TRABAJO DE FIN DE GRADO

Grado en Matemáticas

Título:
PROTOCOLOS PARA VOTACIONES ELECTRÓNICAS

Autor: MARTÍNEZ PINILLA, RAMIRO

Director: MORILLO BOSCH, MARIA PAZ

Departamento: MATEMÁTICAS

Convocatoria: 2016/2017

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat de Matemàtiques i Estadística

1. Sistemas de votación electrónica

Los sistemas de votación electrónica presentan una serie de ventajas respecto a los sistemas tradicionales. Automatizan el recuento, reducen el gasto y facilitan la votación desde el extranjero. Sin embargo es necesario garantizar que el voto no ha sido manipulado, que únicamente han votado aquellos inscritos en el censo, que nadie ha emitido más de un voto, y que nadie puede conocer el sentido del voto de ningún votante. Pueden requerirse también otras cuestiones adicionales, como el hecho de que el votante no pueda probar a otra persona el sentido de su voto, gracias a lo cual se reducirían las coacciones o la compra de votos, pues el comprador no tendría garantías del sentido final del voto. En las votaciones tradicionales esto se consigue haciendo que todo el proceso sea visible, el votante solo puede votar en la mesa que le corresponde y se identifica ante ella, el voto se deposita en un sobre cerrado que no guarda ninguna relación con el votante y el recuento se hace de forma pública.

Sin embargo en un proceso de votación electrónica todas estas operaciones se realizan de forma informática, y es necesario convencernos de que se cumplen cada uno de los requisitos. En particular parece existir un aparente conflicto entre la privacidad (el voto ha de ser secreto) y la verificabilidad (el voto ha de ser emitido por una persona que ha de identificarse como miembro del censo, y ha de garantizarse que el contenido del voto no se ha modificado hasta el momento del recuento).

El primer paso para garantizar el secreto del voto es emitirlo cifrado utilizando un sistema de clave pública, de forma que únicamente el servidor encargado del recuento final sea capaz de descifrarlo y ver su contenido. Para garantizar que el voto ha sido emitido por alguien con derecho a voto el votante deberá firmar el voto cifrado, y un servidor comprobará que esa firma ha sido realizada por alguien que pertenece al censo y únicamente ha emitido ese voto. El problema en este momento viene por el hecho de que mientras que los sobres tradicionales son indistinguibles y por tanto es imposible asignar cada uno de ellos al votante que los depositó, los votos cifrados son únicos, y aunque solo el servidor final que posee la clave secreta puede descifrarlos alguien podría simplemente identificar el voto que ha emitido un votante concreto y esperar a que ese voto sea descifrado.

Para garantizar el anonimato de los votantes se han propuesto dos alternativas. La primera de ellas consiste en utilizar un cifrado homomórfico (la suma de los votos cifrados es un cifrado de la suma de votos) y descifrar únicamente el resultado global.

La opción más utilizada consiste en la utilización de nodos *Mixnet*, que reciben una lista de votos y producen como salida esa misma lista con los votos realeatorizados y permutados. Una vez realeatorizado el voto, es computacionalmente impracticable conocer cuál era el voto original, y como la permutación solo la conoce el propio nodo Mixnet basta con implementar una serie de estos nodos de forma consecutiva. De esta forma, si al menos uno de los nodos es honesto, será imposible identificar los votos de la salida con los de la entrada, y por tanto no habrá forma de asociar los votos que se descifran con los votantes que los emitieron.

Esto genera un problema, pues por una parte necesitamos que no se pueda relacionar individualmente ningún voto de la salida con su correspondiente voto de la entrada, pero además necesitamos garantías de que estos votos de la salida sean exactamente los mismos votos de la entrada (realeatorizados y permutados) sin ninguna otra modificación, manipulación o sustitución de votos. Necesitamos en definitiva que cada nodo mixnet nos demuestre que únicamente ha realeatorizado y permutado los votos sin desvelarnos ninguna información sobre la permutación y los elementos aleatorios empleados para ello. Esto tiene una definición matemática precisa en lo que se llaman pruebas de conocimiento nulo.

Junto con la verificabilidad, la seguridad del cifrado juega un papel importante, pues siendo el sentido

del voto una información tan sensible es necesario garantizar su seguridad a largo plazo. Dado que los votos cifrados son públicos, para probar que no se han manipulado hasta el momento del recuento, un adversario podría almacenarlos y esperar a que llegado el momento pueda descifrarlos si se rompe la seguridad.

Los esquemas de cifrado más utilizados en la actualidad pueden ser rotos empleando computadores cuánticos, que aunque todavía no son una realidad sí pueden serlo en las siguientes décadas. Notamos aquí que en ocasiones esto no supone un problema. Si un servidor quiere demostrar que ha procesado correctamente los votos o un votante quiere identificarse podemos pedirle una información que solo pueda proporcionar si realmente el servidor ha sido honesto, o si efectivamente el votante es quien dice ser, y que únicamente podrían ser falseadas si se supiera calcular un logaritmo discreto. Esto significa que este tipo de pruebas no se podrán seguir utilizando tras la aparición de los ordenadores cuánticos, pues en ese momento dudaríamos de que el servidor o el votante pudieran haber falseado la información, pero sabemos que en las elecciones anteriores sí era un método válido, pues cuando proporcionaron la información no eran capaces de falsearla.

Es únicamente cuando las herramientas criptográficas se utilizan para esconder información cuando nos importa lo que suceda a largo plazo, pues la revelación de datos tan sensibles como el sentido de voto en el pasado sí que acarrearía consecuencias.

Por ello si queremos garantizar la privacidad futura de las votaciones que se realicen hoy en día es necesario desarrollar sistemas de votación electrónica seguros a largo plazo y empezar a utilizarlos desde este momento para no comprometer la privacidad de los votantes actuales. Una de las opciones pasaría por utilizar criptografía cuántica para transmitir los votos, sin embargo es mucho más sencillo desarrollar sistemas post-cuánticos, es decir, que utilicen algoritmos convencionales cuya seguridad está garantizada tanto frente a la computación clásica como a la cuántica.

Este trabajo consiste en la propuesta de una prueba de conocimiento nulo para verificar el buen funcionamiento de un nodo mixnet que funcione con un cifrado post-cuántico, concretamente basado en ring-LWE.

2. Cifrado

Para proteger la privacidad de los votantes y ocultar el sentido del voto estos se envían cifrados. Un cifrado consiste en transformar un mensaje de forma que resulte ininteligible y nadie salvo el receptor pueda conocer su contenido. El descifrado es el método por el que el receptor es capaz de recuperar el mensaje original a partir del cifrado. Esta transformación y su correspondiente inversa se realizan utilizando una información secreta conocida como clave.

En nuestro caso los emisores son cada uno de los votantes y el receptor final sería la mesa electoral que hace el recuento (en la práctica será un servidor o un conjunto de servidores).

En concreto para las votaciones electrónicas se emplea un esquema de cifrado asimétrico, en el que se utilizan dos claves distintas. La clave de cifrado se hace pública (y será conocida por todos los votantes), mientras que la clave para descifrar se mantiene privada y únicamente será conocida por la mesa electoral.

Definición 2.1. Un *esquema de cifrado asimétrico o de clave pública* consta de tres algoritmos:

- **Gen**: recibe como entrada un parámetro de seguridad 1^λ y produce como salida un par de claves (pk, sk), que llamaremos clave pública y clave privada respectivamente. El parámetro de seguridad definirá la longitud de las claves. También define el espacio de los mensajes $\mathcal{M}_{sp}$, el espacio de los textos cifrados $\mathcal{C}_{sp}$ y el espacio de elementos aleatorios $\mathcal{R}_{sp}$ (en un esquema de cifrado probabilístico).
- **Cif**: recibe como entrada la clave pública pk, un mensaje $M \in \mathcal{M}_{sp}$ y opcionalmente un elemento aleatorio $r \in \mathcal{R}_{sp}$, con los que produce como salida un texto cifrado $C = E_{pk}(M, r) \in \mathcal{C}_{sp}$.
- **Descif**: recibe como entrada la clave privada sk y un texto cifrado C, a partir de los cuales recupera el mensaje original $M = D_{sk}(C)$.

Definición 2.2. Decimos que un esquema de cifrado de clave pública es *homomórfico* respecto a dos operaciones $*, \oplus$ si $E_{pk}(a) * E_{pk}(b) = E_{pk}(a \oplus b)$.

Cuando analizamos la seguridad del cifrado hablamos de *seguridad computacional*. Es decir, un adversario con capacidad de computación ilimitada podría llegar a romper la seguridad del cifrado y recuperar el mensaje M a partir del texto cifrado C y la clave pública pk sin conocer la clave secreta sk. A pesar de ello buscamos esquemas de cifrados para los que un adversario que emplee únicamente algoritmos eficientes (cuyo consumo de recursos, en espacio de almacenamiento y número de operaciones, sea polinómico respecto a un parámetro de seguridad) tenga una probabilidad negligible de obtener la respuesta correcta.

Definición 2.3. Una función f es *negligible* en el parámetro de seguridad n si cuando este parámetro crece la función decrece de forma más rápida que cualquier polinomio en n:

$$\forall c > 0 \quad \exists n_0 \in \mathbb{N} \quad | \quad \forall n \geq n_0 \quad f(n) \leq \frac{1}{n^c}$$

Hemos de tener en cuenta que sería deseable que el texto cifrado no proporcionara ninguna información sobre el texto original. Por tanto además de ataques de obtención del mensaje hemos de protegernos contra ataques que intenten distinguir entre cifrados. Es decir, dados dos mensajes y sus correspondientes cifrados queremos que la probabilidad con la que un atacante sea capaz de distinguir qué cifrado corresponde a cada mensaje sea similar al $1/2$ que se obtendría asignando los mensajes a los cifrados al azar. En concreto pedimos que la diferencia con este $1/2$ sea una función negligible.

$$\Pr\left[b = b' \;\middle|\; \begin{array}{c} c_0 \xleftarrow{\$} E_{pk}(m_0), \quad c_1 \xleftarrow{\$} E_{pk}(m_1) \\ b \xleftarrow{\$} \{0,1\}, \quad b' \xleftarrow{\$} A(m_0, m_1, c_b, c_{\bar{b}}, pk) \end{array}\right] \leq \frac{1}{2} + f(x)$$

Esta indistinguibilidad de cifrados es importante en nuestro caso, pues habitualmente el contenido de los votos no es un texto cualquiera sino una de las opciones de una pequeña lista, por lo que únicamente habría que comprobar a cual de estas opciones pertenece el cifrado.

Al disponer el adversario de los mensajes y de la clave pública podría cifrarlos, obteniendo los mismos textos cifrados realeatorizados. Por ello uno de los requisitos del esquema de cifrado es que ha de ser probabilístico para que los cifrados que obtenga sean diferentes de los dados a pesar de compartir textos planos. Esta noción de seguridad implica además que al realeatorizar los textos cifrados resulta imposible para un adversario eficiente identificar a qué cifrado original correspondería cada uno de los cifrados realeatorizados, propiedad que será necesaria en la construcción del nodo mixnet.

Dependiendo de la seguridad que se quiera conseguir podremos asumir o no que el atacante tiene acceso a una serie de muestras de textos planos y sus correspondientes cifrados, siendo estos textos aleatorios o elegidos por el propio atacante, lo que lleva a diferentes definiciones de seguridad.

En última instancia la seguridad se prueba reduciendo un problema ampliamente estudiado que sea computacionalmente difícil al problema de obtención del mensaje o bien al de distinción entre cifrados. Es decir, se demuestra que estos problemas son al menos tan difíciles como otro problema conocido para el que no se conozca ningún algoritmo eficiente. Alguien capaz de romper la seguridad del esquema de cifrado también sería capaz de utilizar ese método para resolver de forma eficiente este problema.

Los problemas computacionales más utilizados son el problema RSA (consistente en calcular raíces e-ésimas módulo n) y el problema de calcular logaritmos discretos, en los que se basan los esquemas de cifrado RSA y ElGamal respectivamente. Para estos algoritmos no se conoce ningún método eficiente en la actualidad. El esquema de cifrado RSA podría ver comprometida su seguridad si se conociera un algoritmo eficiente para factorizar n, pues esto llevaría a calcular trivialmente la clave secreta que permitiría directamente descifrar los mensajes. El problema de la factorización de enteros es también un problema considerado computacionalmente difícil.

Sin embargo tanto el problema de factorización como el del logaritmo discreto pueden ser resueltos en tiempo polinómico utilizando un ordenador cuántico, gracias a los algoritmos desarrollados por Shor [1].

Dado que los votos cifrados suelen hacerse públicos, para que cualquiera pueda verificar que la votación se ha realizado correctamente, esto significa que alguien podría almacenar esos votos y esperar hasta que la tecnología haya avanzado y existan ordenadores cuánticos suficientemente potentes como para poder descifrar todos los votos y conocer las opciones que eligieron cada uno de los votantes.

Por ello es necesario utilizar esquemas de cifrado que basen su seguridad en problemas que sean difíciles tanto en un modelo de computación clásica como cuántica. Una de las alternativas más prometedoras y que más espacio ocupa en la literatura reciente es la criptografía basada en retículos.

2.1 Retículos y cifrado ring-LWE

Definición 2.4. Un *retículo* $\mathcal{L}$ de dimensión n es un subconjunto de puntos de $\mathbb{R}^n$ que cumple las siguientes características:

- Es un subgrupo aditivo: $0 \in \mathcal{L}$ y $\forall x, y \in \mathcal{L} \quad -x, x+y \in \mathcal{L}$
- Es discreto: $\forall x \in \mathcal{L}$ existe un entorno de x en $\mathbb{R}^n$ en el que x es el único punto del retículo.

Dados k vectores linealmente independientes $\mathbf{b_1}, \dots, \mathbf{b_k} \in \mathbb{R}^n$ el retículo generado por ellos es el siguiente conjunto:

$$\mathcal{L}(\mathbf{b_1}, \dots, \mathbf{b_k}) = \left\{ \sum_{i=1}^{k} z_i \mathbf{b_i} \;\middle|\; z_i \in \mathbb{Z} \right\} = \{ B\mathbf{z} \mid z \in \mathbb{Z}^n \} = \mathcal{L}(B)$$

Donde hemos llamado B a la matriz cuyas columnas son los vectores $\mathbf{b_i}$. Decimos que $\mathbf{b_1}, \dots, \mathbf{b_k}$ forman una base del retículo $\mathcal{L}$. Cualquier retículo admite infinitas bases y $\mathcal{L}(B) = \mathcal{L}(B')$ si y solo si existe una matriz U unimodular (matriz cuadrada de enteros con determinante ± 1) tal que $B' = BU$.

Definición 2.5. Un *retículo q-ario* es un retículo $\mathcal{L}$ que satisface $q\mathbb{Z}^n \subseteq \mathcal{L} \subseteq \mathbb{Z}^n$ para un entero q.

En un retículo q-ario la pertenencia de un vector x a $\mathcal{L}$ viene determinada por $x \mod q$.

Definición 2.6. Definimos el *mínimo* $\lambda_i(\mathcal{L})$ como el radio de la menor hiperesfera centrada en el origen que contiene i puntos del retículo linealmente independientes.

Una vez definidos estos conceptos podemos plantear una serie de problemas sobre los retículos, que pueden servir para construir esquemas de cifrado.

Definición 2.7. El *problema del vector más corto aproximado* o γ-SVP consiste en, dada una base B de un retículo encontrar un vector distinto de cero $\mathbf{v} \in \mathcal{L}(B)$ tal que $\|\mathbf{v}\| \leq \gamma \cdot \lambda_1(\mathcal{L})$

Definición 2.8. El *problema del vector más cercano aproximado* o γ-CVP consiste en, dada una base B de un retículo y un vector objetivo $\mathbf{t}$ encontrar un vector $\mathbf{u} \in \mathcal{L}(B)$ tal que si $\mathbf{v} = \arg\min_{v \in \mathcal{L}} \|\mathbf{t} - \mathbf{v}\|$ entonces $\|\mathbf{u} - \mathbf{v}\| \leq \gamma \|\mathbf{t} - \mathbf{v}\|$.

Definición 2.9. Sean n, q enteros positivos (q habitualmente primo), χ una distribución de probabilidad discreta en $\mathbb{Z}$ (habitualmente la distribución Gaussiana discreta) y $\mathbf{s}$ un vector secreto de $\mathbb{Z}_q^n$.

Llamamos $\mathcal{L}_{\mathbf{s},\chi}$ a la distribución de probabilidad en $\mathbb{Z}_q^n \times \mathbb{Z}_q$ obtenida eligiendo $\mathbf{a} \in \mathbb{Z}_q^n$ uniformemente al azar, eligiendo $e \in \mathbb{Z}$ de acuerdo con χ y considerándolo en $\mathbb{Z}_q$ y calculando finalmente $(\mathbf{a}, c = \langle \mathbf{a}, \mathbf{s} \rangle + e) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$.

LWE-Decisional es el problema de decidir si los pares $(\mathbf{a}, c) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ son muestras de $\mathcal{L}_{\mathbf{s},\chi}$ o de la distribución uniforme en $\mathbb{Z}_q^n \times \mathbb{Z}_q$.

LWE-Búsqueda es el problema de recuperar $\mathbf{s}$ a partir de $(a, c = \langle \mathbf{a}, \mathbf{s} \rangle + e) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ con las muestras elegidas siguiendo $\mathcal{L}_{\mathbf{s},\chi}$.

En los problemas LWE el número de muestras m es polinómico en n. Escribiéndolo de forma matricial tenemos $(A, A\mathbf{s} + \mathbf{e})$, por lo que se puede interpretar como un punto de un retículo generado por A al que se le ha añadido un pequeño error $\mathbf{e}$. Resolver estos problemas escogiendo los coeficientes de $\mathbf{s}$ de χ es tan difícil como hacerlo con una $\mathbf{s}$ elegida al azar en $\mathbb{Z}_q^n$ [2].

A su vez Regev [3] demostró que tomando como distribución de error χ una distribución gausiana discreta con desviación típica σ suficientemente grande el problema de LWE-Búsqueda es al menos tan difícil como resolver cuánticamente el problema del vector más corto aproximado γ-SVP, tomando como factor de aproximación $\gamma(n) = (n \cdot q/\sigma)$. Es decir, existe una reducción del peor caso del problema γ-SVP al problema LWE-Búsqueda. Se trata de una demostración de seguridad especialmente fuerte, pues está basada en el peor caso y no solo en la media.

Por otra parte LWE-Decisional y LWE-Búsqueda son problemas equivalentes bajo ciertas condiciones. La reducción del problema decisional a la búsqueda es trivial, pues si la búsqueda encuentra un parámetro $\mathbf{s}$ solo hemos de comprobar si $\mathbf{e}' = \mathbf{c} - A\mathbf{s}$ es pequeño para saber si es una muestra LWE o no. El hecho de que el problema decisional sea difícil nos permitirá usar las muestras LWE como elementos pseudoaleatorios.

La reducción en la otra dirección es [3]:

Teorema 2.10. *Sea $n \geq 1$ un entero, $2 \leq q \leq pol(n)$ un primo y χ una distribución sobre $\mathbb{Z}_q$. Asumimos que tenemos un procedimiento W que para toda $\mathbf{s}$ acepta con probabilidad exponencialmente cercana a 1 las entradas de $\mathcal{L}_{\mathbf{s},\chi}$ y rechaza con probabilidad exponencialmente cercana a 1 las entradas uniformemente aleatorias.*

Entonces existe un algoritmo eficiente W' que, dadas muestras de $\mathcal{L}_{\mathbf{s},\chi}$ para alguna $\mathbf{s}$ produce esa $\mathbf{s}$ con probabilidad exponencialmente cercana a 1.

Demostración. Probaremos cómo obtener de forma eficiente el primero de los coeficientes de $\mathbf{s}$, y el resto se encontrarían de forma análoga. Para cada $k \in \mathbb{Z}_q$ consideramos la siguiente transformación $(\mathbf{a}, c) \xrightarrow{\$} (\mathbf{a} + (l, 0, \ldots, 0), c + lk)$, donde $l \in \mathbb{Z}_q$ se elige uniformemente al azar. Esta transformación lleva la distribución uniforme en $\mathbb{Z}_q^n \times \mathbb{Z}_q$ a sí misma.

Sin embargo si $(\mathbf{a}, c)$ es una muestra de $\mathcal{L}_{\mathbf{s},\chi}$ tenemos que distinguir dos casos. Si $k = s_0$ entonces la transformación lleva $\mathcal{L}_{\mathbf{s},\chi}$ a sí misma. Sin embargo si $k \neq s_0$ entonces la transformación lleva $\mathcal{L}_{\mathbf{s},\chi}$ a la distribución uniforme.

Hay un número polinómico de posibilidades para s_0 (son q posibles valores), por lo que podemos comprobarlos todos. Para cada valor de k transformamos la muestra y la comprobamos con W. Como W puede distinguir $\mathcal{L}_{\mathbf{s},\chi}$ de la distribución uniforme podremos saber si $k = s_0$.

Repitiendo este proceso de forma análoga con cada s_i podemos encontrar $\mathbf{s}$ de forma eficiente. □

Para resolver el problema LWE-Búsqueda se emplea el algoritmo Babai Nearest Plane, que puede ser ejecutado en tiempo polinómico, pero sin embargo la probabilidad de encontrar una solución con este algoritmo depende de la base del retículo que se utilice. Para corregir este problema es posible encontrar otra base del mismo retículo con la que el algoritmo pueda finalmente encontrar el punto $A\mathbf{s}$, pero únicamente se conocen algoritmos que calculen esta nueva base en tiempo exponencial, como puede ser el algoritmo de ortogonalización de la base LLL [2].

La dificultad de estos problemas hace que puedan construirse herramientas criptográficas sobre ellos, sin embargo la necesidad de trabajar con matrices A muy grandes hace que en la práctica las claves y los textos cifrados ocupen demasiado espacio y las operaciones de multiplicaciones entre matrices sean demasiado costosas. Es por ello que en la literatura se ha propuesto el uso de un tipo particular de retículos, los retículos ideales.

Definición 2.11. Un *retículo ideal* es aquel que tiene por base una matriz A construida de la siguiente forma. Sea $\mathbf{a} \in \mathbb{Z}^n$ un vector y $F \in \mathbb{Z}^{n\times n}$ una matriz de transformación definida a su vez por otro vector $\mathbf{f} \in \mathbb{Z}^n$, de la siguiente forma:

$$F = \left[\begin{array}{c|c} 0^T & -f_1 \\ \hline \ddots & -f_2 \\ Id_{n-1} & \vdots \\ & \ddots \quad -f_n \end{array}\right]$$

$$A = \left[\mathbf{a}, F\mathbf{a}, \dots, F^{n-1}\mathbf{a}\right]$$

Los retículos ideales reciben este nombre pues pueden ser caracterizados equivalentemente como ideales en el anillo de polinomios $\mathbb{Z}[x]/\langle f(x)\rangle$, donde $f(x) = x^n + f_n x^{n-1} + \cdots + f_1 \in \mathbb{Z}[x]$.

En particular elegiremos $f = (1, 0, \dots, 0)$, puesto que en este caso concreto se siguen manteniendo todas las pruebas de seguridad y es posible optimizar las operaciones en el anillo resultante. En este caso tenemos que la matriz de la base del retículo es:

$$A = \begin{pmatrix} a_1 & -a_n & -a_{n-1} & \cdots & -a_2 \\ a_2 & a_1 & -a_n & \cdots & -a_3 \\ a_3 & a_2 & a_1 & \cdots & -a_4 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_n & a_{n-1} & a_{n-2} & \cdots & a_1 \end{pmatrix}$$

Habitualmente interpretaremos esta matriz A como el polinomio $a \in \mathbb{Z}[x]/\langle x^n + 1\rangle$.

$$R = \mathbb{Z}[x]/\langle x^n + 1\rangle$$
$$R_q = R/qR$$
$$\chi \text{ distribución gaussiana discreta en } R_q$$

A lo largo de todo el trabajo escribiremos indistintamente $a \in R_q$ como un polinomio $a = a_1 + a_2 x + a_3 x^2 + \cdots + a_n x^{n-1} \in R_q$ o como el vector de sus coeficientes $(a_1, a_2, a_3, \dots, a_n) \in \mathbb{Z}_q^n$, teniendo siempre

en cuenta que aunque escribamos **a** y **p** como vectores entendemos $a \cdot p$ como producto de polinomios en R_q.

Observamos que este producto de polinomios en R_q es equivalente al original producto de una matriz por un vector, donde los a_i y los p_i son los coeficientes de los polinomios tal como los hemos definido anteriormente.

$$a \cdot p = \begin{pmatrix} a_1 & -a_n & -a_{n-1} & \cdots & -a_2 \\ a_2 & a_1 & -a_n & \cdots & -a_3 \\ a_3 & a_2 & a_1 & \cdots & -a_4 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_n & a_{n-1} & a_{n-2} & \cdots & a_1 \end{pmatrix} \begin{pmatrix} p_1 \\ p_2 \\ p_3 \\ \vdots \\ p_n \end{pmatrix}$$

Utilizando estas estructuras Eduard Sanou propone un esquema de cifrado post-cuántico que será el que utilicemos [2], basando su seguridad en la versión $ring - LWE$ del problema LWE restringido a retículos ideales.

El algoritmo generador elige un elemento $a \in R_q$ de forma aleatoria uniforme y otros dos elementos pequeños $s, e \in R$ siguiendo la distribución χ.

- s es la clave privada
- $(a, b = a \cdot s + e) \in R_q \times R_q$ es la clave pública

Para simplificar únicamente consideraremos votos $z \in \{0, 1\}$ de un referendum, pero sería posible utilizar cualquier tipo de voto que se pueda codificar como un polinomio de R_q con coeficientes en un intervalo centrado en el 0 suficientemente pequeño. Para cifrar un voto z escogemos $r, e_1, e_2 \in \chi$ y escribimos:

$$\begin{aligned} u &= a \cdot r + e_1 \mod q \\ v &= b \cdot r + e_2 + \left\lfloor \frac{q}{2} \right\rfloor z \mod q \end{aligned}$$

Para descifrarlo es necesario emplear la clave secreta s:

$$\begin{aligned} v - s \cdot u &= b \cdot r + e_2 + \left\lfloor \frac{q}{2} \right\rfloor z - s(a \cdot r + e_1) \\ &= s \cdot a \cdot r + e \cdot r + e_2 + \left\lfloor \frac{q}{2} \right\rfloor z - s \cdot a \cdot r - s \cdot e_1 \\ &= e \cdot r + e_2 - s \cdot e_1 + \left\lfloor \frac{q}{2} \right\rfloor z \\ &\approx \left\lfloor \frac{q}{2} \right\rfloor z \end{aligned}$$

Cada voto (u, v) lo podemos recifrar en un voto (u', v') de la siguiente forma, eligiendo $r', e_1', e_2' \in \chi$. Definimos ϕ como la función de recifrado.

$$\phi\left((u, v), (r', e_1', e_2')\right) = (u', v')$$

$$\begin{aligned} u' &= u + a \cdot r' + e_1' \mod q \\ v' &= v + b \cdot r' + e_2' \mod q \end{aligned}$$

$$\begin{aligned} v' - s \cdot u' &= v - s \cdot u + (b \cdot r' + e_2') - s(a \cdot r' + e_1') \\ &= e \cdot r + e_2 - s \cdot e_1 + \left\lfloor \frac{q}{2} \right\rfloor z + s \cdot a \cdot r' + e \cdot r' + e_2' - s \cdot a \cdot r' - s \cdot e_1' \\ &= e \cdot r + e_2 - s \cdot e_1 + e \cdot r' + e_2' - s \cdot e_1' + \left\lfloor \frac{q}{2} \right\rfloor z \\ &= e \cdot (r + r') + (e_2 + e_2') - s(e_1 + e_1') + \left\lfloor \frac{q}{2} \right\rfloor z \\ &\approx \left\lfloor \frac{q}{2} \right\rfloor z \end{aligned}$$

Llamamos A y B a las matrices asociadas respectivamente a los polinomios (a, b) de la clave pública.

$$A = \begin{pmatrix} a_1 & -a_n & -a_{n-1} & \cdots & -a_2 \\ a_2 & a_1 & -a_n & \cdots & -a_3 \\ a_3 & a_2 & a_1 & \cdots & -a_4 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_n & a_{n-1} & a_{n-2} & \cdots & a_1 \end{pmatrix} = \left(a_{i,j}\right) \qquad B = \begin{pmatrix} b_1 & -b_n & -b_{n-1} & \cdots & -b_2 \\ b_2 & b_1 & -b_n & \cdots & -b_3 \\ b_3 & b_2 & b_1 & \cdots & -b_4 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ b_n & b_{n-1} & b_{n-2} & \cdots & b_1 \end{pmatrix} = \left(b_{i,j}\right)$$

La dificultad del problema $ring - LWE$ garantiza que sea computacionalmente difícil obtener la clave s y que podamos considerar los textos cifrados como pseudoaleatorios.

3. Pruebas de conocimiento nulo

Una prueba de conocimiento nulo es un protocolo entre un probador $\mathcal{P}$ y un verificador $\mathcal{V}$ en el que el probador intenta demostrar al verificador que conoce un cierto secreto. Podemos formalizar esto de la siguiente manera.

Sea R una relación binaria, es decir, R es un subconjunto de $\{0,1\}^* \times \{0,1\}^*$ en el que imponemos la restricción de que si $(x, w) \in R$ entonces la longitud de w sea como mucho $p(|x|)$ para algún polinomio p. Para $(x, w) \in R$ consideraremos que x es un caso concreto de un determinado problema y w es la solución concreta a ese caso. Generalmente se llama testigo a w.

El ejemplo más clásico de este tipo de relaciones es la del logaritmo discreto o *DL* por sus siglas en inglés.

$$DL = \{(x, w) | x = (p, q, g, h), ord(g) = ord(h) = q, h = g^w\}$$

Donde p, q son primos, $g, h \in \mathbb{Z}_p^*$ y $w \in \mathbb{Z}_q$. En definitiva *DL* contiene todos los problemas de logaritmo discreto de este tipo. Buscamos un método en el que dada una x y una R un probador $\mathcal{P}$ pueda convencer a un verificador $\mathcal{V}$ de que conoce una w tal que $(x, w) \in R$. El ejemplo concreto de este tipo de pruebas que nosotros utilizaremos son los $\Sigma - protocolos$.

Definición 3.1. Un Σ-*protocolo* es un protocolo entre un probador $\mathcal{P}$ y un verificador $\mathcal{V}$ en el que dada una x conocida por ambos $\mathcal{P}$ intenta convencer a $\mathcal{V}$ de que conoce una w tal que $(x, w) \in R$. Tiene una estructura de tres pasos:

1. $\mathcal{P}$ envía un mensaje a
2. $\mathcal{V}$ responde con un reto aleatorio e
3. $\mathcal{P}$ envía una respuesta z, entonces V acepta o rechaza la demostración realizando una serie de comprobaciones sobre la conversación (x, a, e, z).

Y además cumple las siguientes condiciones:

1. Corrección (*completeness*): si el probador $\mathcal{P}$ realmente conoce w tal que $(x, w) \in R$, $\mathcal{V}$ es honesto y ambos siguen el protocolo entonces en el tercer paso $\mathcal{V}$ siempre aceptará la conversación como válida.
2. Consistencia (*soundness*): dado cualquier par de conversaciones aceptadas $(x, a, e, z), (x, a, e', z')$ con $e \neq e'$ es posible utilizarlas para calcular de forma eficiente una w tal que $(x, w) \in R$.
3. Conocimiento nulo (*zero-knowleadge*): existe un simulador que en tiempo polinómico y dados x y una e aleatoria produce una conversación aceptada (x, a, e, z) con la misma distribución de probabilidad que las conversaciones entre $\mathcal{P}$ y $\mathcal{V}$ honestos.

Una generalización inmediata de estos protocolos consiste en relajar la consistencia, permitiendo que para obtener el secreto w sean necesarias hasta m conversaciones que compartan x y a. Se denotan Σ_m-protocolos.

Definición 3.2. Un Σ_m-*protocolo* es un Σ-protocolo en el que la condición de consistencia se ha relajado y se sustituye por:

1. Consistencia (*soundness*): dadas m conversaciones aceptadas que compartan x y a, con $e_i \neq e_j$ para $i \neq j$:

$$(x, a, e_1, z_1)$$
$$(x, a, e_2, z_2)$$
$$\dots$$
$$(x, a, e_m, z_m)$$

   Es posible utilizarlas para calcular de forma eficiente una w tal que $(x, w) \in R$.

Observamos que la validez del protocolo como prueba de conocimiento radica en que se haya realizado en el orden correcto. La conversación por sí misma no revela ninguna información sobre el secreto, pues la propiedad de conocimiento nulo garantiza que es indistinguible de otras conversaciones aceptadas emitidas por un simulador que desconoce el testigo w.

Sin embargo un probador deshonesto tiene únicamente una posibilidad muy pequeña de conseguir que un verficador honesto acepte la conversación sin conocer el testigo w. El mensaje a ha sido enviado antes de que el verificador escoja aleatoriamente el reto. Si el probador pudiera responder satisfactoriamente a dos retos diferentes entonces la consistencia nos garantiza que también podría calcular de forma eficiente el testigo w.

Esto significa que, como mucho, un probador deshonesto que no sepa calcular ningún testigo w únicamente podría responder a uno de los posibles retos que envíe el verificador. Como este reto se elige al azar después de que el probador envíe a, las posibilidades de que el probador pueda engañar al verificador decaen exponencialmente con el número de bits del reto e.

La corrección por su parte nos garantiza que en caso de que el probador y el verificador sean honestos y sigan el protocolo siempre llegarán a una conversación aceptada.

Hasta el momento hemos descrito únicamente pruebas de conocimiento nulo interactivas, en las que se requiere una comunicación entre el probador y el verificador. Estas pruebas pueden hacerse no interactivas si sustituimos el paso en el que el verificador envía un reto aleatorio por una función *hash* del mensaje a enviado por el probador. En el Modelo del Oráculo Aleatorio podemos ver esta función hash como una función aleatoria, y el hecho de depender de a juega un papel equivalente al hecho de que el verificador no envía su reto hasta haber recibido este mensaje. Esto es lo que se conoce como la heurística de Fiat-Shamir.

4. Compromisos

Los esquemas de compromisos permiten comprometerse a un mensaje sin revelar información sobre él, con la posibilidad de posteriormente abrir el compromiso, revelar el mensaje y probar que se había comprometido a ese mismo mensaje desde el principio.

La utilidad de este tipo de esquemas es permitir demostrar que el mensaje en cuestión ya había sido determinado en el momento en que el compromiso se hace público, aunque su contenido no se pueda revelar en ese momento.

Definición 4.1. Un *esquema de compromisos* consta de tres algoritmos:

- **Gen**: el generador recibe como entrada un parámetro de seguridad 1^λ y produce como salida una clave de compromiso pública pk.
- **Com**: el algoritmo de compromisos recibe como entrada una clave pública pk y un mensaje $M \in \mathcal{M}$ y produce como salida un par de elementos (c, d) a los que respectivamente llamaremos compromiso y apertura. Este algoritmo habitualmente es probabilístico, por lo que en ocasiones explicitaremos el elemento aleatorio α que utiliza y escribiremos $(c, d) \leftarrow \mathrm{Com}(m, \alpha)_{pk}$
- **Ver**: el algoritmo de verificación recibe como entrada una clave pública pk, un mensaje m, un compromiso c y una apertura d, a partir de los cueles acepta o rechaza la apertura.

Y además cumple las siguientes condiciones:

1. Corrección: el algoritmo de verificación Ver siempre acepta aperturas honestas, es decir, aquellas en las que tanto el compromiso c como la apertura d sean producto de ejecutar el algoritmo de compromiso Com sobre el mensaje m.

   Si $(c, d) \leftarrow \mathrm{Com}(m, \alpha)_{pk}$ para alguna α del espacio de elementos aleatorios que utilice el algoritmo, entonces el algoritmo verificador acepta siempre.

2. Vinculante: un compromiso no puede ser abierto a mensajes diferentes.

   Es computacionalmente vinculante si no se conoce ningún algoritmo eficiente que pueda generar un compromiso c con dos aperturas d, d' de forma que el verificador acepte ambas.

   Es perfectamente vinculante si es imposible abrir un compromiso a más de un mensaje.

$$\mathrm{Ver}(pk, m, c, d) = \text{acepta} \wedge \mathrm{Ver}(pk, m', c, d') = \text{acepta} \implies m = m'$$

3. Esconde: el compromiso no revela información sobre el mensaje.

   Esconde computacionalmente si, dados dos mensajes elegidos por un adversario, podemos comprometernos a uno de ellos y hacer público el compromiso, y la probabilidad de que el adversario acierte a cuál de ellos nos hemos comprometido es como mucho $1/2 + f(x)$, donde $f(x)$ es una función negligible.

$$\Pr\left[b = b' \;\middle|\; \begin{array}{c} pk \xleftarrow{\$} \mathrm{Gen}(1^\lambda), \quad (m_0, m_1, aux) \xleftarrow{\$} A_1(pk) \\ b \xleftarrow{\$} \{0,1\}, \quad (c, d) \xleftarrow{\$} \mathrm{Com}(m_b)_{pk}, \quad b' \xleftarrow{\$} A_2(c, aux) \end{array} \right] \leq \frac{1}{2} + f(x)$$

Donde *aux* es cualquier información auxiliar que el atacante quiera guardar en el momento en el que escoge los mensajes.

Esconde perfectamente si el compromiso puede ser abierto a cualquier mensaje, y por lo tanto es imposible conocer cuál es el mensaje que se ha comprometido, pues podría ser cualquiera.

Observamos que un compromiso que esconde perfectamente únicamente puede ser computacionalmente vinculante. Esto es lo que necesitamos para ofrecer seguridad a largo plazo, pues ningún adversario podría en el futuro conocer el mensaje si no lo hemos llegado a abrir, pero sin embargo para las pruebas en las que lo hayamos utilizado en ese momento es suficiente que sea computacionalmente vinculante.

En particular pueden ser computacionalmente vinculantes únicamente considerando computación clásica, y podría romperse empleando computación cuántica. Sin embargo en ese caso solo será necesario utilizar otros compromisos más seguros cuando los ordenadores cuánticos realmente existan, pues hasta ese momento las pruebas seguirán siendo válidas y vinculantes. A partir de entonces los ordenadores cuánticos harán que las nuevas pruebas dejen de ser confiables, pero no podrán obtener ninguna información de los compromisos realizados con anterioridad.

4.1 Compromisos de Pedersen

Para comprometernos a un valor $x \in \mathbb{Z}_q$ utilizaremos compromisos de Pedersen [4] $c = \mathrm{Com}(x, \alpha)_{g,h} = g^{\alpha} h^{x}$, donde g y h son dos generadores públicos de un grupo G_q de orden q.

En este esquema de compromiso la clave pública son los dos generadores $pk = (g, h)$ y el parámetro de seguridad 1^{λ} define la longitud en bits de q.

La apertura d se corresponde con el elemento aleatorio $\alpha \in \mathbb{Z}_q$ que se ha utilizado al generar el compromiso, por lo que el verificador acepta la apertura comprobando $c \stackrel{?}{=} g^{\alpha} h^{x}$. De esta forma la corrección es inmediata.

Es esquema de compromiso esconde perfectamente puesto que un compromiso c puede ser abierto a cualquier $y \in \mathbb{Z}_q$, simplemente haciendo $\alpha' = \log_g ch^{-y}$.

Es por otra parte computacionalmente vinculante, puesto que si un adversario es capaz de abrir un compromiso c a dos mensajes $x \neq y$ entonces también podría calcular $\log_h g$.

$$
\begin{aligned}
& x \neq y \\
& c = g^{\alpha} h^{x} = g^{\alpha'} h^{y} \\
& \quad \Longrightarrow \ \alpha \neq \alpha' \mod q \\
& \quad \Longrightarrow \ g^{\alpha - \alpha'} = h^{x-y} \\
& \quad \Longrightarrow \ (\alpha - \alpha') \log_h g = x - y \mod q \\
& \quad \Longrightarrow \ \log_h g = \frac{x-y}{\alpha - \alpha'} \mod q
\end{aligned}
$$

Por lo tanto abrir un compromiso de Pedersen a dos mensajes diferentes es al menos tan difícil como calcular un logaritmo discreto. Además ya hemos visto que, de hecho, saber calcular logaritmos discretos de forma eficiente nos permitiría abrir cualquier compromiso al mensaje que queramos, por lo que en un escenario de computación cuántica no solo dejaríamos de tener una prueba de seguridad sino que conoceríamos el algoritmo que nos permite abrir los compromisos.

Estamos intentando construir un sistema de votación seguro a largo plazo utilizando cifrado ring-LWE y sin embargo empleamos un tipo de compromiso basado en el logaritmo discreto. Hemos de explicar que esto no supone un problema para la seguridad a largo plazo.

El compromiso de Pedersen esconde perfectamente y es computacionalmente vinculante bajo la suposición del logaritmo discreto. Al poder ser abierto a cualquier mensaje es imposible obtener ninguna información sobre el mensaje original.

La aparición de ordenadores cuánticos obligará a utilizar otro tipo de compromisos (proponemos utilizar algún tipo de compromiso homomórfico basado en ring-LWE como el propuesto por Benahamouda [7].

Para comprometeremos a un vector $\mathbf{x} = (x_1, \dots, x_n)$ utilizaremos la versión generalizada del compromiso de Pedersen con $g, g_1, \dots, g_n$ generadores. $\text{Com}(\mathbf{x}, \alpha) = g^{\alpha} \prod_{i=1}^{n} g_i^{x_i}$.

Observamos que si en los compromisos a elementos individuales utilizamos g_i como el generador h entonces podemos obtener un compromiso al vector a partir del compromiso a cada uno de los elementos:

$$\text{Com}(\mathbf{x}, \alpha_1 + \cdots + \alpha_n) = \prod_{i=1}^{n} g^{\alpha_i} g_i^{x_i} = \prod_{i=1}^{n} \text{Com}(x_i, \alpha_i)_{g,g_i}$$

Para comprometernos a una matriz $M \in \mathbb{Z}^{n \times n}$ nos comprometeremos a cada una de sus columnas $(\mathbf{m_1}, \dots, \mathbf{m_n})$. Es decir, el compromiso a una matriz es un vector donde cada uno de sus elementos es un compromiso a cada una de las columnas de la matriz.

$$\text{Com}(M, \alpha_1, \dots, \alpha_n) = (\text{Com}(\mathbf{m_1}, \alpha_1), \dots, \text{Com}(\mathbf{m_n}, \alpha_n)) = \left(g^{\alpha_1} \prod_{i=1}^{n} g_i^{m_{i,1}}, \dots, g^{\alpha_n} \prod_{i=1}^{n} g_i^{m_{i,n}} \right)$$

El producto de una matriz por un vector es una combinación lineal de las columnas de la matriz con coeficientes los elementos del vector. Por ello a partir del compromiso de una matriz $\text{Com}(M, \boldsymbol{\alpha}) = (c_{\mathbf{m_1}}, \dots, c_{\mathbf{m_n}})$ podemos obtener un compromiso del producto de la matriz M por un vector $\mathbf{x}$.

$$\prod_{i=1}^{n} c_{\mathbf{m_i}}^{x_i} = \prod_{i=1}^{n} \left(g^{\alpha_i} \prod_{j=1}^{n} g_j^{m_{j,i}} \right)^{x_i} = g^{\langle \boldsymbol{\alpha}, \mathbf{x} \rangle} \prod_{j=1}^{n} g_j^{\left\langle (m_{j,1}, \dots, m_{j,n}), (x_1, \dots, x_n) \right\rangle} = \text{Com}(M\mathbf{x}, \langle \boldsymbol{\alpha}, \mathbf{x} \rangle)$$

Este tipo de relaciones entre los compromisos y los mensajes comprometidos nos permitirá comprometernos a elementos individuales, probar características sobre estos elementos comprometidos y posteriormente operarlos y abrir el compromiso del resultado final, sin revelar información sobre los elementos originales.

5. Nodo Mixnet

Tal y como hemos explicado en el primer apartado de este trabajo, para garantizar que el contenido de cada voto sea secreto los votos se envían cifrados, y para asegurar que han sido emitidos por votantes inscritos en el censo estos votos se acompañan de una firma que pruebe quién lo ha emitido. El servidor que recibe los votos comprueba de esta forma que la firma se corresponde a un votante registrado en el censo y que cada votante emite un único voto.

Si establecemos una analogía con las votaciones físicas en una urna la firma electrónica sería equivalente a presentar un documento de identidad en la mesa electoral antes de depositar el voto en la urna. En esta analogía los votos cifrados equivaldrían a los sobres cerrados, y solo al abrir el sobre (descifrar el voto) se conocería su contenido. Sin embargo mientras que los sobres cerrados son indistingibles entre sí cada voto cifrado es único, por lo que un adversario únicamente tendría que interceptar los votos cifrados, comprobar a quién pertenece la firma de cada voto cifrado y esperar a que finalmente el servidor de la mesa electoral lo descifre. En la práctica estos votos cifrados se hacen públicos para que cualquiera pueda verificar que la votación se ha realizado correctamente.

Como hemos dicho existen dos alternativas para solucionar este problema. Si el esquema de cifrado es homomórfico sería posible operar con los votos cifrados y descifrar únicamente el resultado global. Esta opción es factible con el esquema de cifrado propuesto basado en ring-LWE, puesto que es un sistema de cifrado homomórfico. Sin embargo solo es aplicable para sistemas electorales en los que el resultado sea únicamente la suma de votos individuales, pero no serviría para sistemas que permitan *write-ins* (escribir el nombre de un candidato que no aparezca en la papeleta) u otro tipo de sistemas electorales más complejos.

Por ello la segunda alternativa, más flexible que la anterior, consiste en permutar y realeatorizar los votos de forma que ningún adversario pueda relacionar individualmente los votos cifrados originales con sus correspondientes votos permutados y realeatorizados. Esto es lo que se conoce como un *nodo mixnet*. Para evitar que un nodo mixnet deshonesto o comprometido filtre a un adversario información sobre la permutación que ha utilizado, en un proceso de votación se emplean varios nodos mixnet en serie, tomando como entrada la salida del anterior, de forma que si al menos uno de ellos es honesto es imposible para un adversario relacionar los votos iniciales firmados por los votantes con los votos realeatorizados que finalmente se descifran.

Sin embargo, dado que el objetivo del nodo mixnet es evitar que se pueda relacionar los votos que toma como entrada de los que produce como salida, podría no ser honesto y sustituir o modificar alguno de los votos. Es por ello que es necesaria una prueba de conocimiento nulo con la que cada nodo mixnet demuestre que conoce una permutación π tal que si la entrada del nodo mixnet son N votos, cada uno compuesto por una pareja (u, v), a los que llamaremos $\left(u^{(1)}, v^{(1)}, \dots, u^{(N)}, v^{(N)}\right)$ entonces la salida $\left(u''^{(1)}, v''^{(1)}, \dots, u''^{(N)}, v''^{(N)}\right)$ está formada por esos mismos votos realeatorizados $(u', v') = \phi\left((u, v), \mathbf{r}, \mathbf{e}_1, \mathbf{e}_2\right)$ y permutados utilizando π, sin revelar ninguna información sobre la permutación y los elementos aleatorios utilizados.

$$\Sigma\text{-proof}\left[\begin{array}{c} \pi \\ \mathbf{r}^{(1)}, \dots, \mathbf{r}^{(N)} \\ \mathbf{e}_1^{(1)}, \dots, \mathbf{e}_1^{(N)} \\ \mathbf{e}_2^{(1)}, \dots, \mathbf{e}_2^{(N)} \end{array} \middle| \left(u''^{(1)}, v''^{(1)}, \dots, u''^{(N)}, v''^{(N)}\right)^T = \begin{pmatrix} \phi\left(\left(u^{\pi(1)}, v^{\pi(1)}\right), \mathbf{r}^{(1)}, \mathbf{e}_1^{(1)}, \mathbf{e}_2^{(1)}\right)^T \\ \dots \\ \phi\left(\left(u^{\pi(N)}, v^{\pi(N)}\right), \mathbf{r}^{(N)}, \mathbf{e}_1^{(N)}, \mathbf{e}_2^{(N)}\right)^T \end{pmatrix}\right]$$

Sabiendo que podemos identificar los polinomios con sus coeficientes, y que el producto de polinomios puede escribirse como el producto de una matriz por un vector podemos describir el comportamiento del nodo mixnet de forma matricial:

$$\underset{N\times 2n}{\begin{pmatrix} u_1''^{(1)} & \cdots & u_n''^{(1)} & v_1''^{(1)} & \cdots & v_n''^{(1)} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ u_1''^{(N)} & \cdots & u_n''^{(N)} & v_1''^{(N)} & \cdots & v_n''^{(N)} \end{pmatrix}} =$$

$$= \underset{N\times N}{\begin{pmatrix} m_{11} & m_{12} & \cdots & m_{1N} \\ m_{21} & m_{22} & \cdots & m_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ m_{N1} & m_{N2} & \cdots & m_{NN} \end{pmatrix}} \underset{N\times 2n}{\begin{pmatrix} u_1^{(1)} & \cdots & u_n^{(1)} & v_1^{(1)} & \cdots & v_n^{(1)} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ u_1^{(N)} & \cdots & u_n^{(N)} & v_1^{(N)} & \cdots & v_n^{(N)} \end{pmatrix}}$$

$$+ \underset{N\times n}{\begin{pmatrix} r_1^{(1)} & \cdots & r_n^{(1)} \\ \vdots & \ddots & \vdots \\ r_1^{(N)} & \cdots & r_n^{(N)} \end{pmatrix}} \underset{n\times 2n}{\begin{pmatrix} a_1 & a_2 & a_3 & \cdots & a_n & b_1 & b_2 & b_3 & \cdots & b_n \\ -a_n & a_1 & a_2 & \cdots & a_{n-1} & -b_n & b_1 & b_2 & \cdots & b_{n-1} \\ -a_{n-1} & -a_n & a_1 & \cdots & a_{n-2} & -b_{n-1} & -b_n & b_1 & \cdots & b_{n-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ -a_2 & -a_3 & -a_4 & \cdots & a_1 & -b_2 & -b_3 & -b_4 & \cdots & b_1 \end{pmatrix}}$$

$$+ \underset{N\times 2n}{\begin{pmatrix} e_{1,1}^{(1)} & \cdots & e_{1,n}^{(1)} & e_{2,1}^{(1)} & \cdots & e_{2,n}^{(1)} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ e_{1,1}^{(N)} & \cdots & e_{1,n}^{(N)} & e_{2,1}^{(N)} & \cdots & e_{2,n}^{(N)} \end{pmatrix}}$$

$$\begin{pmatrix} U'' & V'' \end{pmatrix} = M \begin{pmatrix} U & V \end{pmatrix} + R \begin{pmatrix} A^T & B^T \end{pmatrix} + \begin{pmatrix} E_1 & E_2 \end{pmatrix} \tag{1}$$

Las matrices $\begin{pmatrix} U'' & V'' \end{pmatrix}, \begin{pmatrix} U & V \end{pmatrix}, A, B$ son públicas. Para verificar que el nodo mixnet se comporta correctamente hemos de demostrar que conocemos las matrices M, R, E_1, E_2 de forma que los votos a la entrada y a la salida verifiquen la ecuación 1, que M es una matriz de permutación y que los coeficientes de R, E_1, E_2 son pequeños (donde el tamaño que consideraremos pequeño viene determinado por la distribución χ y en la práctica consistirá en comprobar que todos sus elementos se encuentran entre $-\beta$ y β). De esta forma efectivamente la salida se trata de un realeatorizado y permutado de la entrada.

Antes de la votación nos comprometemos a las cuatro matrices y publicamos los siguientes compromisos:

$$c_{\mathbf{m}_j} = \text{Com}\left(\mathbf{m_j}, \alpha_{m_j}\right) = g^{\alpha_{m_j}} \prod_{i=1}^{N} g_i^{m_{i,j}} \qquad \forall j \in 1 \div N$$

$$c_{\mathbf{r}_j} = \text{Com}\left(\mathbf{r_j}, \alpha_{r_j}\right) = g^{\alpha_{r_j}} \prod_{i=1}^{N} g_i^{r_j^{(i)}} \qquad \forall j \in 1 \div n$$

$$c_{\mathbf{e}_{1,j}} = \text{Com}\left(\mathbf{e_{1,j}}, \alpha_{e_{1,j}}\right) = g^{\alpha_{e_{1,j}}} \prod_{i=1}^{N} g_i^{e_{1,j}^{(i)}} \qquad \forall j \in 1 \div n$$

$$c_{\mathbf{e}_{2,j}} = \text{Com}\left(\mathbf{e_{2,j}}, \alpha_{e_{2,j}}\right) = g^{\alpha_{e_{2,j}}} \prod_{i=1}^{N} g_i^{e_{2,j}^{(i)}} \qquad \forall j \in 1 \div n$$

A partir de estos compromisos podemos construir compromisos de los productos de matrices y de su suma, de forma que:

$$c_{\mathbf{e}_{1,i}} \prod_{j=1}^{N} c_{\mathbf{m}_j}^{u_i^{(j)}} \prod_{j=1}^{n} c_{\mathbf{r}_j}^{a_{i,j}} =$$
$$= \text{Com}\left(\left(u_i''^{(1)}, \ldots, u_i''^{(N)}\right), \alpha_{e_{1,i}} + \left\langle \boldsymbol{\alpha}_M, \left(u_i^{(1)}, \ldots, u_i^{(N)}\right)\right\rangle + \langle \boldsymbol{\alpha}_r, (a_{i,1}, \ldots, a_{i,n})\rangle\right) \quad \forall i \in 1 \div n$$
$$c_{\mathbf{e}_{2,i}} \prod_{j=1}^{N} c_{\mathbf{m}_j}^{v_i^{(j)}} \prod_{j=1}^{n} c_{\mathbf{r}_j}^{b_{i,j}} =$$
$$= \text{Com}\left(\left(v_i''^{(1)}, \ldots, v_i''^{(N)}\right), \alpha_{e_{2,i}} + \left\langle \boldsymbol{\alpha}_M, \left(v_i^{(1)}, \ldots, v_i^{(N)}\right)\right\rangle + \langle \boldsymbol{\alpha}_r, (b_{i,1}, \ldots, b_{i,n})\rangle\right) \quad \forall i \in 1 \div n$$

Todos los elementos necesarios para calcular estos compromisos son públicos, por lo que lo único que tiene que hacer el probador es abrirlos públicamente mostrando:

$$\left(\left\langle \boldsymbol{\alpha}_M, \left(u_i^{(1)}, \ldots, u_i^{(N)}\right)\right\rangle + \langle \boldsymbol{\alpha}_r, (a_{i,1}, \ldots, a_{i,n})\rangle + \alpha_{e_{1,i}}\right) \quad \forall i \in 1 \div n$$
$$\left(\left\langle \boldsymbol{\alpha}_M, \left(v_i^{(1)}, \ldots, v_i^{(N)}\right)\right\rangle + \langle \boldsymbol{\alpha}_r, (b_{i,1}, \ldots, b_{i,n})\rangle + \alpha_{e_{2,i}}\right) \quad \forall i \in 1 \div n$$

El verificador solo tendrá que comprobar que efectivamente son una apertura de compromisos a los votos de la salida para verificar que realmente ha utilizado las matrices M, R, E_1, E_2 a las que el probador se había comprometido en el inicio.

A partir de los compromisos de M, R, E_1, E_2 podemos obtener compromisos de la matriz de votos permutados $M\begin{pmatrix} U & V \end{pmatrix}$ y de la matriz de realeatorizado $\left(R\begin{pmatrix} A^T & B^T \end{pmatrix} + \begin{pmatrix} E_1 & E_2 \end{pmatrix}\right)$. Observamos que las $2n$ combinaciones lineales de las $\alpha_{m_j}, \alpha_{r_j}, \alpha_{e_{1,j}}, \alpha_{e_{2,j}}$ que el probador desvela nos permiten abrir los compromisos a la suma de estas dos matrices, pero no a cada una de ellas por separado. En total estas dos matrices por separado utilizan $4n$ elementos aleatorios en los compromisos de sus columnas, y únicamente se muestran $2n$, sumados dos a dos, por lo que es imposible diferenciarlas.

Estamos utilizando que dados dos elementos aleatorios secretos podemos revelar su suma sin filtrar información sobre ninguno de ellos, es decir, mostrar $\left(\left\langle \boldsymbol{\alpha}_M, \left(u_i^{(1)}, \ldots, u_i^{(N)}\right)\right\rangle + \langle \boldsymbol{\alpha}_r, (a_{i,1}, \ldots, a_{i,n})\rangle + \alpha_{e_{1,i}}\right)$

no nos permite averiguar los valores de $\left(\left\langle \boldsymbol{\alpha}_M, \left(u_i^{(1)}, \dots, u_i^{(N)}\right)\right\rangle\right)$ y $\left(\langle \boldsymbol{\alpha}_r, (a_{i,1}, \dots, a_{i,n})\rangle + \alpha_{e_{1,i}}\right)$ por separado, que son los que podrían desvelar el comportamiento del nodo mixnet (y análogamente con el resto de combinaciones lineales mostradas).

Solo quedaría por tanto demostrar que efectivamente M es una matriz de permutación y R, E_1, E_2 tienen coeficientes pequeños. Ambas demostraciones se pueden realizar previamente durante una fase *offline*, quedando para el momento de la votación únicamente el envío de los elementos aleatorios por parte del probador y verificar que son una apertura de los compromisos por parte del verificador.

Sea M la matriz de la permutación π. Podemos demostrar que efectivamente es una matriz de permutación tal como hace Wikström [5] utilizando el siguiente teorema.

Teorema 5.1. *$M \in \mathbb{Z}_q^{N\times N}$ es una matriz de permutación sí y solo sí cumple las siguientes condiciones.*

1. $M\mathbf{1} = \mathbf{1}$
2. *Sea $\mathbf{x}' = M\mathbf{x}$, donde $\mathbf{x} = (x_1, \dots, x_N)$ es un vector de N variables independientes entonces $\prod_{i=1}^N x_i = \prod_{i=1}^N x_i'$*

Demostración. ($\Longrightarrow$)

Sea $M \in \mathbb{Z}_q^{N\times N}$ la matriz de la permutación π.

$M\mathbf{1} = \mathbf{1}$ y por lo tanto verifica 1.

$\prod_{i=1}^N x_i' = \prod_{i=1}^N x_{\pi(i)} = \prod_{i=1}^N x_i$ por lo que también verifica la condición 2.

($\Longleftarrow$)

Sea $M \in \mathbb{Z}_q^{N\times N}$ una matriz que verifica las condiciones 1 y 2.

Consideramos el polinomio $f(\mathbf{x}) = \prod_{i=1}^N x_i' = \prod_{i=1}^N \left(\sum_{j=1}^N m_{i,j}x_j\right)$. Si alguna de las filas de la matriz fuera un vector de ceros el polinomio f sería 0. Sabemos por la condición 2 que $f(\mathbf{x}) = \prod_{i=1}^N x_i$, por lo tanto podemos garantizar que todas las filas de M tienen al menos un elemento distinto de cero.

Si la fila i de M tuviera dos elementos diferentes de cero entonces el polinomio f tendría un factor $\sum_{j\in J} m_{i,j}x_j, \quad m_{i,j} \neq 0\ \forall j \in J$, y al pertenecer a un dominio de factorización única esto contradice que f sea de la forma $\prod_{i=1}^N x_i$.

Si la columna j tuviera más de un elemento diferente de 0 entonces $\deg_{x_j} f > 1$, lo que también contradice que $f = \prod_{i=1}^N x_i$. Podemos concluir que M es una matriz que tiene exactamente un elemento diferente de 0 en cada fila y en cada columna. La condición 1 nos asegura que estos elementos han de ser 1. □

En nuestro caso tenemos una matriz M comprometida por columnas $\mathrm{Com}(M, \vec{\alpha_M})$ y hemos de probar que cumple estas dos condiciones. Comprobaremos la igualdad de polinomios verificando que se da la igualdad al evaluarlo en un vector aleatorio $\boldsymbol{\lambda} = (\lambda_1, \dots, \lambda_N) \in \mathbb{Z}_q^N$ enviado como reto por el verificador. La probabilidad de que se de esta igualdad siendo diferentes los polinomios viene acotada por el lema de Schwartz-Zippel.

Para ello emplearemos exactamente la prueba propuesta por Wikström, teniendo en cuenta que los valores $t = \left\langle \vec{\alpha_M}, \vec{1}\right\rangle$ y $k = \left\langle \vec{\alpha_M}, \vec{\lambda}\right\rangle$ que la prueba de Wikström considera secretos no serguirían permaneciendo secretos a largo plazo, pues el protocolo propuesto por Wikström esconde estos elementos basándose en la dificultad del logaritmo discreto. Sin embargo esto no es un problema, pues de nuevo

revelar dos combinaciones lineales de estos elementos secretos no ofrece información sobre cada uno de ellos.

$$\Sigma\text{-proof}\left[\boldsymbol{\lambda}' \in \mathbb{Z}_q^N, t, k, z \in \mathbb{Z}_q \;\middle|\; \left(\mathrm{Com}\left(\mathbf{1}, t\right) = a^{\mathbf{1}} \wedge \mathrm{Com}\left(\boldsymbol{\lambda}', k\right) = a^{\boldsymbol{\lambda}} \wedge \prod_{i=1}^{N} \lambda_i = \prod_{i=1}^{N} \lambda_i'\right) \vee g_1 = g^z\right]$$

$$\begin{aligned}
\mathbf{r}, \mathbf{s} &\xleftarrow{\$} \mathbb{Z}_q^N \\
s_\alpha, s_\gamma, s_\delta &\xleftarrow{\$} \mathbb{Z}_q \\
s' &\xleftarrow{\$} \left[0, 2^{n_v+n_r+n_c} - 1\right] \\
B_0 &= g_1 \\
B_i &= g^{r_i} B_{i-1}^{\lambda_i'} \\
\alpha &= g^{s_\alpha} \prod_{i=1}^{N} g_i^{s_i'} \\
\beta_i &= g^{s_i} B_{i-1}^{s_i'} \\
\gamma &= g^{s_\gamma} \\
\delta &= g^{s_\delta}
\end{aligned}$$

$$\mathcal{P} \xrightarrow{B_0, B_i, \alpha, \beta_i, \gamma, \delta} \mathcal{V}$$

$$c \xleftarrow{\$} [0, 2^{n_c} - 1]$$

$$\mathcal{P} \xleftarrow{c} \mathcal{V}$$

$$\begin{aligned}
\lambda_1'' &= s_1 \\
\lambda_i'' &= \lambda_{i-1}'' \lambda_i' + s_i \mod q \\
d_\alpha &= ck + s_\alpha \mod q \\
d_i' &= c\lambda_i' + s_i' \mod q \\
d_i &= cr_i + s_i \mod q \\
d_\gamma &= c\left\langle \mathbf{s}, \mathbf{1} \right\rangle + s_\gamma \mod q \\
d_\delta &= c\lambda_N'' + s_\delta \mod q
\end{aligned}$$

$$\mathcal{P} \xrightarrow{d_\alpha, d_i', d_i, d_\gamma, d_\delta} \mathcal{V}$$

$$\begin{aligned}
\left(a^{\boldsymbol{\lambda}}\right)^c \alpha &\stackrel{?}{=} g^{d_\alpha} \prod_{i=1}^{N} g_i^{d_i'} \\
B_i^c \beta_i &\stackrel{?}{=} g^{d_i} B_{i-1}^{d_i'} \\
\left(a^{\mathbf{1}} \Big/ \prod_{i=1}^{N} g_i\right)^c \gamma &\stackrel{?}{=} g^{d_\gamma} \\
\left(B^N \Big/ g^{\prod_{i=1}^{N} \lambda_i}\right)^c \delta &\stackrel{?}{=} g^{d_\delta}
\end{aligned}$$

El protocolo anterior es perfectamente correcto, consistente y estadísticamente de conocimiento nulo [5]. Podemos construir un simulador eligiendo $B_1, \dots, B_N \in G_q$, $\mathbf{d}, \mathbf{d}' \in \mathbb{Z}_q^N$ y $d_\alpha, d_\gamma, d_\delta \in \mathbb{Z}_q$ aleatoriamente, y calculando $\alpha, \beta_i, \gamma, \delta$ mediante las ecuaciones de verificación. Para demotrar la consistencia es necesario deshacer las recurrencias construidas, tal como se explica en el artículo de Wikström.

Hemos de demostrar también que los coeficientes aleatorios $r_j^i, e_{1,j}^i, e_{2,j}^i$ son realmente pequeños, en nuestro caso requeriremos que pertenezcan al intervalo $[-\beta+1, \beta-1]$, con $\beta = 2^k$. Para ello utilizaremos la estrategia empleada por Ling en sus pruebas de conocimiento nulo de problemas basados en retículos [6].

Descomponemos $r_j^i = \sum_{l=0}^{k-1} r_{j,l}^i 2^l, \quad r_{j,l}^i \in \{-1, 0, 1\}$, y análogamente $e_{1,j}^i = \sum_{l=0}^{k-1} e_{1,j,l}^i 2^l$ y $e_{2,j}^i = \sum_{l=0}^{k-1} e_{2,j,l}^i 2^l$, con $e_{1,j,l}^i, e_{2,j,l}^i \in \{-1, 0, 1\}$. Estas pertenencias se pueden demostrar mediante una $OR-proof$, y se puede hacer en la parte *offline*. Posteriormente a partir de los compromisos a cada uno de los *bits* de la descomposición obtendremos un compromiso de los coeficientes, y a partir de estos un compromiso de cada columna de las matrices correspondientes.

Para escribirlos en forma matricial tenemos:

$$\underset{nN\times 1}{\begin{pmatrix} r_1^{(1)} \\ r_2^{(1)} \\ \vdots \\ r_n^{(1)} \\ r_1^{(2)} \\ \vdots \\ r_n^{(N)} \end{pmatrix}} = \underset{nN\times k}{\begin{pmatrix} r_{1,0}^{(1)} & r_{1,1}^{(1)} & \cdots & r_{1,k-1}^{(1)} \\ r_{2,0}^{(1)} & r_{2,1}^{(1)} & \cdots & r_{2,k-1}^{(1)} \\ \vdots & \vdots & \ddots & \vdots \\ r_{n,0}^{(1)} & r_{n,1}^{(1)} & \cdots & r_{n,k-1}^{(1)} \\ r_{1,0}^{(2)} & r_{1,1}^{(2)} & \cdots & r_{1,k-1}^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ r_{n,0}^{(N)} & r_{n,1}^{(N)} & \cdots & r_{n,k-1}^{(N)} \end{pmatrix}} \underset{k\times 1}{\begin{pmatrix} 2^0 \\ 2^1 \\ \vdots \\ 2^{k-1} \end{pmatrix}}$$

$$\underset{2nN\times 1}{\begin{pmatrix} e_{1,1}^{(1)} \\ \vdots \\ e_{1,n}^{(1)} \\ e_{2,1}^{(1)} \\ \vdots \\ e_{2,n}^{(1)} \\ e_{1,1}^{(2)} \\ \vdots \\ e_{2,n}^{(N)} \end{pmatrix}} = \underset{2nN\times k}{\begin{pmatrix} e_{1,1,0}^{(1)} & e_{1,1,1}^{(1)} & \cdots & e_{1,1,k-1}^{(1)} \\ \vdots & \vdots & \ddots & \vdots \\ e_{1,n,0}^{(1)} & e_{1,n,1}^{(1)} & \cdots & e_{1,n,k-1}^{(1)} \\ e_{2,1,0}^{(1)} & e_{2,1,1}^{(1)} & \cdots & e_{2,1,k-1}^{(1)} \\ \vdots & \vdots & \ddots & \vdots \\ e_{2,n,0}^{(1)} & e_{2,n,1}^{(1)} & \cdots & e_{2,n,k-1}^{(1)} \\ e_{1,1,0}^{(2)} & e_{1,1,1}^{(2)} & \cdots & e_{1,1,k-1}^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ e_{2,n,0}^{(N)} & e_{2,n,1}^{(N)} & \cdots & e_{2,n,k-1}^{(N)} \end{pmatrix}} \underset{k\times 1}{\begin{pmatrix} 2^0 \\ 2^1 \\ \vdots \\ 2^{k-1} \end{pmatrix}}$$

Aunque genéricamente escribamos los compromisos con un generador h en realidad utilizaremos g_i según corresponda. Es decir, los compromisos que tendremos serán $c_{r_j^{(i)}} = \text{Com}\left(r_j^{(i)}, \alpha_{r_j^{(i)}}\right)_{g,g_i}$, $c_{e_{1,j}^{(i)}} = \text{Com}\left(e_{1,j}^{(i)}, \alpha_{e_{1,j}^{(i)}}\right)_{g,g_i}$ y $c_{e_{2,j}^{(i)}} = \text{Com}\left(e_{2,j}^{(i)}, \alpha_{e_{2,j}^{(i)}}\right)_{g,g_i}$, y por tanto también tendremos que utilizar estos g_i en los compromisos de los *bits*.

El protocolo para demostrar que $x \in \{-1, 0, 1\}$ está basado en una combinación de las pruebas individuales para demostrar cada uno de los valores.

$$\Sigma\text{-proof}\,[x \mid x \in \{-1, 0, 1\}, c = g^r h^x]$$

Entrada común: c

La prueba consiste en la realización de tres pruebas simultáneas $x = -1, x = 0, x = 1$ de las cuales dos serán simuladas y solo aquella que se corresponda con el verdadero valor de x será real y dependerá efectivamente del reto que nos envíe el verificador. La corrección, consistencia y conocimiento nulo se derivan de las de las pruebas individuales.

$$s, t_{x+1}, t_{x-1}, e_{x+1}, e_{x-1} \xleftarrow{\$} \mathbb{Z}_q$$

$$d_0 = \begin{cases} g^{t_0} c^{-e_0} & \text{si } x \neq 0 \\ g^s & \text{si } x = 0 \end{cases}$$

$$d_1 = \begin{cases} g^{t_1} (c/h)^{-e_1} & \text{si } x \neq 1 \\ g^s & \text{si } x = 1 \end{cases}$$

$$d_{-1} = \begin{cases} g^{t_{-1}} (ch)^{-e_{-1}} & \text{si } x \neq -1 \\ g^s & \text{si } x = -1 \end{cases}$$

$$\mathcal{P} \xrightarrow{d_0, d_1, d_{-1}} \mathcal{V}$$

$$k \xleftarrow{\$} \mathbb{Z}_q$$

$$\mathcal{P} \xleftarrow{k} \mathcal{V}$$

$$e_x = k - e_{x+1} - e_{x-1}$$
$$t_x = s + re_x$$

$$\mathcal{P} \xrightarrow[t_0, t_1, t_{-1}]{e_0, e_1, e_{-1}} \mathcal{V}$$

$$k \stackrel{?}{=} e_0 + e_1 + e_{-1}$$
$$g^{t_0} \stackrel{?}{=} c^{e_0} d_0$$
$$g^{t_1} \stackrel{?}{=} (c/h)^{e_1} d_1$$
$$g^{t_{-1}} \stackrel{?}{=} (ch)^{e_{-1}} d_{-1}$$

Notamos que en t_{x-1}, t_x, t_{x+1} entendemos los subíndices x, $x-1$ y $x+1$ mód 3 y siempre tomando como representantes $\{-1, 0, 1\}$, y por tanto se corresponden con t_{-1}, t_0, t_1 convenientemente reordenados. Con esta notación podemos escribir las ecuaciones de una forma más compacta:

$s, t_{x+1}, t_{x-1}, e_{x+1}, e_{x-1} \xleftarrow{\$} \mathbb{Z}_q$

$$d_y = \begin{cases} g^s & \text{si } y = x \\ g^{t_y} \left(ch^{-y}\right)^{-e_y} & \text{si } y \in \{x-1, x+1\} \end{cases}$$

$$\mathcal{P} \xrightarrow{d_0, d_1, d_{-1}} \mathcal{V}$$

$$k \xleftarrow{\$} \mathbb{Z}_q$$

$$\mathcal{P} \xleftarrow{k} \mathcal{V}$$

$e_x = k - e_{x+1} - e_{x-1}$

$t_x = s + re_x$

$$\mathcal{P} \xrightarrow[t_0, t_1, t_{-1}]{e_0, e_1, e_{-1}} \mathcal{V}$$

$$k \stackrel{?}{=} e_0 + e_1 + e_{-1}$$

$$g^{t_y} \stackrel{?}{=} (ch^{-y})^{e_y} d_y, \qquad \forall y \in \{-1, 0, 1\}$$

La *corrección* es fácil de comprobar, si ambos siguen el protocolo es evidente que $k \stackrel{?}{=} e_0 + e_1 + e_{-1}$ es una igualdad pues $e_x = k - e_{x+1} - e_{x-1}$.

En el resto de ecuaciones separaremos el caso x:

$$\begin{aligned} g^{s+re_x} &= g^{s+re_x} \\ g^{s+re_x} &= (g^r h^x h^{-x})^{e_x} g^s \\ g^{t_x} &= (ch^{-x})^{e_x} d_x \end{aligned}$$

Y los casos $y \in \{x-1, x+1\}$:

$$\begin{aligned} g^{t_y} &= g^{t_y} \\ g^{t_y} &= (ch^{-y})^{e_y} g^{t_y} \left(ch^{-y}\right)^{-e_y} \\ g^{t_y} &= (ch^{-y})^{e_y} d_y \end{aligned}$$

Comprobaremos también la *consistencia*. Dadas dos conversaciones aceptadas:

$$\begin{aligned} &(d_0, d_1, d_{-1}, k, t_0, t_1, t_{-1}, e_0, e_1, e_{-1}) \\ &\left(d_0, d_1, d_{-1}, k', t_0', t_1', t_{-1}', e_0', e_1', e_{-1}'\right) \\ &k \neq k' \end{aligned}$$

Como k es diferente de k' una de las e_y ha de ser también distinta de e'_y.

$$\begin{aligned} & e_{-1} + e_0 + e_1 = k \neq k' = e'_{-1} + e'_0 + e'_1 \\ \implies & \exists y \in \{-1, 0, 1\} \text{ tq } e_y \neq e'_y \\ \implies & (e_y - e'_y) \neq 0 \in \mathbb{Z}_q \end{aligned}$$

Por otra parte como ambas son conversaciones son aceptadas tenemos:

$$\begin{aligned} g^{t_y} &= (ch^{-y})^{e_y} d_y \\ g^{t'_y} &= (ch^{-y})^{e'_y} d_y \end{aligned}$$

Dividiendo ambas:

$$\begin{aligned} g^{t_y - t'_y} &= (ch^{-y})^{e_y - e'_y} \\ g^{(t_y - t'_y)/(e_y - e'_y)} h^y &= c \end{aligned}$$

Finalmente $\big((t_y - t'_y)/(e_y - e'_y), y\big)$ sería una apertura del compromiso c a un valor $y \in \{-1, 0, 1\}$, que es lo que queríamos demostrar conocer.

El protocolo es de conocimiento nulo puesto que existe un simulador que genera conversaciones aceptadas completamente indistinguibles de las conversaciones reales producto de la interacción entre el probador y el verificador.

$$\begin{aligned} t_{-1}, t_0, t_1, e_{-1}, e_0, e_1 &\xleftarrow{\$} \mathbb{Z}_q \\ k &= e_{-1} + e_0 + e_1 \\ d_{-1} &= g^{t_{-1}} (ch)^{-e_{-1}} \\ d_0 &= g^{t_0} c^{-e_0} \\ d_1 &= g^{t_1} (c/h)^{-e_1} \\ & (d_0, d_1, d_{-1}, k, t_0, t_1, t_{-1}, e_0, e_1, e_{-1}) \text{ es una conversación válida} \end{aligned}$$

Observamos que este protocolo no filtra ninguna información sobre cuál de las tres posibles aperturas del compromiso se conoce. En un contexto en el que un adversario utilizara un ordenador cuántico podría emplearlo para encontrar aperturas del compromiso a los tres posibles valores $-1, 0, 1$, pero a partir de las conversaciones no podría encontrar ninguna información sobre cuál de las aperturas es la que conoce el probador. Este adversario podría encontrar aperturas $\alpha_{r_j^{(i)}}, \alpha_{e_{1,j}^{(i)}}, \alpha_{e_{2,j}^{(i)}}$ de los compromisos de $r_j^{(i)}, e_{1,j}^{(i)}, e_{2,j}^{(i)}$ para cada posible valor en $[-\beta + 1, \beta - 1]$. Es aquí donde la propiedad de esconder perfectamente hace que esto no suponga un problema, pues el adversario no puede distinguir cuáles de estas posibles aperturas conoce el probador, y hay exactamente tantas aperturas como posibles valores de los elementos comprometidos.

6. Conclusiones

En este trabajo hemos presentado una prueba de conocimiento nulo para verificar el comportamiento de un nodo mixnet que utilice un cifrado basado en ring-LWE. Gran parte de las operaciones pueden ser precalculadas previamente, en lo que como Wikström llamaríamos fase *offline*, y solo los últimos pasos de la prueba han de ser realizados una vez se ha producido la votación. De esta forma se reduce el tiempo que transcurre desde que se produce la votación hasta que se verifican los resultados.

Queda por estudiar la eficiencia del protocolo con una implementación real, en la que en primer lugar habría que definir los diferentes parámetros del cifrado ring-LWE siguiendo criterios que garanticen la seguridad a largo plazo del cifrado. Sería necesario también especificar las optimizaciones necesarias, como el uso de la Transformada Rápida de Fourier para las multiplicaciones de polinomios y técnicas eficientes de exponenciación para los compromisos.

Si una vez realizada una correcta implementación del cifrado y de la prueba de conocimiento nulo su consumo de recursos es eficiente y resulta ser una propuesta factible para conseguir un sistema de votación electrónica post-cuántico entonces sería necesario complementarla con el resto de pruebas de conocimiento nulo auxiliares que se utilizan en un esquema de voto electrónico completo: pruebas de igualdad de textos planos, pruebas de descifrado, pruebas de validez de los votos cifrados, descifrados parciales compartiendo la clave secreta entre diferentes servidores etc.

En el caso particular de la prueba de igualdad de textos planos es posible adaptar una de las pruebas con compromisos basados en ring-LWE de Benhamouda et. al. [7], que consiguen conocimiento nulo desechando parte de las muestras para que la distribución de las salidas no revele ninguna información, idea propuesta originalmente por Lyubashevsky, coautor del artículo anterior [8]. Teniendo en cuenta que como cada cifrado en nuestro caso utiliza tres polinomios secretos $\mathbf{r}, \mathbf{e}_1, \mathbf{e}_2$ en este caso sería necesario repetir este procedimiento tres veces, por lo que el tiempo esperado de ejecución de la prueba se vería incrementado considerablemente y sería necesario estudiar su viabilidad.

Ha de tenerse en cuenta que por razones de eficiencia esta prueba de conocimiento nulo está basada en un sistema de cifrado post-cuántico pero utiliza unos compromisos que escondiendo perfectamente son solo vinculantes en un escenario pre-cuántico. Sin embargo el esquema de la prueba puede ser adaptado para utilizar otro tipo de compromisos post-cuánticos que tengan las mismas propiedades homomórficas. Se propone utilizar los compromisos de Banhamouda [7], que permiten demostrar que un compromiso es suma o producto de dos compromisos dados. Estos compromisos serían aplicables en un escenario post-cuántico, pues están basados también el problema ring-LWE, sin embargo se descarta su uso por el momento pues cada compromiso pasaría de ser un elemento de un grupo G_q a ser a su vez un polinomio, aumentando el tamaño final y los recursos que consumirían las pruebas.

7. Bibliografía

Para la realización de este trabajo se han consultado habitualmente las tesis *Individual Verifiability in Electronic Voting* de Sandra Guasch Castelló y *Privacidad a largo plazo en sistemas de votación electrónica* de Núria Costa Mirada.

Referencias

[1] P.W. Shor, *Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer*. SIAM J. Comput., 26(5):1484 1509, (1997)

[2] Eduard Sanou, *Post Quantum Cryptography: Lattice-based encryption*, PFM

[3] Oded Regev, *On lattices, learning with errors, random linear codes, and cryptography* J. ACM 56.6 (2009)

[4] Torben Pryds Pedersen *Non-Interactive and Information-Theoretic Secure Verifiable Secret Sharing* CRYPT0 1991, LNCS 576, pp. 129-140, (1992)

[5] Björn Terelius and Douglas Wikström. *Proofs of Restricted Shuffles*, AFRICACRYPT 2010, LNCS 6055, pp. 100–113, (2010)

[6] San Ling, Khoa Nguyen, Damien Stehlé, Huaxiong Wang, *Improved Zero-knowledge Proofs of Knowledge for the ISIS Problem, and Applications*, Public-Key Cryptography – PKC 2013, pp 107-124, (2013)

[7] Fabrice Benhamouda, Stephan Krenn, Vadim Lyubashevsky, and Krzysztof Pietrzak, *Efficient Zero-Knowledge Proofs for Commitments from Learning with Errors over Rings* ESORICS 2015, Part I, LNCS 9326, pp. 305–325, (2015)

[8] Vadim Lyubashevsky, *Lattice Signatures Without Trapdoors* EUROCRYPT 2012, LNCS 7237, pp. 738-755, (2012)