{A}_1, \mathbf{A}_2) \in \hat{\mathcal{R}}_{\hat{q}}^{n\times m_1} \times \hat{\mathcal{R}}_{\hat{q}}^{n\times m_2}$.

Successivamente, si campionano vettori $\mathbf{y}_i \leftarrow D_{\sigma_i}^{m_i}$ per $i = 1,2$, e si calcola

$$\mathbf{w} := \mathbf{A}_1\mathbf{y}_1 + \mathbf{A}_2\mathbf{y}_2.$$

Generiamo ora un vettore polinomiale

$$\mathbf{y}_3 \leftarrow D_{\sigma_3}^{256}$$

e ne calcoliamo il commitment

$$\mathbf{t}_y := \mathbf{B}_y\mathbf{s}_2 + \mathbf{y}_3,$$

dove $\mathbf{B}_y \in \hat{\mathcal{R}}_{\hat{q}}^{256/\hat{d}\times m_2}$.

Infine, campioniamo un vettore polinomiale

$$\mathbf{g} \in \{x \in \hat{\mathcal{R}}_{\hat{q}} : \tilde{x} = 0\}^{\tau}$$

e ne calcoliamo il commitment

$$\mathbf{t}_g := \mathbf{B}_g\mathbf{s}_2 + \mathbf{g}, \quad \mathbf{B}_g \in \hat{\mathcal{R}}_{\hat{q}}^{\tau\times m_2}.$$

Il primo messaggio e la challenge corrispondente sono quindi

$$a_1 := (\mathbf{t}_A, \mathbf{t}_y, \mathbf{t}_g, \mathbf{w}), \quad (\mathfrak{R}_0, \mathfrak{R}_1) = H(1, \mathrm{crs}, x, a_1) \in \{0,1\}^{256\times m_1\hat{d}} \times \{0,1\}^{256\times m_1\hat{d}}.$$

Secondo round

Data la prima challenge $(\mathfrak{R}_0, \mathfrak{R}_1)$, definiamo

$$\mathfrak{R} := \mathfrak{R}_0 - \mathfrak{R}_1.$$

Siano $\vec{y}_3 := \mathrm{Coeffs}(\mathbf{y}_3)$ e $\vec{s}_1 := \mathrm{Coeffs}(\mathbf{s}_1)$. Il prover calcola quindi la risposta

$$\vec{z}_3 := \vec{y}_3 + \mathfrak{R}\,\vec{s}_1 \in \mathbb{Z}^{256},$$

ed applica rejection sampling su $\vec{z}_3$.

Il secondo messaggio è definito come

$$a_2 := \vec{z}_3.$$

La challenge successiva viene ottenuta come

$$(\gamma_{i,j})_{i\in[\tau],\, j\in[256+n+3]} = H(2, \mathrm{crs}, x, a_1, a_2) \in \mathbb{Z}_{\hat{q}}^{\tau\times(256+n+3)}.$$

Terzo round

L'obiettivo del prover in questo round è dimostrare le seguenti relazioni su $\mathbb{Z}_{\hat{q}}$:

$$\begin{cases} \vec{z}_3 = \vec{y}_3 + \mathfrak{R}\,\vec{s}_1, \\ P\vec{s} = B\vec{u} + C\begin{bmatrix}\vec{m}\\ \vec{r}\end{bmatrix}, \\ \langle \vec{s}, \vec{s} \rangle = B_s^2, \\ \langle \vec{r}, \vec{r} \rangle = B_r^2, \\ \langle \vec{u}, \vec{u} - \vec{1} \rangle = 0. \end{cases}$$

I termini segreti sono quelli impegnati nel commitment.

Utilizziamo ora la seguente osservazione: per ogni $\vec{x}_0, \vec{x}_1 \in \mathbb{Z}_{\hat{q}}^n$, il coefficiente costante del polinomio

$$\text{Coeffs}^{-1}(\vec{x}_0)^T\, \sigma\big(\text{Coeffs}^{-1}(\vec{x}_1)\big) \in \hat{\mathcal{R}}_{\hat{q}}$$

coincide con il prodotto scalare

$$\langle \vec{x}_0, \vec{x}_1 \rangle \in \mathbb{Z}_{\hat{q}}.$$

Infatti possiamo scrivere

$$x_0 = \text{Coeffs}^{-1}(\vec{x}_{0,j}) = \sum_k b_{0,k} X^k$$

e abbiamo dunque che $\text{Coeffs}^{-1}(\vec{x}_0)^T$ è un polinomio della forma A.1. Inoltre $\sigma\big(\text{Coeffs}^{-1}(\vec{x}_1)\big) = \sigma(\sum_k b_{1,k} X^k) = \sum_k b_{1,k} X^{-k}$ e abbiamo quindi che:

$$\text{Coeffs}^{-1}(\vec{x}_0)^T\, \sigma\big(\text{Coeffs}^{-1}(\vec{x}_1)\big) = \Big(\sum_k b_{0,k} X^k\Big)\Big(\sum_k b_{1,k} X^{-k}\Big)$$

. Si ha dunque che il termine corrispondende ad X_0 del prodotto è

$$\sum_k b_{0,k} b_{1,k} = \langle \vec{x}_0, \vec{x}_1 \rangle$$

Riformulazione delle equazioni. La prima equazione è equivalente a richiedere che, per ogni $j \in [256]$, il coefficiente costante di

$$\sigma(\mathbf{r}_j)^T \mathbf{s}_1 + \sigma(\mathbf{e}_j)^T \mathbf{y}_3 - z_{3,j}$$

sia nullo, dove:

- $\mathbf{r}_j$ è il polinomio il cui vettore dei coefficienti è la j-esima riga di $\mathfrak{R}$;
- $\mathbf{e}_j$ è tale che $\text{Coeffs}(\mathbf{e}_j)$ è il vettore unitario con 1 in j-esima posizione e 0 altrove.

Analogamente, la seconda equazione è equivalente a richiedere che, per ogni $j \in [n]$, il coefficiente costante di

$$\sigma(\mathbf{p}_j)^T \mathbf{s} - \sigma(\boldsymbol{\beta}_j)^T \mathbf{u} - m_{C,j} - \sigma(\mathbf{c}_{r,j})^T \mathbf{r}$$

sia nullo, dove:

- $\mathbf{p}_j$ è il polinomio associato alla j-esima riga di P;
- $\boldsymbol{\beta}_j$ è il polinomio associato alla j-esima riga di B;
- scriviamo $C = [C_m\ C_r]$ e definiamo $\vec{m}_C := C_m\vec{m}$;
- $\mathbf{c}_{r,j}$ è il polinomio associato alla j-esima riga di C_r.

Le ultime tre equazioni sono equivalenti a richiedere che i coefficienti costanti di:

$$\sigma(\mathbf{s})^T\mathbf{s} - B_s^2, \quad \sigma(\mathbf{r})^T\mathbf{r} - B_r^2, \quad \sigma(\mathbf{u})^T\mathbf{u} - \text{Coeffs}^{-1}(\vec{1})^T\mathbf{u}$$

siano nulli.

Aggregazione delle equazioni. Per ogni $i \in [\tau]$, definiamo:

$$\begin{aligned} h_i := \mathbf{g}_i &+ \sum_{j=1}^{256} \gamma_{i,j}\left(\sigma(\mathbf{r}_j)^T\mathbf{s}_1 + \sigma(\mathbf{e}_j)^T\mathbf{y}_3 - z_{3,j}\right) \\ &+ \sum_{j=1}^{n} \gamma_{i,256+j}\left(\sigma(\mathbf{p}_j)^T\mathbf{s} - \sigma(\boldsymbol{\beta}_j)^T\mathbf{u} - m_{C,j} - \sigma(\mathbf{c}_{r,j})^T\mathbf{r}\right) \\ &+ \gamma_{i,256+n+1}\left(\sigma(\mathbf{s})^T\mathbf{s} - B_s^2\right) \\ &+ \gamma_{i,256+n+2}\left(\sigma(\mathbf{r})^T\mathbf{r} - B_r^2\right) \\ &+ \gamma_{i,256+n+3}\left(\sigma(\mathbf{u})^T\mathbf{u} - \text{Coeffs}^{-1}(\vec{1})^T\mathbf{u}\right) \in \hat{\mathcal{R}}_{\hat{q}}. \end{aligned} \tag{A.2}$$

Per costruzione, i coefficienti costanti di $h_1, \ldots, h_\tau$ sono nulli.

Il terzo messaggio è quindi definito come

$$a_3 := (h_1, \ldots, h_\tau),$$

mentre la challenge finale è

$$\mu := (\mu_i)_{i\in[\tau]} = H(3, \text{crs}, x, a_1, a_2, a_3) \in \hat{\mathcal{R}}_{\hat{q}}^{\tau}.$$

Quarto round

L'obiettivo del prover è dimostrare le τ equazioni quadratiche su $\hat{\mathcal{R}}_{\hat{q}}$ definite in A.2.

A tal fine, le equazioni vengono combinate linearmente utilizzando i coefficienti μ_i, ottenendo un'unica equazione:

$$\begin{aligned} 0 = \sum_{i=1}^{\tau} \mu_i \Bigg(&\sum_{j=1}^{256} \gamma_{i,j}(\sigma(\mathbf{r}_j)^T\mathbf{s}_1 + \sigma(\mathbf{e}_j)^T\mathbf{y}_3 - z_{3,j}) \\ &+ \sum_{j=1}^{n} \gamma_{i,256+j}(\sigma(\mathbf{p}_j)^T\mathbf{s} - \sigma(\boldsymbol{\beta}_j)^T\mathbf{u} - m_{C,j} - \sigma(\mathbf{c}_{r,j})^T\mathbf{r}) \\ &+ \gamma_{i,256+n+1}(\sigma(\mathbf{s})^T\mathbf{s} - B_s^2) + \gamma_{i,256+n+2}(\sigma(\mathbf{r})^T\mathbf{r} - B_r^2) \\ &+ \gamma_{i,256+n+3}(\sigma(\mathbf{u})^T\mathbf{u} - \text{Coeffs}^{-1}(\vec{1})^T\mathbf{u}) + \mathbf{g}_i - h_i \Bigg). \end{aligned} \tag{A.3}$$

Forma quadratica compatta. Definiamo:

$$\mathbf{B} := \begin{bmatrix} \mathbf{B}_y \\ \mathbf{B}_g \end{bmatrix}, \quad \mathbf{t}_B := \begin{bmatrix} \mathbf{t}_y \\ \mathbf{t}_g \end{bmatrix}, \quad \mathbf{m} := \begin{bmatrix} \mathbf{y}_3 \\ \mathbf{g} \end{bmatrix} \in \hat{\mathcal{R}}_{\hat{q}}^{256/\hat{d}+\tau},$$

e

$$\hat{\mathbf{s}} := \begin{bmatrix} \mathbf{s}_1 \\ \sigma(\mathbf{s}_1) \\ \mathbf{m} \\ \sigma(\mathbf{m}) \end{bmatrix}.$$

Ricordando che $\mathbf{s}_1 = (\mathbf{s}, \mathbf{r}, \mathbf{u})$ e scrivendo $\mathbf{r}_j := (\mathbf{r}_{s,j}, \mathbf{r}_{r,j}, \mathbf{r}_{u,j})$ otteniamo:

$$\sigma(\mathbf{r}_j)^T \mathbf{s}_1 = \sigma(\mathbf{r}_{s,j})^T \mathbf{s} + \sigma(\mathbf{r}_{r,j})^T \mathbf{r} + \sigma(\mathbf{r}_{u,j})^T \mathbf{u}.$$

L'equazione (A.8) può quindi essere riscritta nella forma:

$$\hat{\mathbf{s}}^T D_2 \hat{\mathbf{s}} + d_1^T \hat{\mathbf{s}} + d_0 = 0,$$

dove $\mathbf{D}_2$, $\mathbf{d}_1$ e d_0 sono definiti come segue.

Definizione dei coefficienti. La matrice $\mathbf{D}_2$ è data da:

$$\mathbf{D}_2 = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \sum_{i=1}^{\tau} \mu_i \gamma_{i,256+n+1} \mathbf{I}_{m/d} & \sum_{i=1}^{\tau} \mu_i \gamma_{i,256+n+2} \mathbf{I}_{\ell_r/d} & \sum_{i=1}^{\tau} \mu_i \gamma_{i,256+n+3} \mathbf{I}_{n/d} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}.$$

Il termine lineare $\mathbf{d}_1$ è:

$$\mathbf{d}_1 = \begin{bmatrix} \sum_{i=1}^{\tau} \mu_i \Big(\sum_{j=1}^{256} \gamma_{i,j} \sigma(\mathbf{r}_{s,j}) + \sum_{j=1}^{n} \gamma_{i,256+j} \sigma(\mathbf{p}_j) \Big) \\ \sum_{i=1}^{\tau} \mu_i \Big(\sum_{j=1}^{256} \gamma_{i,j} \sigma(\mathbf{r}_{r,j}) - \sum_{j=1}^{n} \gamma_{i,256+j} \sigma(\mathbf{c}_{r,j}) \Big) \\ \sum_{i=1}^{\tau} \mu_i \Big(\sum_{j=1}^{256} \gamma_{i,j} \sigma(\mathbf{r}_{u,j}) - \sum_{j=1}^{n} \gamma_{i,256+j} \sigma(\boldsymbol{\beta}_j) - \gamma_{i,256+n+3} \mathrm{Coeffs}^{-1}(\vec{1}) \Big) \\ \mathbf{0} \\ \sum_{i=1}^{\tau} \sum_{j=1}^{256} \mu_i \gamma_{i,j} \sigma(\mathbf{e}_j) \\ \mu \\ \mathbf{0} \end{bmatrix}.$$

Il termine costante è:

$$d_0 = -\sum_{i=1}^{\tau} \mu_i \Big(\sum_{j=1}^{256} z_{3,j} + \sum_{j=1}^{n} m_{C,j} + \gamma_{i,256+n+1} B_s^2 + \gamma_{i,256+n+2} B_r^2 + h_i \Big).$$

Riduzione a una singola equazione quadratica. A questo punto, eseguiamo il protocollo per dimostrare una singola equazione quadratica con automorfismi.

Definiamo:

$$\mathbf{y} := \begin{bmatrix} \mathbf{y}_1 \\ \sigma(\mathbf{y}_1) \\ -\mathbf{B}\mathbf{y}_2 \\ -\sigma(\mathbf{B}\mathbf{y}_2) \end{bmatrix} \in \hat{\mathcal{R}}_{\hat{q}}^{2(m_1+256/\hat{d}+\tau)}.$$

Calcoliamo quindi il termine di mascheramento:

$$f_1 = \hat{\mathbf{s}}^T \mathbf{R}_2 \mathbf{y} + \mathbf{y}^T \mathbf{R}_2 \hat{\mathbf{s}} + r_1^T \mathbf{y},$$

e il relativo commitment:

$$t = \mathbf{b}^T \mathbf{s}_2 + f_1.$$

Poniamo inoltre:

$$f_0 = \mathbf{y}^T R_2 \mathbf{y} + \mathbf{b}^T \mathbf{y}_2.$$

Il quarto messaggio è quindi:

$$a_4 := (t, f_0),$$

e la challenge finale è:

$$c = H(4, \mathrm{crs}, x, a_1, a_2, a_3, a_4) \in \mathcal{C}.$$

Quinto round

Dato il challenge c, calcoliamo

$$\mathbf{z}_i := c\mathbf{s}_i + \mathbf{y}_i \quad \text{per } i \in \{1,2\},$$

e applichiamo rejection sampling su $\mathbf{z}_1, \mathbf{z}_2$.

Il messaggio finale del prover è quindi

$$a_5 := (\mathbf{z}_1, \mathbf{z}_2),$$

e la prova completa è

$$\pi := (a_1, a_2, a_3, a_4, a_5).$$

Round di Verifica

Sia data la prova

$$\pi = (a_1, a_2, a_3, a_4, a_5).$$

Il verificatore ricostruisce i challenge corrispondenti e calcola i parametri

$$\mathbf{D}_2,\ \mathbf{d}_1,\ d_0.$$

$$\mathbf{z} := \begin{bmatrix} \mathbf{z}_1 \\ \sigma(\mathbf{z}_1) \\ c\mathbf{t}_B - \mathbf{B}\mathbf{z}_2 \\ \sigma(c\mathbf{t}_B - \mathbf{B}\mathbf{z}_2) \end{bmatrix} \in \hat{\mathcal{R}}_{\hat{q}}^{2m_1+512/\hat{d}+2\tau}$$

Il verificatore accetta se tutte le seguenti relazioni valgono:

$$\begin{cases} \|\mathbf{z}_1\| \overset{?}{\leq} B_1, \\ \|\mathbf{z}_2\| \overset{?}{\leq} B_2, \\ \|\mathbf{z}_3\| \overset{?}{\leq} B_3, \\ \tilde{h}_i \overset{?}{=} 0, \quad \forall i \in [\tau], \\ \mathbf{A}_1\mathbf{z}_1 + \mathbf{A}_2\mathbf{z}_2 \overset{?}{=} \mathbf{w} + c\mathbf{t}_A, \\ \mathbf{z}^\top \mathbf{D}_2 \mathbf{z} + c\,\mathbf{d}_1^\top \mathbf{z} + c^2 d_0 - (ct - \mathbf{b}^\top \mathbf{z}_2) \overset{?}{=} f_0 \end{cases}$$

Prima di dimostrare le proprietà di sicurezza son riportati gli schemi degli algoritmi presentati

$$\Pi_{\mathsf{NIZK}}^{\mathsf{ISIS}} := (\mathsf{Prove}_H^{\mathsf{ISIS}}, \mathsf{Verify}_H^{\mathsf{ISIS}})$$

Algoritmo 14: $\mathsf{Prove}^{\mathsf{ISIS}}_{H}$

Input: Input pubblico: $\mathrm{crs} = (\mathbf{A}_1, \mathbf{A}_2, \mathbf{B}_y, \mathbf{B}_g, \mathbf{b})$, $x = (P, C, \vec{m}, B, \mathrm{Bounds}, \mathrm{aux})$
Input segreto: $w = (\vec{s}, \vec{r}, \vec{u})$
Output: Prova π
begin
  $\mathtt{rst} \leftarrow 0,\ \mathtt{idx} \leftarrow 0,\ \mathbf{s}_1 := (\mathbf{s}, \mathbf{r}, \mathbf{u})$;
  while $rst = 0 \wedge idx < N$ **do**
    $\mathtt{idx} \leftarrow \mathtt{idx} + 1$;
    $\mathbf{s}_2 \leftarrow \chi^{m_2}$;
    $\mathbf{t}_A = \mathbf{A}_1\mathbf{s}_1 + \mathbf{A}_2\mathbf{s}_2$;
    $\mathbf{y}_1 \leftarrow D^{m_1}_{\mathbf{0},\sigma_1},\ \mathbf{y}_2 \leftarrow D^{m_2}_{\mathbf{0},\sigma_2},\ \mathbf{y}_3 \leftarrow D^{256}_{\mathbf{0},\sigma_3}$;
    $\mathbf{g} \leftarrow \{x : \tilde{x} = 0\}^{\tau}$;
    $\mathbf{w} = \mathbf{A}_1\mathbf{y}_1 + \mathbf{A}_2\mathbf{y}_2$;
    $\mathbf{t}_y = \mathbf{B}_y\mathbf{s}_2 + \mathbf{y}_3$;
    $\mathbf{t}_g = \mathbf{B}_g\mathbf{s}_2 + \mathbf{g}$;
    $a_1 = (\mathbf{t}_A, \mathbf{t}_y, \mathbf{t}_g, \mathbf{w})$;
    $(\mathfrak{R}_0, \mathfrak{R}_1) = H(1, \mathrm{crs}, x, a_1)$;
    $\mathfrak{R} = \mathfrak{R}_0 - \mathfrak{R}_1$;
    $\vec{z}_3 = \vec{y}_3 + \mathfrak{R}\vec{s}_1$;
    $b_3 \leftarrow \mathrm{Rej}(\vec{z}_3, \mathfrak{R}\vec{s}_1)$;
    $a_2 = \vec{z}_3$;
    $(\gamma_{i,j}) = H(2, \mathrm{crs}, x, a_1, a_2)$;
    $a_3 = (h_1, \ldots, h_\tau)$;
    $(\mu_i) = H(3, \mathrm{crs}, x, a_1, a_2, a_3)$;
    $f_1 = \hat{\mathbf{s}}^{\top}\mathbf{R}_2\mathbf{y} + \mathbf{y}^{\top}\mathbf{R}_2\hat{\mathbf{s}} + \mathbf{r}_1^{\top}\mathbf{y}$;
    $f_0 = \mathbf{y}^{\top}\mathbf{R}_2\mathbf{y} + \mathbf{b}^{\top}\mathbf{y}_2$;
    $t = \mathbf{b}^{\top}\mathbf{s}_2 + f_1$;
    $a_4 = (t, f_0)$;
    $c = H(4, \mathrm{crs}, x, a_1, a_2, a_3, a_4)$;
    $\mathbf{z}_i = \mathbf{y}_i + c\mathbf{s}_i$;
    $b_i \leftarrow \mathrm{Rej}(\mathbf{z}_i, c\mathbf{s}_i)$;
    $\pi = (\mathbf{t}_A, \mathbf{t}_y, \mathbf{t}_g, \mathbf{w}, \vec{z}_3, \mathbf{h}, t, f_0, \mathbf{z}_1, \mathbf{z}_2)$;
    $\mathtt{rst} = b_1b_2b_3$;
    if $rst = 1$ **then**
      return π;
  return $\perp$;

Algoritmo 15: $\mathsf{Verify}_H^{\mathsf{ISIS}}$

Input: Input pubblico: $\text{crs} = (\mathbf{A}_1, \mathbf{A}_2, \mathbf{B}_y, \mathbf{B}_g, \mathbf{b})$, $x = (P, C, \vec{m}, B, \text{Bounds}, \text{aux})$
Input: prova π
Output: Bit b
begin
$\pi = (\mathbf{t}_A, \mathbf{t}_y, \mathbf{t}_g, \mathbf{w}, \vec{z}_3, \mathbf{h}, t, f_0, \mathbf{z}_1, \mathbf{z}_2)$;
$a_1 = (\mathbf{t}_A, \mathbf{t}_y, \mathbf{t}_g, \mathbf{w}), \quad a_2 = \vec{z}_3$;
$a_3 = (h_1, \ldots, h_\tau) = \mathbf{h}, \quad a_4 = (t, f_0)$;
$(\mathfrak{R}_0, \mathfrak{R}_1) = H(1, \text{crs}, x, a_1)$;
$(\gamma_{i,j}) = H(2, \text{crs}, x, a_1, a_2)$;
$(\mu_i) = H(3, \text{crs}, x, a_1, a_2, a_3)$;
$c = H(4, \text{crs}, x, a_1, a_2, a_3, a_4)$;
$\mathbf{B} = (\mathbf{B}_y \ \mathbf{B}_g), \quad \mathbf{t}_B = (\mathbf{t}_y \ \mathbf{t}_g)$;
$\mathbf{z} = \begin{bmatrix} \mathbf{z}_1 \\ \sigma(\mathbf{z}_1) \\ c\mathbf{t}_B - \mathbf{B}\mathbf{z}_2 \\ \sigma(c\mathbf{t}_B - \mathbf{B}\mathbf{z}_2) \end{bmatrix}$;
Calcola $\mathbf{D}_2, \mathbf{d}_1, d_0$;
if $\|\mathbf{z}_1\| \leq B_1$ ***and*** $\|\mathbf{z}_2\| \leq B_2$ ***and*** $\|\vec{z}_3\| \leq B_3$ ***and***
$\tilde{h}_i = 0 \ \forall i \in [\tau]$ ***and***
$\mathbf{A}_1\mathbf{z}_1 + \mathbf{A}_2\mathbf{z}_2 = \mathbf{w} + c\,\mathbf{t}_A$ ***and***
$\mathbf{z}^\top \mathbf{D}_2 \mathbf{z} + c\,\mathbf{d}_1^\top \mathbf{z} + c^2 d_0 - (ct - \mathbf{b}^\top \mathbf{z}_2) = f_0$ **then**
  return 1;
else
  return 0;

A.1.1 Correttezza

Prima di procedere con la dimostrazione di correttezza per $\Pi_{\text{NIZK}}^{\text{ISIS}} = (\text{Prove}_{\text{ISIS}}^{\text{H}}, \text{Verify}_{\text{ISIS}}^{\text{H}})$ enunciamo due lemmi sul rejection sampling. Il primo è una versione più specifica del teorema 3.3.1.

Lemma A.1.1. *Sia $V \subseteq \mathcal{R}^\ell$ un insieme di polinomi con norma limitata da T, cioè*

$$\|\mathbf{v}\| \leq T \quad \forall \mathbf{v} \in V,$$

e sia $\rho : V \to [0,1]$ una distribuzione di probabilità su V.
Sia fissata la deviazione standard

$$\sigma = \alpha T, \quad \text{con } \alpha = O(\sqrt{\lambda}),$$

e

$$M = \exp\left(\sqrt{\frac{2(\lambda+1)}{\log(e)}} \cdot \frac{1}{\alpha} + \frac{1}{2\alpha^2}\right) = O(1).$$

Sia $\mathbf{v} \leftarrow \rho$ *e* $\mathbf{y} \leftarrow D_{\mathbf{0},\sigma}^{\ell}$*, e si definisca*

$$\mathbf{z} = \mathbf{y} + \mathbf{v}.$$

Esegui

$$b \leftarrow Rej(\mathbf{z}, \mathbf{v}, \sigma, M)$$

Allora:

1. *La probabilità che* $b = 1$ *soddisfa*

$$\mathbb{P}[b = 1] \geq \frac{1 - 2^{-\lambda}}{M}.$$

2. *La distribuzione congiunta* $(\mathbf{v}, \mathbf{z})$ *condizionata a* $b = 1$ *è a distanza statistica al più* $2^{-\lambda}$ *dalla distribuzione prodotto*

$$\rho \times D_{\mathbf{0},\sigma}^{\ell}.$$

Lemma A.1.2. *Siano* $\alpha, T > 0$ *e* $\varepsilon \leq 1/2$*. Sia* $\mathbf{v} \in \mathcal{R}_q^m \quad con \quad \|\mathbf{v}\| \leq T$*, e siano* $\mathbf{A} \in \mathcal{R}_q^{n \times m}, \quad \mathbf{w} \in \mathcal{R}_q^n, \quad \mathbf{t} := \mathbf{A}\mathbf{v} \in \mathcal{R}_q^n$*.*

Sia $\sigma \geq \max(\alpha T, \eta_\varepsilon(\Lambda_q^\perp(\mathbf{A})))$*, dove* $\Lambda_q^\perp(\mathbf{A})$ *denota il reticolo* $\Lambda_q^\perp(\mathbf{A}) = \{\mathbf{x} \in \mathcal{R}^m : \mathbf{A}\mathbf{x} = \mathbf{0} \pmod q\}$ *e* $\eta_\varepsilon(\Lambda)$ *è* $arg\min_{\sigma>0}\{\rho_{0,\frac{1}{\sigma\sqrt{2\pi}}}(\Lambda^* - \{0\}) \leq \epsilon, \epsilon > 0\}$*.*

Definiamo

$$M := \exp\Big(\frac{1}{2\alpha^2} + t/\alpha\Big), \quad \varepsilon' := \frac{2 + 2\varepsilon}{1 + \varepsilon} \cdot \exp(-2t^2(\pi - 1)/\pi).$$

Allora, sia $(\mathbf{z}, b)$ *la coppia ottenuta dal procedimento di* Rejection Sampling *standard con input* $(\mathbf{A}, \mathbf{t}, \mathbf{w}, \sigma, M)$ *e valgono le seguenti proprietà:*

1. *La distanza statistica tra la distribuzione effettiva* $(\mathbf{v}, \mathbf{z})$ *condizionata all'accettazione e la distribuzione prodotto*

$$\mathbf{v} \sim \rho, \quad \mathbf{y} \sim D_{\mathbf{0},\sigma}^m$$

è al più

$$\frac{\varepsilon'}{2M} + \frac{2\varepsilon}{M}.$$

2. *La probabilità che il* Rejection Sampling *restituisca un output accettato è almeno*

$$\frac{1 - \varepsilon'}{M}\Big(1 - \frac{4\varepsilon}{(1 + \varepsilon)^2}\Big).$$

Nota 16. *Il parametro di smoothing di un reticolo* $\mathcal{L}$*,* $\eta_\varepsilon(\mathcal{L})$*, indica la deviazione standard minima necessaria affinché una gaussiana discreta sia sufficientemente "liscia" da risultare quasi uniforme.*

Teorema A.1.3. *Siano* $\alpha_1, \alpha_2, \alpha_3, \bar{\alpha}_1, \bar{\alpha}_2 \in O(\sqrt{\lambda})$ *e* $N, \hat{d} \in O(\lambda)$. *Inoltre, sia* χ *una distribuzione di probabilità su* $\mathcal{S}_1 = \{\mathbf{x} \in \mathcal{R}_q^m : \|\mathbf{x}\|_\infty \leq 1\}$. *Siano* $\omega_{\min}(\lambda)$ *e* $\omega_{\max}(\lambda)$ *due funzioni per cui, data una matrice* $\mathfrak{R} \leftarrow Bin_\xi^{256\times m}$, *valga:*

$$\mathbb{P}[\|\mathfrak{R}\vec{w}\|^2 > \xi\|\vec{w}\|^2\omega_{\min}(\lambda)^2] \leq 2^{-2\lambda} \quad \mathbb{P}[\|\mathfrak{R}\vec{w}\|^2 < \xi\|\vec{w}\|^2\omega_{\max}(\lambda)^2] \leq 2^{-\lambda}.$$

Sia $\nu > 0$ *un parametro di scaling, definiamo:*

$$\sigma_1 := \alpha_1\nu\sqrt{B_s^2 + B_r^2 + t}, \qquad \sigma_2 := \alpha_2\nu\sqrt{m_2\hat{d}}, \qquad \sigma_3 := \alpha_3\,\omega_{\max}(\lambda)\sqrt{B_s^2 + B_r^2 + t},$$

$$B_1 := \sigma_1\sqrt{2m_1\hat{d}}, \qquad B_2 := \sigma_2\sqrt{2m_2\hat{d}}, \qquad B_3 := 1.7\sigma_3\sqrt{256}.$$

Definiamo i corrispondenti tassi di rifiuto del canonico algoritmo di rejection sampling:

$$M_i := \exp\left(\sqrt{\frac{2(\lambda+1)}{\log(e)}}\frac{1}{\alpha_i} + \frac{1}{2\alpha_i^2}\right) \quad \textit{per } i \in \{1{,}2{,}3\}.$$

Allora, $\Pi_{\mathsf{ISIS}}^{\mathsf{NIZK}} := (\mathsf{Prove}_{\mathsf{ISIS}}^{H}, \mathsf{Verify}_{\mathsf{ISIS}}^{H})$ *è corretto.*

Dimostrazione. Mostriamo che l'algoritmo del prover restituisce $\perp$ con probabilità trascurabile e che le equazioni di verifica sono soddisfatte con alta probabilità.

Correttezza del prover. Per come è stato definito lo spazio delle challenge nella sezione 4.6.4 si ha che le seguenti disuguaglianze sono soddisfatte con alta probabilità:

$$\|c\mathbf{s}_1\| \leq \nu\sqrt{B_s^2 + B_r^2 + t}, \qquad \|c\mathbf{s}_2\| \leq \nu\sqrt{m_2\hat{d}}, \qquad \|\mathfrak{R}\vec{s}_1\| \leq \omega_{\max}(\lambda)\sqrt{B_s^2 + B_r^2 + t}.$$

Applicando il lemma A.1.1 al rejection sampling, deduciamo che

$$\mathbb{P}[b_1b_2b_3 = 1] \geq \prod_i \left(\frac{1}{M_i} - negl(\lambda)\right) \geq \frac{1}{M_1M_2M_3} - \mathrm{negl}(\lambda).$$

Pertanto, la probabilità che $\mathsf{Prove}_{\mathsf{ISIS}}^{\mathsf{H}}$ restituisca $\perp$, ovvero fallisca per N tentativi consecutivi, è minore o uguale a:

$$\left(1 - \frac{1}{M_1M_2M_3} + \mathrm{negl}(\lambda)\right)^N.$$

Poiché $M_1, M_2, M_3 \in O(1)$ e $N \in O(\lambda)$, segue che:

$$\left(1 - \frac{1}{M_1M_2M_3} + \mathrm{negl}(\lambda)\right)^N = \mathrm{negl}(\lambda),$$

da cui la probabilità di fallimento del prover è trascurabile.

Correttezza della verifica. Per quanto riguarda le equazioni di verifica, dal lemma 4.6.1 con parametro $t = \sqrt{2}$ otteniamo che:

$$\mathbb{P}\left[\|\mathbf{z}_1\| \leq B_1 \ \wedge \ \|\mathbf{z}_2\| \leq B_2\right] = \prod_i \mathbb{P}[\|\mathbf{z}_i\| \leq B_i] = \prod_i \left(1 - \mathbb{P}[\|\mathbf{z}_i\| > B_i]\right) > \prod_i \left(1 - (\sqrt{2}e^{-0.5})^{m_i}\right) = 1 - \text{negl}(\lambda).$$

Analogamente, applicando il lemma 4.6.1 con $t = 1.7$, otteniamo:

$$\mathbb{P}\left[\|\mathbf{z}_3\| \leq B_3\right] = 1 - \mathbb{P}\left[\|\mathbf{z}_3\| > B_3\right] > 1 - (1.7e^{0.5-1.7^2/2})^{256/\hat{d}} = 1 - negl(\lambda).$$

Per costruzione abbiamo che $\tilde{h}_i = \tilde{g}_i = 0$ per definizione di $\mathbf{g}$. Inoltre,

$$\mathbf{A}_1\mathbf{z}_1 + \mathbf{A}_2\mathbf{z}_2 = \mathbf{A}_1(\mathbf{y}_1 + c\mathbf{s}_1) + \mathbf{A}_2(\mathbf{y}_2 + c\mathbf{s}_2) = \mathbf{A}_1\mathbf{y}_1 + \mathbf{A}_2\mathbf{y}_2 + c(\mathbf{A}_1\mathbf{s}_1 + \mathbf{A}_2\mathbf{s}_2) = \mathbf{w} + c\mathbf{t}_A.$$

Verifichiamo ora se $\mathbf{z}^\top \mathbf{D}_2\mathbf{z} + c\,\mathbf{d}_1^\top\mathbf{z} + c^2 d_0 - (ct - \mathbf{b}^\top\mathbf{z}_2) = f_0$. Siccome vale:

$$\mathbf{z}_i = c\mathbf{s}_i + \mathbf{y}_i, \quad \text{per } i \in \{1,2\},$$

segue che

$$\mathbf{z} = c\hat{\mathbf{s}} + \mathbf{y}.$$

Espandiamo il termine quadratico:

$$\mathbf{z}^\top D_2\mathbf{z} = (c\hat{\mathbf{s}} + \mathbf{y})^\top D_2(c\hat{\mathbf{s}} + \mathbf{y}) = c^2\hat{\mathbf{s}}^\top D_2\hat{\mathbf{s}} + 2c\hat{\mathbf{s}}^\top D_2\mathbf{y} + \mathbf{y}^\top D_2\mathbf{y}.$$

Analogamente, il termine lineare diventa:

$$c\,d_1^\top\mathbf{z} = c^2 d_1^\top\hat{\mathbf{s}} + c\,d_1^\top\mathbf{y}.$$

Sommando i termini otteniamo:

$$\mathbf{z}^\top D_2\mathbf{z} + cd_1^\top\mathbf{z} + c^2 d_0 = c^2(\hat{\mathbf{s}}^\top D_2\hat{\mathbf{s}} + d_1^\top\hat{\mathbf{s}} + d_0) + 2c\hat{\mathbf{s}}^\top D_2\mathbf{y} + cd_1^\top\mathbf{y} + \mathbf{y}^\top D_2\mathbf{y}.$$

Poiché per costruzione vale

$$\hat{\mathbf{s}}^\top D_2\hat{\mathbf{s}} + d_1^\top\hat{\mathbf{s}} + d_0 = 0,$$

segue che:

$$\mathbf{z}^\top D_2\mathbf{z} + cd_1^\top\mathbf{z} + c^2 d_0 = 2c\hat{\mathbf{s}}^\top D_2\mathbf{y} + cd_1^\top\mathbf{y} + \mathbf{y}^\top D_2\mathbf{y}.$$

Osserviamo ora che, per definizione di f_1, si ha:

$$f_1 = \hat{\mathbf{s}}^\top D_2\mathbf{y} + \mathbf{y}^\top D_2\hat{\mathbf{s}} + d_1^\top\mathbf{y}.$$

Per simmetria del termine quadratico,

$$\hat{\mathbf{s}}^\top D_2\mathbf{y} + \mathbf{y}^\top D_2\hat{\mathbf{s}} = 2\hat{\mathbf{s}}^\top D_2\mathbf{y},$$

e dunque:

$$f_1 = 2\hat{\mathbf{s}}^\top D_2\mathbf{y} + d_1^\top\mathbf{y}.$$

Ne consegue che:

$$2c\hat{\mathbf{s}}^\top D_2\mathbf{y} + cd_1^\top\mathbf{y} = cf_1.$$

Pertanto:

$$\mathbf{z}^\top D_2\mathbf{z} + cd_1^\top\mathbf{z} + c^2d_0 = cf_1 + \mathbf{y}^\top D_2\mathbf{y}.$$

Richiamando ora le definizioni:

$$t = \mathbf{b}^\top\mathbf{s}_2 + f_1, \quad f_0 = \mathbf{y}^\top D_2\mathbf{y} + \mathbf{b}^\top\mathbf{y}_2,$$

si ottiene:

$$ct = c\mathbf{b}^\top\mathbf{s}_2 + cf_1,$$

e

$$\mathbf{b}^\top\mathbf{z}_2 = \mathbf{b}^\top(c\mathbf{s}_2 + \mathbf{y}_2) = c\mathbf{b}^\top\mathbf{s}_2 + \mathbf{b}^\top\mathbf{y}_2.$$

Da cui:

$$ct - \mathbf{b}^\top\mathbf{z}_2 = cf_1 - \mathbf{b}^\top\mathbf{y}_2.$$

Infine,

$$\mathbf{z}^\top D_2\mathbf{z} + cd_1^\top\mathbf{z} + c^2d_0 - (ct - \mathbf{b}^\top\mathbf{z}_2) = \mathbf{y}^\top D_2\mathbf{y} + \mathbf{b}^\top\mathbf{y}_2 = f_0.$$

Conclusione. Combinando i risultati precedenti, segue che il protocollo $\Pi_{\mathsf{ISIS}}^{\mathsf{NIZK}}$ è corretto. □

A.1.2 Zero-Knowledge

Teorema A.1.4. *Siano* $m_2 \geq 256/\hat{d} + \tau + 1$, $\alpha_1, \alpha_2, \alpha_3 \in O(\sqrt{\lambda})$ *e* $N, \hat{d} \in O(\lambda)$. *Allora il protocollo* Π_{NIZK}^{ISIS} *è zero-knowledge sotto l'assunzione di difficoltà del problema*

$$\mathsf{MLWE}_{n+256/\hat{d}+\tau+1, m_2, \chi, \hat{q}}.$$

Dimostrazione. Dimostriamo il risultato mediante un *hybrid argument.*

Nota 17 (Hybrid argument)**.** *Date* $\mathcal{D}_0, \mathcal{D}_1$ *due distribuzioni su uno stesso spazio* $\mathcal{X}$. *Un* hybrid argument *consiste nel definire una sequenza di distribuzioni intermedie*

$$\mathcal{D}_0, \mathcal{D}_1, \ldots, \mathcal{D}_k$$

tali che:

- $\mathcal{D}_0$ *è la distribuzione reale;*
- $\mathcal{D}_k$ *è la distribuzione simulata;*
- *per ogni* $i \in \{0, \ldots, k-1\}$, *le distribuzioni* $\mathcal{D}_i$ *e* $\mathcal{D}_{i+1}$ *sono computazionalmente indistinguibili.*

In altre parole, per ogni avversario PPT $\mathcal{A}$ esiste una funzione trascurabile $\varepsilon(\lambda)$ tale che:

$$\left|\mathbb{P}_{x\leftarrow\mathcal{D}_i}[\mathcal{A}(x)=1]-\mathbb{P}_{x\leftarrow\mathcal{D}_{i+1}}[\mathcal{A}(x)=1]\right|\leq\varepsilon(\lambda)\quad\forall i.$$

Segue per disuguaglianza triangolare che:

$$|\mathbb{P}_{x\leftarrow\mathcal{D}_0}[\mathcal{A}(x)=1]-\mathbb{P}_{x\leftarrow\mathcal{D}_k}[\mathcal{A}(x)=1]|\leq\sum_{i=0}^{k-1}\varepsilon(\lambda)=\mathrm{negl}(\lambda).$$

L'idea dell'hybrid argument è quindi quella di dimostrare l'indistinguibilità tra due distribuzioni complesse non direttamente, ma tramite una sequenza di passaggi intermedi più semplici.

Invece di confrontare direttamente $\mathcal{D}_0$ (distribuzione reale) con $\mathcal{D}_k$ (distribuzione simulata), si costruisce una catena di distribuzioni

$$\mathcal{D}_0\to\mathcal{D}_1\to\cdots\to\mathcal{D}_k,$$

in cui ogni passaggio modifica solo una piccola parte della distribuzione.

Se ogni coppia consecutiva $\mathcal{D}_i,\mathcal{D}_{i+1}$ è indistinguibile, allora anche la distribuzione iniziale e quella finale risultano indistinguibili.

Sia $\mathcal{A}$ un avversario PPT che effettua al più Q_0 query all'oracolo random H e al più Q_1 query all'algoritmo $\mathsf{Prove}^H_{\mathrm{ISIS}}$, e che restituisce 1 con probabilità ε. L'obiettivo della dimostrazione di zero-knowledge è far notare che le prove generate dal prover possono essere simulate senza conoscere il witness. In questo contesto:

- i primi passi servono a scollegare le challenge dall'oracolo random;
- i passaggi centrali utilizzano il rejection sampling per rendere le distribuzioni indipendenti dal witness;
- i passaggi finali sostituiscono i commitment con valori uniformi sfruttando l'assunzione MLWE.

Costruiamo due algoritmi PPT $\mathcal{S}_0$ e $\mathcal{S}_1$, che condividono uno stato interno, tali che:

- $\mathcal{S}_0$ simula l'oracolo random H mediante *lazy sampling*(valori campionati casualmente solo al momento della prima query e salvati nelle successive);
- $\mathcal{S}_1$ simula il prover.

Indichiamo con ε_i la probabilità che $\mathcal{A}$ restituisca 1 nell'i-esimo gioco.

Game 1. $\mathcal{S}_0$ simula l'oracolo random, mentre $\mathcal{S}_1$ esegue onestamente $\mathsf{Prove}^H_{\mathrm{ISIS}}(crs,x,w)$. Per costruzione:

$$\varepsilon_1=\varepsilon.$$

Game 2. Quando $\mathcal{S}_1$ riceve una query, campiona una challenge $c \leftarrow \mathcal{C}$ all'inizio dell'esecuzione e la utilizza al posto dell'output dell'oracolo H nella fase finale.

Se la tupla $(4, crs, x, a_1, a_2, a_3, a_4)$ è già stata interrogata all'oracolo $\mathcal{S}_0$, allora $\mathcal{S}_1$ restituisce $\perp$.

L'unica differenza rispetto al gioco precedente è quindi la programmazione dell'oracolo random: adesso programmiamo direttamente c senza controllare se la tupla $a = (4, crs, x, a_1, a_2, a_3, a_4)$ fosse già stata impostata. Dato che $\mathcal{A}$ può interrogare $\mathcal{S}_0$ al più Q_0 volte e $\mathcal{S}_1$ al più Q_1 volte, possono essere stati impostate al più $Q_0 + NQ_1$ tuple a e dunque al più $Q_0 + NQ_1$ valori di $\mathbf{w}$. Facendo riferimento a Esgin et al. [2019] (Appendice C) in Bootle et al. [2023] si conclude che

$$\mathbb{P}[\mathbf{A}_1\mathbf{y}_1 + \mathbf{A}_2\mathbf{y}_2 = \mathbf{w}, \mathbf{y}_1 \leftarrow D^{m_1\hat{d}}_{\mathbf{0},\sigma_1}, \mathbf{y}_2 \leftarrow D^{m_2\hat{d}}_{\mathbf{0},\sigma_2}] \leq 2^{-n\hat{d}}.$$

Pertanto se $\mathcal{S}_1$ viene interrogato al più Q_1 volte e la probabilità di avere una collisione ogni volta è al più $(Q_0 + NQ_1)2^{-n\hat{d}}$, allora la probabilità di avere una collisione dopo Q_1 queries è al più $NQ_1(Q_0 + NQ_1)2^{-n\hat{d}}$. Da cui si conclude che

$$|\varepsilon_2 - \varepsilon_1| \leq NQ_1(Q_0 + NQ_1)2^{-n\hat{d}} + \text{negl}(\lambda) = \text{negl}(\lambda).$$

Game 3. $\mathcal{S}_1$ riformula le variabili utilizzando direttamente:

$$\mathbf{z}_i = \mathbf{y}_i + c\mathbf{s}_i \quad \text{per } i \in \{1,2\},$$

e

$$\mathbf{w} = \mathbf{A}_1\mathbf{z}_1 + \mathbf{A}_2\mathbf{z}_2 - c\mathbf{t}_A.$$

Osserviamo che, utilizzando la relazione $\mathbf{t}_A = \mathbf{A}_1\mathbf{s}_1 + \mathbf{A}_2\mathbf{s}_2$, vale:

$$\begin{aligned}\mathbf{A}_1\mathbf{z}_1 + \mathbf{A}_2\mathbf{z}_2 &= \mathbf{A}_1(\mathbf{y}_1 + c\mathbf{s}_1) + \mathbf{A}_2(\mathbf{y}_2 + c\mathbf{s}_2) \\ &= \mathbf{A}_1\mathbf{y}_1 + \mathbf{A}_2\mathbf{y}_2 + c(\mathbf{A}_1\mathbf{s}_1 + \mathbf{A}_2\mathbf{s}_2) \\ &= \mathbf{A}_1\mathbf{y}_1 + \mathbf{A}_2\mathbf{y}_2 + c\mathbf{t}_A.\end{aligned}$$

Da cui segue che

$$\mathbf{A}_1\mathbf{y}_1 + \mathbf{A}_2\mathbf{y}_2 = \mathbf{A}_1\mathbf{z}_1 + \mathbf{A}_2\mathbf{z}_2 - c\mathbf{t}_A,$$

che coincide con la definizione di $\mathbf{w}$ nel Game 2.

Pertanto, la trasformazione tra $(\mathbf{y}_1, \mathbf{y}_2)$ e $(\mathbf{z}_1, \mathbf{z}_2)$ è una semplice traslazione deterministica, invertibile una volta fissato c e il witness $(\mathbf{s}_1, \mathbf{s}_2)$. Ne consegue che la distribuzione delle variabili osservabili dall'avversario rimane invariata. Pertanto si ha che

$$\varepsilon_3 = \varepsilon_2.$$

Game 4. $\mathcal{S}_1$ campiona direttamente

$$\mathbf{z}_2 \leftarrow D^{m_2\hat{d}}_{\mathbf{0},\sigma_2}$$

e accetta con probabilità $1/M_2$.

Dal teorema 2.2.1, segue che la distribuzione risultante è indistinguibile da quella originale:

$$|\varepsilon_4 - \varepsilon_3| = \text{negl}(\lambda).$$

Game 5. $\mathcal{S}_1$ sostituisce i commitment con valori della forma:

$$\begin{bmatrix} \mathbf{t}_A \\ \mathbf{t}_y \\ \mathbf{t}_g \\ t \end{bmatrix} = \boldsymbol{\nu} + \begin{bmatrix} \mathbf{A}\mathbf{s}_1 \\ \mathbf{y}_3 \\ \mathbf{g} \\ f_1 \end{bmatrix},$$

dove $\boldsymbol{\nu}$ è uniforme.

Se esistesse un algoritmo che possa distinguere le differenze tra questo gioco e il precedente, si potrebbe costruire un avversario che viola l'assunzione MLWE. Pertanto:

$$|\varepsilon_5 - \varepsilon_4| = \text{negl}(\lambda).$$

Game 6. $\mathcal{S}_1$ campiona direttamente:

$$(\mathbf{t}_A, \mathbf{t}_y, \mathbf{t}_g, t) \leftarrow \hat{\mathcal{R}}_{\hat{q}}^{n+256/\hat{d}+\tau+1}.$$

Questa modifica non altera la distribuzione perché nel gioco precedente questi valori erano già uniformemente distribuiti su $\hat{\mathcal{R}}_{\hat{q}}^{n+256/\hat{d}+\tau+1}$ grazie alla casualità delle variabili da cui dipendevano (come, ad esempio $\mathbf{s}_i$ e $\mathbf{y}_3$). Possiamo quindi concludere che:

$$\varepsilon_6 = \varepsilon_5.$$

Game 7. $\mathcal{S}_1$ campiona direttamente

$$\mathbf{z}_1 \leftarrow D_{\mathbf{0},\sigma_1}^{m_1\hat{d}}$$

e applica rejection sampling. Si ha per la seconda conseguenza del lemma A.1.1:

$$|\varepsilon_7 - \varepsilon_6| \leq N2^{-\lambda} = \text{negl}(\lambda).$$

Game 8. $\mathcal{S}_1$ campiona direttamente

$$h_i \leftarrow \{x \in \hat{\mathcal{R}}_{\hat{q}} : \tilde{x} = 0\}.$$

Poiché $\mathbf{g}$ era già uniforme nello stesso insieme:

$$\varepsilon_8 = \varepsilon_7.$$

Game 9. $\mathcal{S}_1$ campiona direttamente

$$\vec{z}_3 \leftarrow D_{\vec{0},\sigma_3}^{256}$$

e applica rejection sampling. Di nuovo, per il lemma A.1.1 si ottiene:

$$|\varepsilon_9 - \varepsilon_8| = \text{negl}(\lambda).$$

Conclusione. Nel gioco finale, il simulatore $\mathcal{S}_1$ genera una prova senza utilizzare il witness w.

Combinando tutti i passaggi, si ottiene:

$$|\varepsilon_9 - \varepsilon| = \text{negl}(\lambda),$$

da cui segue che le distribuzioni dei transcript reali e simulati sono indistinguibili.

Pertanto, il protocollo $\Pi^{\text{ISIS}}_{\text{NIZK}}$ è zero-knowledge. □

A.2 Non Interactive Zero Knowledge Proof $\Pi^{\text{Com}}_{\text{NIZK}}$

In questa sezione costruiamo una $\Pi^{\text{Com}}_{\text{NIZK}}$ multi-proof extractable per dimostrare la conoscenza di un vettore corto $\vec{s} \in \mathbb{Z}_{\hat{q}}^m$ tale che:

$$\|\vec{s}\| \leq B \quad \text{e} \quad P\vec{s} = \vec{u} \ \text{ in } \mathbb{Z}_{\hat{q}},$$

dove $P \in \mathbb{Z}_{\hat{q}}^{n \times m}$ e $\vec{u} \in \mathbb{Z}_{\hat{q}}^n$ sono pubblici.

Definiamo quindi la relazione:

$$R_{\text{Com}} := \left\{ (x, w) : \begin{array}{c} x = (P, \vec{u}),\ w = \vec{s}, \\ P \in \mathbb{Z}_{\hat{q}}^{n \times m},\ \vec{u} \in \mathbb{Z}_{\hat{q}}^n,\ \vec{s} \in \mathbb{Z}_{\hat{q}}^m, \\ P\vec{s} = \vec{u} \ \wedge \ \|\vec{s}\| \leq B \end{array} \right\}. \tag{A.4}$$

Seguiamo l'approccio di Lyubashevsky in Bootle et al. [2023] combinato con la trasformazione di Katsumata per ottenere multi-proof extractability.

Per il teorema della somma di quattro quadrati, la condizione $\|\vec{s}\| \leq B$ è equivalente all'esistenza di $a_1, a_2, a_3, a_4 \in \mathbb{Z}$ tali che:

$$B^2 - \|\vec{s}\|^2 = a_1^2 + a_2^2 + a_3^2 + a_4^2.$$

Definiamo quindi:

$$\vec{a} := (a_1, a_2, a_3, a_4, 0, \ldots, 0) \in \mathbb{Z}_{\hat{q}}^{\hat{d}}, \quad \vec{s} := (\vec{s}, \vec{a}) \in \mathbb{Z}_{\hat{q}}^{m+1}.$$

Ponendo opportunamente:

$$P := [P \mid 0],$$

otteniamo:

$$\|\vec{s}\| = B \quad \text{e} \quad P\vec{s} = \vec{u}. \tag{A.5}$$

Il CRS è definito come:

$$\text{crs} := (\mathbf{A}_1, \mathbf{A}_2, \mathbf{B}_y, \mathbf{B}_g, \mathbf{b}), \tag{A.6}$$

dove le matrici sono uniformemente casuali negli opportuni spazi su $\hat{\mathcal{R}}_{\hat{q}}^{n \times m_1} \times \hat{\mathcal{R}}_{\hat{q}}^{n \times m_2} \times \hat{\mathcal{R}}_{\hat{q}}^{256/d \times \hat{m}_2} \times \hat{\mathcal{R}}_{\hat{q}}^{\tau \times m_2} \times \hat{\mathcal{R}}_{\hat{q}}^{m_2}$.

La dichiarazione è:

$$x := (P, \vec{u}, B).$$

Primo round. Sia $\mathbf{s} = \text{Coeffs}^{-1}(\vec{s})$. Il prover:

- campiona $\mathbf{s}_2 \leftarrow \chi^{m_2}$;
- calcola il commitment:
$$\mathbf{t}_A = \mathbf{A}_1\mathbf{s}_1 + \mathbf{A}_2\mathbf{s}_2;$$
- campiona $\mathbf{y}_1, \mathbf{y}_2$ da $D^{m_i}_{\mathbf{0},\sigma_i}$ e pone:
$$\mathbf{w} = \mathbf{A}_1\mathbf{y}_1 + \mathbf{A}_2\mathbf{y}_2;$$
- campiona $\mathbf{y}_3 \leftarrow D^{256}_{\mathbf{0},\sigma_3}$ e calcola:
$$\mathbf{t}_y = \mathbf{B}_y\mathbf{s}_2 + \mathbf{y}_3 \quad \mathbf{B}_y \in \hat{\mathcal{R}}_{\hat{q}}^{256/\hat{d}\times m_2};$$
- campiona $\mathbf{g}$ tale che $\tilde{g} = 0$ e calcola:
$$\mathbf{t}_g = \mathbf{B}_g\mathbf{s}_2 + \mathbf{g}. \quad \mathbf{B}_g \in \hat{\mathcal{R}}_{\hat{q}}^{\tau\times m_2}$$

Il primo messaggio è:
$$a_1 := (\mathbf{t}_A, \mathbf{t}_y, \mathbf{t}_g, \mathbf{w}),$$
e la challenge:
$$(\mathfrak{R}_0, \mathfrak{R}_1) = H(1, \text{crs}, x, a_1).$$

Secondo round. Poniamo $\mathfrak{R} := \mathfrak{R}_0 - \mathfrak{R}_1$.

Il prover calcola:
$$\vec{z}_3 := \vec{y}_3 + \mathfrak{R}\vec{s}_1 \in \mathbb{Z}_{\hat{q}}^{256},$$
dove $\vec{y}_3 = \text{Coeffs}(\mathbf{y}_3)$ e $\vec{s}_1 = \text{Coeffs}(\mathbf{s}_1)$.

Applica quindi rejection sampling su $\vec{z}_3$.

Il secondo messaggio è:
$$\mathbf{a}_2 := \vec{z}_3,$$
e la corrispettiva challenge è:
$$(\gamma_{i,j})_{i\in[\tau],j\in[256+n+1]} = H(2, \text{crs}, x, a_1, a_2).$$

Terzo round. L'obiettivo del prover è dimostrare le seguenti relazioni:
$$\begin{cases} \vec{z}_3 = \vec{y}_3 + \mathfrak{R}\vec{s}_1, \\ P\vec{s}_1 = \vec{u}, \\ \langle \vec{s}_1, \vec{s}_1 \rangle = B^2. \end{cases}$$

Tali relazioni vengono trasformate esattamente come nella sezione A.1 tramite l'automorfismo σ.

Per ogni $i \in [\tau]$, il prover definisce:

$$\begin{aligned} h_i := \mathbf{g}_i &+ \sum_{j=1}^{256} \gamma_{i,j} \cdot \Big(\sigma(\mathbf{r}_j)^T \mathbf{s}_1 + \sigma(\mathbf{e}_j)^T \mathbf{y}_3 - \vec{z}_{3,j} \Big) \\ &+ \sum_{j=1}^{n} \gamma_{i,256+j} \cdot \Big(\sigma(\mathbf{p}_j)^T \mathbf{s}_1 - \sigma(\mathbf{e}'_j)^T \mathbf{u} \Big) \\ &+ \gamma_{i,256+n+1} \cdot \Big(\sigma(\mathbf{s}_1)^T \mathbf{s}_1 - B^2 \Big). \end{aligned} \tag{A.7}$$

Per costruzione, i coefficienti costanti di h_i $\tilde{h}_i$ sono nulli.
Il terzo messaggio è:

$$a_3 := (h_1, \ldots, h_\tau),$$

e la challenge successiva:

$$\boldsymbol{\mu} = (\mu_i)_{i \in [\tau]} = H(3, \mathrm{crs}, x, a_1, a_2, a_3).$$

Quarto round. L'obiettivo del prover è dimostrare le τ equazioni quadratiche su $\hat{\mathcal{R}}_{\hat{q}}$ definite in A.2.

A tal fine, le equazioni vengono combinate linearmente utilizzando i coefficienti μ_i, ottenendo un'unica equazione:

$$\begin{aligned} 0 = \sum_{i=1}^{\tau} \mu_i \Bigg(&\sum_{j=1}^{256} \gamma_{i,j} (\sigma(\mathbf{r}_j)^T \mathbf{s}_1 + \sigma(\mathbf{e}_j)^T \mathbf{y}_3 - z_{3,j}) \\ &+ \sum_{j=1}^{n} \gamma_{i,256+j} (\sigma(\mathbf{p}_j)^T \mathbf{s}_1 - \sigma(\mathbf{e}'_j)^T \mathbf{u}) \\ &+ \gamma_{i,256+n+1} (\sigma(\mathbf{s}_1)^T \mathbf{s}_1 - B^2) + \mathbf{g}_i - h_i \Bigg). \end{aligned} \tag{A.8}$$

Definiamo:

$$\mathbf{B} := \begin{bmatrix} \mathbf{B}_y \\ \mathbf{B}_g \end{bmatrix}, \quad \mathbf{t}_B := \begin{bmatrix} \mathbf{t}_y \\ \mathbf{t}_g \end{bmatrix}, \quad \mathbf{m} := \begin{bmatrix} \mathbf{y}_3 \\ \mathbf{g} \end{bmatrix} \in \hat{\mathcal{R}}_{\hat{q}}^{256/\hat{d}+\tau},$$

e

$$\hat{\mathbf{s}} := \begin{bmatrix} \mathbf{s}_1 \\ \sigma(\mathbf{s}_1) \\ \mathbf{m} \\ \sigma(\mathbf{m}) \end{bmatrix}.$$

Allora la relazione può essere scritta come:

$$\hat{\mathbf{s}}^T \mathbf{D}_2 \hat{\mathbf{s}} + \mathbf{d}_1^T \hat{\mathbf{s}} + d_0 = 0$$

dove la matrice $\mathbf{D}_2$ è data da:

$$\mathbf{D}_2 = \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \sum_{i=1}^{\tau} \mu_i \gamma_{i,256+n+1} \mathbf{I}_{m_1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}$$

il termine lineare $\mathbf{d}_1$ è:

$$\mathbf{d}_1 = \begin{bmatrix} \sum_{i=1}^{\tau} \mu_i \Big(\sum_{j=1}^{256} \gamma_{i,j} \sigma(\mathbf{r}_j) + \sum_{j=1}^{n} \gamma_{i,256+j} \sigma(\mathbf{p}_j) \Big) \\ \mathbf{0} \\ \sum_{i=1}^{\tau} \mu_i \Big(\sum_{j=1}^{256} \gamma_{i,j} \sigma(\mathbf{e}_j) \Big) \\ \boldsymbol{\mu} \\ \mathbf{0} \end{bmatrix}.$$

Il termine costante è:

$$d_0 = -\sum_{i=1}^{\tau} \mu_i \Big(\sum_{j=1}^{256} z_{3,j} + \sum_{j=1}^{n} \sigma(\mathbf{e}'_j)^T \mathbf{u} + \gamma_{i,256+n+1} B^2 + h_i \Big).$$

Il prover esegue ora il sottoprotoccolo per equazioni quadratiche.

Campiona:

$$\mathbf{y} := \begin{bmatrix} \mathbf{y}_1 \\ \sigma(\mathbf{y}_1) \\ -\mathbf{B}\mathbf{y}_2 \\ -\sigma(\mathbf{B}\mathbf{y}_2) \end{bmatrix} \in \hat{\mathcal{R}}_{\hat{q}}^{2m_1+512/\hat{d}+2\tau},$$

e calcola:

$$f_1 = \hat{\mathbf{s}}^T \mathbf{R}_2 \mathbf{y} + \mathbf{y}^T \mathbf{R}_2 \hat{\mathbf{s}} + \mathbf{r}_1^T \mathbf{y}.$$

Il commitment di f_1 è:

$$t = \mathbf{b}^T \mathbf{s}_2 + f_1,$$

e definisce:

$$f_0 = \mathbf{y}^T \mathbf{R}_2 \mathbf{y} + \mathbf{b}^T \mathbf{y}_2.$$

Il quarto messaggio è:

$$a_4 := (t, f_0),$$

e la challenge finale:

$$c = H(4, \mathrm{crs}, x, a_1, a_2, a_3, a_4).$$

Round finale. Il prover calcola:

$$\mathbf{z}_1 = c\mathbf{s}_1 + \mathbf{y}_1, \quad \mathbf{z}_2 = c\mathbf{s}_2 + \mathbf{y}_2,$$

e applica rejection sampling.

Il messaggio finale è:

$$a_5 := (\mathbf{z}_1, \mathbf{z}_2).$$

La prova è:

$$\pi := (a_1, a_2, a_3, a_4, a_5).$$

Verifica. Il verificatore ricostruisce tutte le challenge e definisce:

$$\mathbf{z} := \begin{bmatrix} \mathbf{z}_1 \\ \sigma(\mathbf{z}_1) \\ c\mathbf{t}_B - \mathbf{B}\mathbf{z}_2 \\ \sigma(c\mathbf{t}_B - \mathbf{B}\mathbf{z}_2) \end{bmatrix}.$$

Accetta se e solo se:

$$\begin{cases} \|\mathbf{z}_1\| \leq B_1, \\ \|\mathbf{z}_2\| \leq B_2, \\ \|\vec{z}_3\| \leq B_3, \\ \tilde{h}_i = 0 \ \ \forall i \in [\tau], \\ \mathbf{A}_1\mathbf{z}_1 + \mathbf{A}_2\mathbf{z}_2 = \mathbf{w} + c\mathbf{t}_A, \\ \mathbf{z}^T\mathbf{D}_2\mathbf{z} + c\mathbf{d}_1^T\mathbf{z} + c^2 d_0 - (ct - \mathbf{b}^T\mathbf{z}_2) = f_0. \end{cases}$$

A.2.1 Multi-Proof Extractability e Trasformazione di Katsumata

Dopo aver definito $\Pi_{\mathsf{NIZK}}^{\mathsf{Com}}$, introduciamo ora una proprietà di sicurezza più forte, nota come *multiproof extractability*. Tale proprietà assicura che, anche nel caso in cui un avversario produca simultaneamente molteplici proof valide, sia possibile estrarre in maniera efficiente i corrispondenti witness per tutte le istanze accettate.

Definizione A.2.1 (Multi-Proof Extractability)**.** *Sia Π_{NIZK} un sistema NIZK nel modello Random Oracle. Richiediamo che valgano le seguenti proprietà.*
CRS Simulatability. *Per ogni avversario PPT $\mathcal{A}$, si ha:*

$$\mathsf{Adv}^{\mathsf{crs}}_{\Pi_{\mathsf{NIZK}}}(\mathcal{A}) := |\mathbb{P}\left[\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda) : 1 \leftarrow \mathcal{A}^H(\mathsf{crs})\right] - \mathbb{P}\left[(\widetilde{\mathsf{crs}}, \tau) \leftarrow \mathsf{S}_{\mathsf{crs}}(1^\lambda) : 1 \leftarrow \mathcal{A}^H(\widetilde{\mathsf{crs}})\right]| = \mathsf{negl}(\lambda).$$

Straight-Line Multiproof Extractability. *Esistono costanti e_1, e_2, c tali che, per ogni $Q_H, Q_s \in \mathrm{poly}(\lambda)$ e per ogni avversario PPT $\mathcal{A}$ che effettua al più Q_H query all'oracolo casuale, se vale:*

$$\mathbb{P}\left[\begin{array}{l} (\widetilde{\mathsf{crs}}, \tau) \leftarrow \mathsf{S}_{\mathsf{crs}}(1^\lambda), \\ \{(x_i, \pi_i)\}_{i\in[Q_s]} \leftarrow \mathcal{A}^H(\widetilde{\mathsf{crs}}) : \\ \forall i \in [Q_s],\ \mathsf{Verify}^H(\widetilde{\mathsf{crs}}, x_i, \pi_i) = 1 \end{array}\right] \geq \varepsilon(\lambda),$$

dove $\varepsilon(\lambda)$ è non trascurabile, allora:

$$\mathbb{P}\left[\begin{array}{l} (\widetilde{\mathsf{crs}}, \tau) \leftarrow \mathsf{S}_{\mathsf{crs}}(1^\lambda), \\ \{(x_i, \pi_i)\}_{i\in[Q_s]} \leftarrow \mathcal{A}^H(\widetilde{\mathsf{crs}}), \\ \{w_i \leftarrow \mathsf{MultiExtract}(Q_H, Q_s, 1/\varepsilon, \widetilde{\mathsf{crs}}, \tau, x_i, \pi_i)\}_{i\in[Q_s]} : \\ \forall i \in [Q_s],\ (\widetilde{\mathsf{crs}}, x_i, w_i) \in R \ \wedge\ \mathsf{Verify}^H(\widetilde{\mathsf{crs}}, x_i, \pi_i) = 1 \end{array}\right] \geq \frac{1}{2}\varepsilon(\lambda) - \mathsf{negl}(\lambda),$$

dove il tempo di esecuzione dell'estrattore è limitato superiormente da:

$$Q_H^{e_1} \cdot Q_s^{e_2} \cdot \left(\frac{1}{\varepsilon(\lambda)}\right)^c \cdot \mathrm{poly}(\lambda).$$

Questa proprietà garantisce quindi che, a partire da un avversario che produce simultaneamente più proof valide con probabilità non trascurabile, sia possibile costruire un estrattore capace di recuperare tutti i witness corrispondenti con probabilità non trascurabile.
Al fine di rendere la proof $\Pi^{\mathsf{Com}}_{\mathsf{NIZK}}$ *multi-proof extractable*, applichiamo la trasformazione di Katsumata Katsumata [2021].
Per rendere il protocollo *straight-line extractable*, utilizziamo un *extractable linear homomorphic commitment* (LHC), che può essere visto come uno schema di cifratura linearmente omomorfo con chiavi pubbliche pseudocasuali che consente di cifrare sia $\mathbf{s}_i$ sia $\mathbf{y}_i$. Fissato $i \in \{1,2\}$, la chiave di commitment (CRS) è data da: $\mathsf{crs}_i^{\mathsf{LHC}} = (\bar{\mathbf{A}}_i, \bar{\mathbf{B}}_i)$, dove $\bar{\mathbf{A}}_i, \bar{\mathbf{B}}_i$ sono matrici uniformemente distribuite in $\mathcal{R}_{\hat{q}}^{m_i \times n_i}$. Il costrutto LHC definisce tre algoritmi: $\mathsf{LHC.Com}_i$, $\mathsf{LHC.Open}_i$, $\mathsf{LHC.Verify}_i$.

Algoritmo 16: $\mathrm{LHC.Com}_i$

Input: Input pubblico: $\mathrm{crs}^{\mathrm{LHC}} = (\bar{\mathbf{A}}, \bar{\mathbf{B}})$
Input: $\mathbf{s}_i, \mathbf{y}_i$
Output: $(\mathbf{com}, \mathbf{st})$
begin
$(\bar{\mathbf{e}}_{i,1}, \bar{\mathbf{e}}_{i,2}, \bar{\mathbf{e}}_{i,3}) \leftarrow S_{\eta_i}^{n_i} \times S_{\eta_i}^{m_i} \times S_{\eta_i}^{m_i}$;
$\bar{\mathbf{t}}_{i,1} = \bar{p} \cdot (\bar{\mathbf{A}}\bar{\mathbf{e}}_{i,1} + \bar{\mathbf{e}}_{i,2})$;
$\bar{\mathbf{t}}_{i,2} = \bar{p} \cdot (\bar{\mathbf{B}}\bar{\mathbf{e}}_{i,1} + \bar{\mathbf{e}}_{i,3}) + s_i$;
$(\bar{\mathbf{f}}_{i,1}, \bar{\mathbf{f}}_{i,2}, \bar{\mathbf{f}}_{i,3}) \leftarrow D_{\mathbf{0},\bar{\sigma}_i}^{n_i} \times D_{\mathbf{0},\bar{\sigma}_i}^{m_i} \times D_{\mathbf{0},\bar{\sigma}_i}^{m_i}$;
$\bar{\mathbf{w}}_{i,1} = \bar{p} \cdot (\bar{\mathbf{A}}\bar{\mathbf{f}}_{i,1} + \bar{\mathbf{f}}_{i,2})$;
$\bar{\mathbf{w}}_{i,2} = \bar{p} \cdot (\bar{\mathbf{B}}\bar{\mathbf{f}}_{i,1} + \bar{\mathbf{f}}_{i,3}) + y_i$;
$\mathbf{com} = (\bar{\mathbf{t}}_{i,1}, \bar{\mathbf{t}}_{i,2}, \bar{\mathbf{w}}_{i,1}, \bar{\mathbf{w}}_{i,2})$;
$\mathbf{st} = (\bar{\mathbf{e}}_{i,1}, \bar{\mathbf{e}}_{i,2}, \bar{\mathbf{e}}_{i,3}, \bar{\mathbf{f}}_{i,1}, \bar{\mathbf{f}}_{i,2}, \bar{\mathbf{f}}_{i,3})$;
return $(\mathbf{com}, \mathbf{st})$;

Algoritmo 17: $\mathrm{LHC.Open}_i$

Input: Input: $(\mathbf{com}, c), \mathbf{st}$
Output: **op**
begin
for $j \in \{1,2,3\}$ **do**
$\bar{\mathbf{z}}_{i,j} = \bar{\mathbf{f}}_{i,j} + c \cdot \bar{\mathbf{e}}_{i,j}$;
$b \leftarrow \mathrm{Rej}([\bar{\mathbf{z}}_{i,1} \| \bar{\mathbf{z}}_{i,2} \| \bar{\mathbf{z}}_{i,3}], 4, c \cdot [\bar{\mathbf{e}}_{i,1} \| \bar{\mathbf{e}}_{i,2} \| \bar{\mathbf{e}}_{i,3}], \bar{\sigma}_i, \bar{\mathbf{M}}_i)$;
if $b = 0$ **then**
return op $= \perp$;
else
return op $= [\bar{\mathbf{z}}_{i,1} \| \bar{\mathbf{z}}_{i,2} \| \bar{\mathbf{z}}_{i,3}]$;

Algoritmo 18: LHC.Verify$_i$

Input: Input: $(\mathbf{com}, c, (\mathbf{z}, \mathbf{op} \neq \perp))$
Output: Bit b
begin
  $(\bar{\mathbf{t}}_{i,1}, \bar{\mathbf{t}}_{i,2}, \bar{\mathbf{w}}_{i,1}, \bar{\mathbf{w}}_{i,2}) \leftarrow \mathbf{com}$;
  $[\bar{\mathbf{z}}_{i,1} \| \bar{\mathbf{z}}_{i,2} \| \bar{\mathbf{z}}_{i,3}] \leftarrow \mathbf{op}$;
  if $\|\bar{\mathbf{z}}_{i,1}\| > \bar{\sigma_i}\sqrt{2n_i d}$ **then**
    return 0;
  for $j \in \{2,3\}$ **do**
    if $\|\bar{\mathbf{z}}_{i,j}\| > \bar{\sigma_i}\sqrt{2m_i \hat{d}}$ **then**
      return 0;
  $\bar{\mathbf{z}}_A = c\bar{\mathbf{t}}_{i,1} + \bar{\mathbf{w}}_{i,1} - \bar{p} \cdot (\bar{\mathbf{A}}\bar{\mathbf{z}}_{i,1} + \bar{\mathbf{z}}_{i,2})$;
  $\bar{\mathbf{z}}_B = c\bar{\mathbf{t}}_{i,2} + \bar{\mathbf{w}}_{i,2} - \bar{p} \cdot (\bar{\mathbf{B}}\bar{\mathbf{z}}_{i,1} + \bar{\mathbf{z}}_{i,3})$;
  if $\bar{\mathbf{z}}_A \neq 0 \vee \bar{\mathbf{z}}_B \neq \mathbf{z}$ **then**
    return 0;
  return 1;

Nel seguito, sia $p < \hat{q}$ un intero dispari coprimo con $\hat{q}$.

Riformuliamo ora la proof

$$\Pi^{\mathsf{Com}}_{\mathsf{NIZK}} := (\mathsf{Prove}^{\mathsf{Com}}_H, \mathsf{Verify}^{\mathsf{Com}}_H)$$

per la relazione R_{Com}, come segue:

Algoritmo 19: $\mathsf{Prove}^{\mathsf{Com}}_{H}$

Input: Input pubblico: $\text{crs} = (\mathbf{A}_1, \mathbf{A}_2, \mathbf{B}_y, \mathbf{B}_g, \mathbf{b}, \bar{\mathbf{A}}_1, \bar{\mathbf{A}}_2, \bar{\mathbf{B}}_1, \bar{\mathbf{B}}_2)$, $x = (P, \vec{u}, B)$
Input: $w = \vec{s}$
Output: Prova π
begin
$\mathbf{rst} \leftarrow 0, \quad \mathbf{idx} \leftarrow 0, \quad \mathbf{s}_1 = \mathbf{s}$;
while $\mathbf{rst} = 0 \wedge \mathbf{idx} < N$ **do**
$\mathbf{idx} \leftarrow \mathbf{idx} + 1$;
$\mathbf{s}_2 \leftarrow \chi^{m_2}$;
$\mathbf{t}_A = \mathbf{A}_1\mathbf{s}_1 + \mathbf{A}_2\mathbf{s}_2$;
$\mathbf{y}_1 \leftarrow D^{m_1\hat{d}}_{\mathbf{0},\sigma_1}, \ \mathbf{y}_2 \leftarrow D^{m_2\hat{d}}_{\mathbf{0},\sigma_2}, \ \mathbf{y}_3 \leftarrow D^{256}_{0,\sigma_3}$;
$\mathbf{g} \leftarrow \{x \in \hat{\mathcal{R}}_{\hat{q}} : \tilde{x} = 0\}^{\tau}$;
$\mathbf{w} = \mathbf{A}_1\mathbf{y}_1 + \mathbf{A}_2\mathbf{y}_2$;
$\mathbf{t}_y = \mathbf{B}_y\mathbf{s}_2 + \mathbf{y}_3, \quad \mathbf{t}_g = \mathbf{B}_g\mathbf{s}_2 + \mathbf{g}$;
$(\mathbf{com}_1, \mathbf{st}_1) \leftarrow \text{LHC.Com}_1((\bar{\mathbf{A}}_1, \bar{\mathbf{B}}_1), \mathbf{s}_1, \mathbf{y}_1)$;
$(\mathbf{com}_2, \mathbf{st}_2) \leftarrow \text{LHC.Com}_2((\bar{\mathbf{A}}_2, \bar{\mathbf{B}}_2), \mathbf{s}_2, \mathbf{y}_2)$;
$a_1 = (\mathbf{t}_A, \mathbf{t}_y, \mathbf{t}_g, \mathbf{w}, \mathbf{v}, \mathbf{com}_1, \mathbf{com}_2)$;
$(\mathfrak{R}_0, \mathfrak{R}_1) = H(1, \text{crs}, x, a_1)$;
$\mathfrak{R} = \mathfrak{R}_0 - \mathfrak{R}_1$;
$\vec{s}_1 = \text{Coeffs}(\mathbf{s}_1), \quad \vec{y}_3 = \text{Coeffs}(\mathbf{y}_3)$;
$\vec{z}_3 = \vec{y}_3 + \mathfrak{R}\vec{s}_1$;
$b_3 \leftarrow \text{Rej}(\vec{z}_3, \mathfrak{R}\vec{s}_1, \sigma_3, M_3)$;
$a_2 = \vec{z}_3$;
$(\gamma_{i,j})_{i\in[\tau],j\in[256+n+1]} = H(2, \text{crs}, x, a_1, a_2)$;
for $i \in [\tau]$ **do**
calcola h_i;
$a_3 = h = (h_1, \ldots, h_\tau)$;
$(\mu_i)_{i\in[\tau]} = H(3, \text{crs}, x, a_1, a_2, a_3)$;
calcola $\mathbf{B}, \mathbf{t}_B, \hat{\mathbf{s}}, \mathbf{y}, \mathbf{m}$;
calcola $\mathbf{D}_2, \mathbf{d}_1, d_0$;
$f_1 = \mathbf{s}^T\mathbf{R}_2\mathbf{y} + \mathbf{y}^T\mathbf{R}_2\mathbf{s} + \mathbf{r}_1^T\mathbf{y}$;
$f_0 = \mathbf{y}^T\mathbf{R}_2\mathbf{y} + \mathbf{b}^T\mathbf{y}_2$;
$t = \mathbf{b}^T\mathbf{s}_2 + f_1$;
$a_4 = (t, f_0)$;
$c = H(4, \text{crs}, x, a_1, a_2, a_3, a_4)$;
for $i \in \{1,2\}$ **do**
$\mathbf{z}_i = \mathbf{y}_i + c\mathbf{s}_i$;
$b_i \leftarrow \text{Rej}(\mathbf{z}_i, c\mathbf{s}_i, \sigma_i, M_i)$;
$\mathbf{op}_i \leftarrow \text{LHC.Open}_i((\bar{\mathbf{A}}_i, \bar{\mathbf{B}}_i), (\mathbf{com}_i, c), \mathbf{st}_i)$;
$\bar{b}_i = \begin{cases} 0 & \text{if } \mathbf{op}_i = \perp \\ 1 & \text{otherwise} \end{cases}$;
$\pi = (\mathbf{t}_A, \mathbf{t}_y, \mathbf{t}_g, \mathbf{w}, \mathbf{com}_1, \mathbf{com}_2, \vec{z}_3, \mathbf{h}, t, f_0, \mathbf{z}_1, \mathbf{z}_2, \mathbf{op}_1, \mathbf{op}_2)$;
$\mathbf{rst} = b_1b_2b_3\bar{b}_1\bar{b}_2$;
if $\mathbf{rst} = 1$ **then**
return π;
return $\perp$;

Algoritmo 20: $\mathsf{Verify}_H^{\mathsf{Com}}$

Input: Input pubblico: $\mathrm{crs} = (\mathbf{A}_1, \mathbf{A}_2, \mathbf{B}_y, \mathbf{B}_g, \mathbf{b}, \bar{\mathbf{A}}_1, \bar{\mathbf{A}}_2, \bar{\mathbf{B}}_1, \bar{\mathbf{B}}_2)$, $x = (P, \vec{u}, B)$
Input: prova π
Output: Bit b
begin
$\pi = (\mathbf{t}_A, \mathbf{t}_y, \mathbf{t}_g, \mathbf{w}, \mathbf{com}_1, \mathbf{com}_2, \vec{z}_3, \mathbf{h}, t, f_0, \mathbf{z}_1, \mathbf{z}_2, \mathbf{op}_1, \mathbf{op}_2)$;
$a_1 = (\mathbf{t}_A, \mathbf{t}_y, \mathbf{t}_g, \mathbf{w}, \mathbf{com}_1, \mathbf{com}_2)$;
$a_2 = \vec{z}_3$;
$a_3 = \mathbf{h}$;
$a_4 = (t, f_0)$;
$(\mathfrak{R}_0, \mathfrak{R}_1) \leftarrow H(1, \mathrm{crs}, x, a_1)$;
$(\gamma_{i,j})_{i\in[\tau], j\in[256+n+1]} \leftarrow H(2, \mathrm{crs}, x, a_1, a_2)$;
$(\mu_i)_{i\in[\tau]} \leftarrow H(3, \mathrm{crs}, x, a_1, a_2, a_3)$;
$c \leftarrow H(4, \mathrm{crs}, x, a_1, a_2, a_3, a_4)$;
$\mathbf{B} = \begin{bmatrix} \mathbf{B}_y \\ \mathbf{B}_g \end{bmatrix}, \quad \mathbf{t}_B = \begin{bmatrix} \mathbf{t}_y \\ \mathbf{t}_g \end{bmatrix}$;
$\mathbf{z} = \begin{bmatrix} \mathbf{z}_1 \\ \sigma(\mathbf{z}_1) \\ c\mathbf{t}_B - \mathbf{B}\mathbf{z}_2 \\ \sigma(c\mathbf{t}_B - \mathbf{B}\mathbf{z}_2) \end{bmatrix}$;
calcola $\mathbf{D}_2, \mathbf{d}_1, d_0$;
if $\|\mathbf{z}_1\| \le B_1 := \sigma_1\sqrt{2m_1\hat{d}}$ ***and***
$\|\mathbf{z}_2\| \le B_2 := \sigma_2\sqrt{2m_2\hat{d}}$ ***and***
$\|\vec{z}_3\| \le B_3 := 1.7\sqrt{256}$ ***and***
$\tilde{h}_i = 0$ *per* $i \in [\tau]$ ***and***
$\mathbf{A}_1\mathbf{z}_1 + \mathbf{A}_2\mathbf{z}_2 = \mathbf{w} + c\mathbf{t}_A$ ***and***
$\mathbf{z}^T\mathbf{D}_2\mathbf{z} + c\mathbf{d}_1^T\mathbf{z} + c^2 d_0 - (ct - \mathbf{b}^T\mathbf{z}_2) = f_0$ ***and***
$\mathrm{LHC.Verify}_i((\bar{\mathbf{A}}_i, \bar{\mathbf{B}}_i), c, (\mathbf{z}_i, \mathbf{op}_i)) = 1$ *per* $i \in \{1,2\}$ **then**
return 1;
else
return 0;

Teorema A.2.1. *Siano* $\alpha_1, \alpha_2, \alpha_3, \bar{\alpha}_1, \bar{\alpha}_2 \in O(\sqrt{\lambda})$ *e* $N, \hat{d} \in O(\lambda)$. *Siano* $\omega_{\min}(\lambda)$ *e* $\omega_{\max}(\lambda)$ *le funzioni definite in A.1.3. Sia* $\nu > 0$ *un parametro di scaling, definiamo:*

$$\sigma_1 := \alpha_1\nu B, \qquad \sigma_2 := \alpha_2\nu\sqrt{m_2\hat{d}}, \qquad \sigma_3 := \alpha_3\,\omega_{\max}(\lambda)B,$$

$$\bar{\sigma_1} := \bar{\alpha}_1\eta_1\nu\sqrt{n_1 d + 2m_1 d}, \qquad \bar{\sigma}_2 := \bar{\alpha}_2\eta_2\nu\sqrt{n_2 d + 2m_2 d}$$

Definiamo i corrispondenti tassi di rifiuto del canonico algoritmo di rejection sampling:

$$M_i := \exp\left(\sqrt{\frac{2(\lambda+1)}{\log(e)}}\frac{1}{\alpha_i} + \frac{1}{2\alpha_i^2}\right) \quad \textit{per } i \in \{1,2,3\}.$$

$$\bar{M}_j := \exp\left(\sqrt{\frac{2(\lambda+1)}{\log(e)}}\frac{1}{\bar{\alpha}_j} + \frac{1}{2\bar{\alpha}_j^2}\right) \quad \textit{per } j \in \{1,2\}.$$

Allora, $\Pi_{\mathsf{Com}}^{\mathsf{NIZK}} := (\mathsf{Prove}_{\mathsf{Com}}^{H}, \mathsf{Verify}_{\mathsf{Com}}^{H})$ *è corretto.*

Dimostrazione. La dimostrazione di correttezza di $\mathsf{Prove}_{\mathsf{Com}}^{\mathsf{H}}$ viene svolta in modo analogo a quella di correttezza di $\mathsf{Prove}_{\mathsf{ISIS}}^{\mathsf{H}}$ contenuta nel teorema A.1.3. Mostriamo quindi che l'algoritmo del prover restituisce $\perp$ con probabilità trascurabile e che le equazioni di verifica sono soddisfatte con alta probabilità.

Correttezza del prover. Per come è stato definito lo spazio delle challenge nella sezione 4.6.4 si ha che le seguenti disuguaglianze sono soddisfatte con alta probabilità:

$$\|c\mathbf{s}_1\| \leq \nu B, \qquad \|c\mathbf{s}_2\| \leq \nu\sqrt{m_2\hat{d}}, \qquad \|\mathfrak{R}\vec{s}_1\| \leq \omega_{\max}(\lambda)B.$$

$$\|c[\mathbf{e}_{i,j}]_{j=1}^{3}\| \leq \eta_i\nu\sqrt{n_i d + 2m_i d} \quad i \in \{1,2\}$$

Applicando il lemma A.1.1 al rejection sampling, deduciamo che

$$\mathbb{P}[b_1b_2b_3\bar{b}_4\bar{b}_5 = 1] \geq \prod_i\left(\frac{1}{M_i} - \mathrm{negl}(\lambda)\right) \geq \frac{1}{M_1M_2M_3\bar{M}_4\bar{M}_5} - \mathrm{negl}(\lambda).$$

Pertanto, la probabilità che $\mathsf{Prove}_{\mathsf{Com}}^{\mathsf{H}}$ restituisca $\perp$, ovvero fallisca per N tentativi consecutivi, è minore o uguale a:

$$\left(1 - \frac{1}{M_1M_2M_3\bar{M}_4\bar{M}_5} + \mathrm{negl}(\lambda)\right)^N.$$

Poiché $M_1, M_2, M_3, \bar{M}_4, \bar{M}_5 \in O(1)$ e $N \in O(\lambda)$, segue che:

$$\left(1 - \frac{1}{M_1M_2M_3\bar{M}_4\bar{M}_5} + \mathrm{negl}(\lambda)\right)^N = \mathrm{negl}(\lambda),$$

da cui la probabilità di fallimento del prover è trascurabile.

Correttezza della verifica. Per quanto riguarda le equazioni di verifica, dal lemma 4.6.1 con parametro $t = \sqrt{2}$ otteniamo che:

$$\mathbb{P}\left[\|\mathbf{z}_1\| \leq B_1 \ \wedge \ \|\mathbf{z}_2\| \leq B_2\right] = \prod_i \mathbb{P}[\|\mathbf{z}_i\| \leq B_i] = \prod_i\left(1-\mathbb{P}[\|\mathbf{z}_i\| > B_i]\right) > \prod_i\left(1-(\sqrt{2}e^{-0.5})^{m_i}\right) = 1-\mathrm{negl}(\lambda).$$

Allo stesso modo si dimostra che

$$\mathbb{P}\left[\|\bar{\mathbf{z}}_{i,1}\| \leq \bar{\sigma}_i\sqrt{2n_i\hat{d}}\right] = 1 - \text{negl}(\lambda) \quad i \in \{1,2\}$$

$$\mathbb{P}\left[\|\bar{\mathbf{z}}_{i,j}\| \leq \bar{\sigma}_i\sqrt{2m_i\hat{d}}\right] = 1 - \text{negl}(\lambda) \quad (i,j) \in \{1,2\} \times \{2,3\}$$

Analogamente, applicando il lemma 4.6.1 con $t = 1.7$, otteniamo:

$$\mathbb{P}\left[\|\vec{z}_3\| \leq B_3\right] = 1 - \mathbb{P}\left[\|\vec{z}_3\| > B_3\right] > 1 - (1.7e^{0.5-1.7^2/2})^{256/\hat{d}} = 1 - \text{negl}(\lambda).$$

Le condizioni $\tilde{h}_i = 0$ per $i \in [\tau]$, $\mathbf{A}_1\mathbf{z}_1 + \mathbf{A}_2\mathbf{z}_2 = \mathbf{w} + c\mathbf{t}_A$, $\mathbf{z}^T\mathbf{D}_2\mathbf{z} + c\mathbf{d}_1^T\mathbf{z} + c^2d_0 - (ct - \mathbf{b}^T\mathbf{z}_2) = f_0$ vengono verificate in modo identico a quanto svolto nella dimostrazione del teorema A.1.3 e la correttezza della condizione $\text{LHC.Verify}_i((\bar{\mathbf{A}}_i, \bar{\mathbf{B}}_i), c, (\mathbf{z}_i, \mathbf{op}_i)) = 1$ per $i \in \{1,2\}$ segue dalla correttezza degli algoritmi LHC dimostrata in Katsumata [2021] □

A.2.2 Zero-Knowledge

Teorema A.2.2. *Il protocollo Π_{NIZK}^{Com} è zero-knowledge sotto l'assunzione di difficoltà dei problemi*

$$\mathsf{MLWE}_{n+256/\hat{d}+\tau+1,\, m_2,\, \chi,\, \hat{q}}, \mathsf{MLWE}_{2m_1,\, 2m_1+n_1,\, \eta_1}, \mathsf{MLWE}_{2m_2,\, 2m_2+n_2,\, \eta_2}.$$

Dimostrazione. La dimostrazione viene svolta in modo analogo a quella del teorema A.1.4. Costruiamo due algoritmi PPT $\mathcal{S}_0$ e $\mathcal{S}_1$, che condividono uno stato interno, tali che:

- $\mathcal{S}_0$ simula l'oracolo random H mediante *lazy sampling*;
- $\mathcal{S}_1$ simula il prover.

Indichiamo con ε_i la probabilità che $\mathcal{A}$ restituisca 1 nell'i-esimo gioco.

Game 1. $\mathcal{S}_0$ simula l'oracolo random, mentre $\mathcal{S}_1$ esegue onestamente $\mathsf{Prove}_{\text{Com}}^{H}(crs, x, w)$. Per costruzione:

$$\varepsilon_1 = \varepsilon.$$

Game 2. Quando $\mathcal{S}_1$ riceve una query, campiona una challenge $c \leftarrow \mathcal{C}$ all'inizio dell'esecuzione e la utilizza al posto dell'output dell'oracolo H nella fase finale.

Se la tupla $(4, crs, x, a_1, a_2, a_3, a_4)$ è già stata interrogata all'oracolo $\mathcal{S}_0$, allora $\mathcal{S}_1$ restituisce $\perp$.

L'unica differenza rispetto al gioco precedente è quindi la programmazione dell'oracolo random: adesso programmiamo direttamente c senza controllare se la tupla $a = (4, crs, x, a_1, a_2, a_3, a_4)$ fosse già stata impostata. In modo identico a quanto detto nel Game 2 della dimostrazione di A.1.4 si deduce che la probabilità di avere una collisione dopo Q_1 queries è al più $NQ_1(Q_0 + NQ_1)2^{-n\hat{d}}$. Da cui si conclude che

$$|\varepsilon_2 - \varepsilon_1| \leq NQ_1(Q_0 + NQ_1)2^{-n\hat{d}} + \text{negl}(\lambda) = \text{negl}(\lambda).$$

Game 3. $\mathcal{S}_1$ dopo aver campionato $c \leftarrow \mathcal{C}$ e $\mathbf{y}_1, \mathbf{y}_2$ calcola $\mathbf{z}_i = \mathbf{y}_i + c\mathbf{s}_i$ per $i \in \{1,2\}$ e avvia l'algoritmo $(\mathbf{com}_i, \mathbf{op}_i) \leftarrow \text{LHC.ZKSim}_i((\bar{\mathbf{A}}_i, \bar{\mathbf{B}}_i), c, \mathbf{z}_i)$. Siccome LHC è zero-knowledge si conclude che

$$|\varepsilon_3 - \varepsilon_2| \leq \text{negl}(\lambda).$$

Algoritmo 21: LHC.ZKSim_i

Input: Input: $(\text{crs}^{\text{LHC}}, c, \mathbf{z}_i)$
Output: Tuple $(\mathbf{com}, \mathbf{op})$
begin
  $(\bar{\mathbf{z}}_{i,1}, \bar{\mathbf{z}}_{i,2}, \bar{\mathbf{z}}_{i,3}) \leftarrow D^{n_i}_{\mathbf{0},\bar{\sigma}_i} \times D^{m_i}_{\mathbf{0},\bar{\sigma}_i} \times D^{m_i}_{\mathbf{0},\bar{\sigma}_i}$;
  $(\bar{\mathbf{t}}_{i,1}, \bar{\mathbf{t}}_{i,2}) \leftarrow \mathcal{R}^{m_i}_{\hat{q}} \times \mathcal{R}^{m_i}_{\hat{q}}$;
  $\bar{\mathbf{w}}_{i,1} = -c\bar{\mathbf{t}}_{i,1} + \bar{p} \cdot (\bar{\mathbf{A}}\bar{\mathbf{z}}_{i,1} + \bar{\mathbf{z}}_{i,2})$;
  $\bar{\mathbf{w}}_{i,2} = -c\bar{\mathbf{t}}_{i,2} + \bar{p} \cdot (\bar{\mathbf{A}}\bar{\mathbf{z}}_{i,1} + \bar{\mathbf{z}}_{i,3}) + \mathbf{z}_i$;
  $\mathbf{com}_i = (\bar{\mathbf{t}}_{i,1}, \bar{\mathbf{t}}_{i,2}, \bar{\mathbf{w}}_{i,1}, \bar{\mathbf{w}}_{i,2})$;
  $u \leftarrow [0,1)$;
  if $u > 1/\bar{M}_i$ **then**
    $\mathbf{op}_i \leftarrow \perp$;
  else
    $\mathbf{op}_i \leftarrow [\bar{\mathbf{z}}_{i,1} \| \bar{\mathbf{z}}_{i,2} \| \bar{\mathbf{z}}_{i,3}]$;
  return $(\mathbf{com}_i, \mathbf{op}_i)$;

Game 4. $\mathcal{S}_1$ riformula le variabili utilizzando direttamente:

$$\mathbf{z}_i = \mathbf{y}_i + c\mathbf{s}_i \quad \text{per } i \in \{1,2\},$$

e

$$\mathbf{w} = \mathbf{A}_1\mathbf{z}_1 + \mathbf{A}_2\mathbf{z}_2 - c\mathbf{t}_A.$$

Osserviamo che, utilizzando la relazione $\mathbf{t}_A = \mathbf{A}_1\mathbf{s}_1 + \mathbf{A}_2\mathbf{s}_2$, vale:

$$\begin{aligned}\mathbf{A}_1\mathbf{z}_1 + \mathbf{A}_2\mathbf{z}_2 &= \mathbf{A}_1(\mathbf{y}_1 + c\mathbf{s}_1) + \mathbf{A}_2(\mathbf{y}_2 + c\mathbf{s}_2) \\ &= \mathbf{A}_1\mathbf{y}_1 + \mathbf{A}_2\mathbf{y}_2 + c(\mathbf{A}_1\mathbf{s}_1 + \mathbf{A}_2\mathbf{s}_2) \\ &= \mathbf{A}_1\mathbf{y}_1 + \mathbf{A}_2\mathbf{y}_2 + c\mathbf{t}_A.\end{aligned}$$

Da cui segue che

$$\mathbf{A}_1\mathbf{y}_1 + \mathbf{A}_2\mathbf{y}_2 = \mathbf{A}_1\mathbf{z}_1 + \mathbf{A}_2\mathbf{z}_2 - c\mathbf{t}_A,$$

che coincide con la definizione di $\mathbf{w}$ nel Game 2.

Pertanto, la trasformazione tra $(\mathbf{y}_1, \mathbf{y}_2)$ e $(\mathbf{z}_1, \mathbf{z}_2)$ è una semplice traslazione deterministica, invertibile una volta fissato c e il witness $(\mathbf{s}_1, \mathbf{s}_2)$. Ne consegue che la distribuzione delle variabili osservabili dall'avversario rimane invariata. Pertanto si ha che

$$\varepsilon_4 = \varepsilon_3.$$

Game 5. $\mathcal{S}_1$ campiona direttamente

$$\mathbf{z}_2 \leftarrow D^{m_2\hat{d}}_{\mathbf{0},\sigma_2}$$

e accetta con probabilità $1/M_2$.

Dal teorema 2.2.1, segue che la distribuzione risultante è indistinguibile da quella originale:

$$|\varepsilon_5 - \varepsilon_4| = \text{negl}(\lambda).$$

Game 6. $\mathcal{S}_1$ sostituisce i commitment con valori della forma:

$$\begin{bmatrix} \mathbf{t}_A \\ \mathbf{t}_y \\ \mathbf{t}_g \\ t \end{bmatrix} = \boldsymbol{\nu} + \begin{bmatrix} \mathbf{A}\mathbf{s}_1 \\ \mathbf{y}_3 \\ \mathbf{g} \\ f_1 \end{bmatrix},$$

dove $\boldsymbol{\nu}$ è uniforme.

Se esistesse un algoritmo che possa distinguere le differenze tra questo gioco e il precedente, si potrebbe costruire un avversario che viola l'assunzione MLWE. Pertanto:

$$|\varepsilon_6 - \varepsilon_5| = \text{negl}(\lambda).$$

Game 7. $\mathcal{S}_1$ campiona direttamente:

$$(\mathbf{t}_A, \mathbf{t}_y, \mathbf{t}_g, t) \leftarrow \hat{\mathcal{R}}_{\hat{q}}^{n+256/\hat{d}+\tau+1}.$$

Questa modifica non altera la distribuzione perché nel gioco precedente questi valori erano già uniformemente distribuiti su $\hat{\mathcal{R}}_{\hat{q}}^{n+256/\hat{d}+\tau+1}$ grazie alla casualità delle variabili da cui dipendevano (come, ad esempio $\mathbf{s}_i$ e $\mathbf{y}_3$). Possiamo quindi concludere che:

$$\varepsilon_7 = \varepsilon_6.$$

Game 8. $\mathcal{S}_1$ campiona direttamente

$$\mathbf{z}_1 \leftarrow D^{m_1\hat{d}}_{\mathbf{0},\sigma_1}$$

e applica rejection sampling. Si ha per la seconda conseguenza del lemma A.1.1:

$$|\varepsilon_8 - \varepsilon_7| \leq N2^{-\lambda} = \text{negl}(\lambda).$$

Game 9. $\mathcal{S}_1$ campiona direttamente

$$h_i \leftarrow \{x \in \hat{\mathcal{R}}_{\hat{q}} : \tilde{x} = 0\}.$$

Poiché $\mathbf{g}$ era già uniforme nello stesso insieme:

$$\varepsilon_9 = \varepsilon_8.$$

Game 10. $\mathcal{S}_1$ campiona direttamente

$$\vec{z}_3 \leftarrow D^{256}_{\vec{0},\sigma_3}$$

e applica rejection sampling. Di nuovo, per il lemma A.1.1 si ottiene:

$$|\varepsilon_{10} - \varepsilon_9| = \text{negl}(\lambda).$$

Conclusione. Nel gioco finale, il simulatore $\mathcal{S}_1$ genera una prova senza utilizzare il witness w.

Combinando tutti i passaggi, si ottiene:

$$|\varepsilon_{10} - \varepsilon| = \text{negl}(\lambda),$$

da cui segue che le distribuzioni dei transcript reali e simulati sono indistinguibili.

Pertanto, il protocollo $\Pi^{\text{Com}}_{\text{NIZK}}$ è zero-knowledge. □

A.2.3 Multi-Proof Extractability

Teorema A.2.3. *Sia*

$$B := 8\nu\sqrt{B_1^2 + B_2^2}$$

e sia

$$q > \max\left\{B,\ 16m_1\hat{d}B_3,\ \frac{2}{\omega_{\min}(\lambda)^2}B_3^2\right\}.$$

Allora, la prova $\Pi^{\text{NIZK}}_{\text{Com}}$ *per la relazione* R_{Com} *è* multi-proof extractable *sotto le assunzioni*

$$\text{MLWE}_{m_1,m_1+n_1,\eta_1},\quad \text{MLWE}_{m_2,m_2+n_2,\eta_2},\quad \text{MSIS}_{n,m_1+m_2,B}.$$

Dimostrazione. Dimostriamo la proprietà di multi-proof extractability seguendo l'approccio dell'articolo Bootle et al. [2023].

CRS Simulatability. Definiamo il simulatore $\mathcal{S}_{\mathsf{crs}}$ che campiona:

$$\widetilde{\mathsf{crs}} := (\mathbf{A}_1, \mathbf{A}_2, \mathbf{B}_y, \mathbf{B}_g, \mathbf{b}) \leftarrow \hat{\mathcal{R}}_{\hat{q}}^{n\times m_1} \times \hat{\mathcal{R}}_{\hat{q}}^{n\times m_2} \times \hat{\mathcal{R}}_{\hat{q}}^{256/\hat{d}\times m_2} \times \hat{\mathcal{R}}_{\hat{q}}^{\tau\times m_2} \times \hat{\mathcal{R}}_{\hat{q}}^{m_2}.$$

Inoltre campiona:

$$(\bar{\mathbf{A}}_1, \bar{\mathbf{A}}_2) \leftarrow \hat{\mathcal{R}}_{\hat{q}}^{m_1\times n_1} \times \hat{\mathcal{R}}_{\hat{q}}^{m_2\times n_2}.$$

Per $i \in \{1,2\}$, campiona:

$$\bar{\mathbf{D}}_{i,1} \leftarrow S_{\eta_i}^{m_i\times m_i},\quad \bar{\mathbf{D}}_{i,2} \leftarrow S_{\eta_i}^{m_i\times n_i},$$

e definisce:

$$\bar{\mathbf{B}}_i := \bar{\mathbf{D}}_{i,1}\bar{\mathbf{A}}_i + \bar{\mathbf{D}}_{i,2}.$$

Restituisce quindi:

$$\widetilde{\mathsf{crs}} = (\mathbf{A}_1, \mathbf{A}_2, \mathbf{B}_y, \mathbf{B}_g, \mathbf{b}, \bar{\mathbf{A}}_1, \bar{\mathbf{A}}_2, \bar{\mathbf{B}}_1, \bar{\mathbf{B}}_2)$$

e la trapdoor:

$$\mathsf{td} = (\bar{\mathbf{D}}_{1,1}, \bar{\mathbf{D}}_{1,2}, \bar{\mathbf{D}}_{2,1}, \bar{\mathbf{D}}_{2,2}).$$

Per le assunzioni $\mathsf{MLWE}_{m_1,n_1,\eta_1}$ e $\mathsf{MLWE}_{m_2,n_2,\eta_2}$, $\widetilde{\mathsf{crs}}$ è indistinguibile da un crs scelto casualmente.

Straight-line extractability. Sia $\mathcal{A}$ un avversario PPT che effettua al più Q_H query all'oracolo random e restituisce Q_s coppie $(x_i, \pi_i)_{i\in[Q_s]}$ tali che:

$$\mathbb{P}\left[\forall i,\ \mathsf{Verify}^{H}_{\mathsf{Com}}(\widetilde{\mathsf{crs}}, x_i, \pi_i) = 1\right] \geq \varepsilon(\lambda).$$

Consideriamo il transcript completo associato a π:

$$\mathsf{tr} = ((\mathbf{t}_A, \mathbf{t}_y, \mathbf{t}_g, \mathsf{com}_1, \mathsf{com}_2), (\mathfrak{R}_0, \mathfrak{R}_1), \vec{z}_3, (\gamma_{i,j}), \mathbf{h}, (\mu_i), (t, f_0), c, (\mathbf{z}_1, \mathbf{z}_2, \mathsf{op}_1, \mathsf{op}_2)),$$

dove le challenge sono determinate dall'oracolo H.

Separiamo il transcript come:

$$\mathsf{tr} = (\mathsf{pref_tr}, (\mathbf{z}_1, \mathbf{z}_2, \mathbf{op}_1, \mathbf{op}_2)),$$

dove $\mathsf{pref_tr}$ contiene tutti i messaggi eccetto l'ultimo.

Definiamo quindi l'insieme:

$$\Gamma(x, \mathsf{tr}) := \{c' \in \mathcal{C} \ :\ \exists(\mathbf{z}'_1, \mathbf{z}'_2, \mathbf{op}'_1, \mathbf{op}'_2) \text{ tali che } (\mathsf{pref_tr}, (\mathbf{z}'_1, \mathbf{z}'_2, \mathbf{op}'_1, \mathbf{op}'_2)) \text{ è accettato}\}.$$

Osserviamo che, se esistono due transcript validi con stessa $\mathsf{pref_tr}$ ma challenge diverse $c \neq c'$, allora per la correttezza e assenza di errore di decrittazione di LHC si ha:

$$\mathbf{z}_1 = \mathbf{z}'_1, \quad \mathbf{z}_2 = \mathbf{z}'_2.$$

Algoritmo di estrazione. L'algoritmo $\mathsf{MultiExtract}$, dato (x, π), procede come segue:

- ricostruisce tr;
- inizializza $\mathsf{BadChall} = \{c\}$;
- ripete:
  - campiona $c' \leftarrow \mathcal{C} \setminus \mathsf{BadChall}$;
  - usa la trapdoor τ per decriptare i commitment LHC e ottenere $\mathbf{z}'_1, \mathbf{z}'_2$;
  - verifica le equazioni del verificatore;
  - se falliscono, aggiunge c' a $\mathsf{BadChall}$.

Definiamo $\mathsf{GoodChall}(x, \mathsf{tr})$ come l'insieme dei c' per cui l'algoritmo ha successo. Vale:

$$\Gamma(x, \mathsf{tr}) \subseteq \mathsf{GoodChall}(x, \mathsf{tr}).$$

Lower bound su $\Gamma(x, \mathsf{tr})$. Per il lemma 6.4 di Bootle et al. [2023], si ha che con probabilità almeno $\varepsilon/2 - \mathrm{negl}(\lambda)$:

$$|\Gamma(x, \mathsf{tr})| \geq \frac{\varepsilon}{2Q_H}|\mathcal{C}|.$$

Algoritmo 22: MultiExtract

Input: Input: $Q_H, Q_s, 1/\varepsilon, \widetilde{\text{crs}} = (\mathbf{A}_1, \mathbf{A}_2, \mathbf{B}_y, \mathbf{B}_g, \mathbf{b}, \bar{\mathbf{A}}_1, \bar{\mathbf{A}}_2, \bar{\mathbf{B}}_1, \bar{\mathbf{B}}_2)$
Trapdoor: $\mathsf{td} = (\bar{\mathbf{D}}_{1,1}, \bar{\mathbf{D}}_{1,2}, \bar{\mathbf{D}}_{2,1}, \bar{\mathbf{D}}_{2,2})$
$x = (P, \vec{u}, B)$, prova π
Output: Witness w, oppure $(\perp, (x, \pi, \text{aux}))$, oppure $(\perp, \perp)$
begin
  $\pi = (\mathbf{t}_A, \mathbf{t}_y, \mathbf{t}_g, \mathbf{w}, \mathsf{com}_1, \mathsf{com}_2, \vec{z}_3, \mathbf{h}, t, f_0, \mathbf{z}_1, \mathbf{z}_2, \mathsf{op}_1, \mathsf{op}_2)$;
  $\mathbf{B} = \begin{bmatrix} \mathbf{B}_y \\ \mathbf{B}_g \end{bmatrix}, \quad \mathbf{t}_B = \begin{bmatrix} \mathbf{t}_y \\ \mathbf{t}_g \end{bmatrix}$;
  calcola $\mathbf{D}_2, \mathbf{d}_1, d_0$;
  if *una delle condizioni di verifica fallisce* **then**
    return $(\perp, \perp)$;
  $\mathsf{com}_i = (\bar{\mathbf{t}}_{i,1}, \bar{\mathbf{t}}_{i,2}, \bar{\mathbf{w}}_{i,1}, \bar{\mathbf{w}}_{i,2})$ per $i \in \{1,2\}$;
  $\mathsf{BadChall} \leftarrow \{c\}$;
  $t \leftarrow 0$;
  while $t \leq T_{\max} := \lambda \cdot \frac{2Q_H}{\varepsilon}$ **do**
    campiona $c' \leftarrow \mathcal{C} \setminus \mathsf{BadChall}$;
    for $i \in \{1,2\}$ **do**
      $\mathbf{z}'_i = (\bar{\mathbf{w}}_{i,2} + c'\bar{\mathbf{t}}_{i,2}) - \bar{\mathbf{D}}_{i,1}(\bar{\mathbf{w}}_{i,1} + c'\bar{\mathbf{t}}_{i,1}) \bmod \bar{p}$;
    $\mathbf{z}' = \begin{bmatrix} \mathbf{z}'_1 \\ \sigma(\mathbf{z}'_1) \\ c'\mathbf{t}_B - \mathbf{B}\mathbf{z}'_2 \\ \sigma(c'\mathbf{t}_B - \mathbf{B}\mathbf{z}'_2) \end{bmatrix}$;
    $\delta := c't - \mathbf{b}^T\mathbf{z}'_2$;
    $f'_0 = \mathbf{z}'^T\mathbf{D}_2\mathbf{z}' + c'\mathbf{d}_1^T\mathbf{z}' + c'^2 d_0 - \delta$;
    $b \leftarrow 1$;
    if $\|\mathbf{z}'_1\| > B_1$ ***or*** $\|\mathbf{z}'_2\| > B_2$ ***or*** $\mathbf{A}_1\mathbf{z}'_1 + \mathbf{A}_2\mathbf{z}'_2 \neq \mathbf{w} + c'\mathbf{t}_A$ ***or*** $f'_0 \neq f_0$ **then**
      $b \leftarrow 0$;
    if $b = 1$ **then**
      $c^* = c' - c$;
      for $i \in \{1,2\}$ **do**
        $\mathbf{s}^*_i = \frac{\mathbf{z}'_i - \mathbf{z}_i}{c' - c}$;
      $w = \text{Coeffs}(\mathbf{s}^*_1)$;
      if $(x, w) \in R^{\text{Com}}$ **then**
        return w;
      else
        $\text{aux} = (\mathbf{s}^*_1, \mathbf{s}^*_2, c^*)$;
        return $(\perp, (x, \pi, \text{aux}))$;
    $\mathsf{BadChall} \leftarrow \mathsf{BadChall} \cup \{c'\}$;
    $t \leftarrow t + 1$;
  return $(\perp, \perp)$;

Probabilità di fallimento. Ad ogni iterazione, la probabilità di scegliere $c' \in \mathsf{GoodChall}$ è almeno $\varepsilon/(2Q_H)$. Ripetendo per

$$T_{\max} = \lambda \cdot \frac{2Q_H}{\varepsilon}$$

iterazioni, la probabilità di fallire è trascurabile. Applicando la disuguaglianza di Boole su $Q_s = \mathsf{poly}(\lambda)$ istanze:

$$\varepsilon_{\text{fail}} \leq \frac{\varepsilon}{2} + \text{negl}(\lambda).$$

Binding ed estrazione. Grazie alla proprietà di binding estraibile del commitment LHC e alla difficoltà del problema $\mathsf{MSIS}_{n,m_1+m_2,B}$, si dimostra che:

$$\varepsilon_{\text{bind}} = \text{negl}(\lambda).$$

Conclusione. Combinando i risultati:

$$\mathbb{P}\left[\begin{array}{l}(\widetilde{\mathsf{crs}}, \mathsf{td}) \leftarrow \mathsf{S}_{\mathsf{crs}}(1^\lambda),\\ \{(x_i, \pi_i)\}_{i\in[Q_s]} \leftarrow \mathcal{A}^H(\widetilde{\mathsf{crs}}),\\ \{w_i \leftarrow \mathsf{MultiExtract}(Q_H, Q_s, 1/\varepsilon, \widetilde{\mathsf{crs}}, \tau, x_i, \pi_i)\}_{i\in[Q_s]} :\\ \forall i \in [Q_s],\ (x_i, w_i) \in R^{Com}\end{array}\right] \geq \varepsilon - \varepsilon_{\text{fail}} - \varepsilon_{\text{bind}} \geq \frac{\varepsilon}{2} - \text{negl}(\lambda).$$

Questo conclude la dimostrazione della multi-proof extractability. □

Bibliografia

Miklós Ajtai. Generating hard instances of lattice problems. In *Proceedings of the 28th Annual ACM Symposium on Theory of Computing (STOC)*, pages 99–108, 1996.

Fabrice Benhamouda, Özgür Dagdelen, Thibaut Lepoint, Vadim Lyubashevsky, and Elisabeth Oswald. On the hardness of lattice problems with polynomial modulus. In *Progress in Cryptology – AFRICACRYPT 2013*, volume 7918 of *Lecture Notes in Computer Science*, pages 1–18, 2013. doi: 10.1007/978-3-642-38553-7_1.

Jonathan Bootle, Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Alessandro Sorniotti. A framework for practical anonymous credentials from lattices. *Lecture Notes in Computer Science*, 14082:384–417, 2023. doi: 10.1007/978-3-031-38545-2_13.

Muhammed F. Esgin, Aggelos Kiayias, and Alexander Russell. Matrict: Efficient, scalable and post-quantum blockchain confidential transactions protocol. In *Proceedings of the 2019 ACM CCS*, pages 567–584. ACM, 2019. doi: 10.1145/3319535.3354215.

Shuichi Katsumata. A new simple technique to bootstrap various lattice zero-knowledge proofs to qrom secure nizks. In *Advances in Cryptology – CRYPTO 2021*, volume 12827 of *Lecture Notes in Computer Science*, pages 580–610, 2021. doi: 10.1007/978-3-030-84245-1_20.

Vadim Lyubashevsky. Lattice signatures without trapdoors. In *Proceedings of EUROCRYPT 2012*, volume 7237, pages 738–755. Springer, Lecture Notes in Computer Science, 2012. doi: 10.1007/978-3-642-29011-4_43.

Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Maxime Plançon. Lattice-based zero-knowledge proofs and applications: Shorter, simpler, and more general. In *Advances in Cryptology – CRYPTO 2022*, volume 13508 of *Lecture Notes in Computer Science*, pages 71–101, 2022. doi: 10.1007/978-3-031-15979-4_3.

Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on gaussian measures. *SIAM Journal on Computing*, 37(1):267–302, 2007. doi: 10.1137/S0097539705447360.

Roberto Previtali. Lyubashevsky's lattice-based digital signature schemes: analysis of the parameters and the rejection sampling technique. Master's thesis, Politecnico di Torino, 2025.