Universidad de Valladolid

Facultad de Ciencias

TRABAJO FIN DE GRADO

Compartición de secretos y votación electrónica: implementación y aplicaciones

Autor: Iván Muñoz Martín

Tutor: Diego Ruano Benito

Agradecimientos

En primera instancia me gustaría dar las gracias a José Ignacio Farrán y a Diego Ruano por haber tutorizado mi Trabajo de Fin de Grado. Les agradezco principalmente el haberme transmitido los conocimientos que he podido adquirir durante el desarrollo del trabajo, así como su disposición a lo largo del final de esta etapa. También me gustaría mencionar la importancia del resto del profesorado de la Universidad de Valladolid, gracias a los cuales he podido estudiar y posteriormente profundizar en estas materias. En definitiva, gracias a todo el equipo de profesores que me han acompañado durante el grado, por haberme ayudado y apoyado a formarme en lo que será mi futuro profesional.

Resumen

En muchos sectores y ámbitos profesionales existe cierta información confidencial que no conviene exponer. Dado que se trata de un tipo de información cuyo acceso podría tener numerosas consecuencias, es necesario utilizar algún método que oculte dicho contenido. En este sentido, la criptografía se encarga de proteger estos datos al convertirlos en secuencias que resulten ilegibles para cualquier elemento que no tenga implicación directa en ellos.

El presente documento se centrará en los esquemas de compartición de secretos (ECS) y la computación segura entre múltiples partes (MPC). Hemos realizado un estudio de los diferentes tipos de ECS, dando sentido a todos los conceptos vinculados a ellos.
El análisis de los ECS se ha orientado a los esquemas de umbral, con su ejemplo más característico el esquema de Shamir. Además, se ha estudiado la utilización de los códigos lineales para formar un esquema de compartición de secretos a través de la idea de Massey. La idea de Massey se ha extendido para poder trabajar con ficheros más grandes.

El estudio de los ECS, ha tenido como finalidad llegar a conseguir un aplicativo de votación electrónica, gracias a la unión con el protocolo de computación segura entre múltiples partes, MPC.

Para finalizar hemos explicado todos los conceptos correspondientes para el entendimiento de la votación electrónica, en el que todo el mundo colabora y obtenemos como resultado final la suma de todos los votos de los participantes. Siempre se ha intentado aportar ejemplos numéricos para ilustrar a la perfección los algoritmos que lo componen.

Abstract

In many industries and professional fields there is certain confidential information that should not be exposed. Since this type of information could have many consequences if accessed, it is necessary to use some method to conceal its content. Cryptography protects this data by converting it into sequences that are unreadable to anyone who has no direct involvement with it.

This paper will focus on (ECS) secret sharing schemes and (MPC) multiparty secure computing. We have conducted a survey of the different types of ECS, making sense of all the concepts linked to them.
The analysis of ECS has been oriented towards threshold schemes, with its most characteristic example being Shamir's scheme. In addition, the use of linear codes to form

a secret sharing scheme has been studied through Massey's idea. Massey's idea has been extended to work with larger files.

The study of of the *ECS* was to achieve a electronic voting application, thanks to the union with the secure multi-party computing protocol, *MPC*.

Finally, we have explained all the corresponding concepts for the understanding of the electronic voting, in which everybody collaborates and we obtain as final result the sum of all the votes of the participants. We have always tried to provide numerical examples to understand perfectly the algorithms that compose it.

Índice general

Capítulo 1

Introducción

En todo protocolo criptográfico el objetivo principal es asegurar la privacidad de la información, se pretende plasmar la idea de que existe una serie de herramientas criptográficas que se encargan de dar consistencia a los mensajes. Entre estos algoritmos criptográficos se encuentran los esquemas de compartición de secretos (ECS). Los ECS son una técnica criptográfica que consiste en dividir en particiones un secreto y repartir estas particiones entre diversos participantes. La división del mensaje implica que la reconstrucción sea más factible, con el conocimiento del mayor número de particiones en las que se dividió el secreto.

Este tipo de esquemas sirven como almacén seguro de información. El secreto se encuentra distribuido en diferentes lugares, lo cual garantiza una mayor protección a la hora de recibir posibles ataques de agentes externos. Además, desde esta perspectiva, no sería suficiente con simplemente dividir la información, sino que se debe ofrecer garantías de protección de acceso a estos datos desde cualquier tipo de medio.

Para comprender mejor los esquemas de compartición de secretos, podemos entenderlos como una especie de rompecabezas. Las piezas del rompecabezas se reparten entre los distintos jugadores con el objetivo de garantizar la seguridad de la solución. Así, varios jugadores podrían tener piezas que por separado carecen de sentido, pero al juntarse se llegará a construir la figura en su plenitud.

La idea general es sencilla, vamos a suponer que nosotros queremos guardar el código de seguridad de nuestra tarjeta de crédito (compuesto por cuatro números). Si nosotros facilitamos cada dígito a cuatro amigos diferentes, entonces una persona ajena a esta información tendrá unas 10^4 combinaciones posibles para descifrarlo. La seguridad que hemos conseguido para que no se pueda detectar el número secreto sería buena, pero si se juntan los tres amigos con sus respectivos dígitos confidenciales, la probabilidad que tienen de adivinar dicho número sería mayor, pues solo existirían 10 posibilidades para llegar a saber el código de seguridad de dicha tarjeta. Este ejem-

plo en un caso sencillo en el que solo se divide el secreto en cuatro unidades, pero es suficiente para observar cómo se complica la obtención del mensaje dependiendo de la información de la que se disponga.

Los precursores de los esquemas de compartición de secretos fueron Blackley y Shamir en el año 1979. Ambos trabajaron de manera independiente; Shamir basó su algoritmo en los polinomios interpoladores y Blacley en esquemas vectoriales. A pesar de enfocar sus algoritmos de distintas maneras, los dos procedían igual en el siguiente aspecto: si tenemos un número de partes n del secreto S, fijando un número t, el cual corresponde con el mínimo número de particiones a partir del cual se podría acceder a la información. A este tipo de esquemas se les denominó esquemas $(t, n)-$umbrales, en los que si tenemos un número menor o igual de t particiones no sabemos nada del secreto. Sin embargo, si el número de particiones que tenemos en nuestro poder es mayor a t se tiene pleno conocimiento sobre el secreto.
Para llevar a cabo esto se define el número t, y se generan aleatoriamente los coeficientes necesarios para construir un polinomio de grado t. El método para recuperar el secreto S, es reconstruir el polinomio $f(x)$ a partir de las particiones, esto se hace mediante la interpolación de Lagrange.

El esquema de Shamir presenta como principal desventaja que para la realización de este esquema de compartición de secretos, es necesario utilizar un cuerpo finito con mayor tamaño que el número de participantes. Esto quiere decir que los cuerpos finitos pequeños son desechables en muchas implementaciones. Para resolver este problema utilizaremos los códigos lineales para crear esquemas de compartición de secretos, gracias al método de Massey, el cual permite usar un cuerpo finito arbitrario. Pero a su vez el método Massey no es lo suficiente efectivo en la transmisión de bits, por eso hemos tratado de extender este método y analizar los denominados esquemas en rampa cuyas particiones conjuntas pueden dar alguna información parcial del secreto, aumentando la tasa de información. Mientras que en el método de Massey para compartir n bits es necesario enviar n bits, en los esquemas en rampa para distribuir n bits es suficiente con mandar uno solo.

Los esquemas de compartición de secretos tienen diversas aplicaciones en el mundo real. Entre alguna de estas aplicaciones se encuentra: el control de accesos, la apertura de cajas de seguridad, la inicialización de dispositivos militares, y procesos electrónicos como la subasta y la votación electrónica. La finalidad de nuestro estudio es explicar como uniendo los ECS con la computación segura entre múltiples partes MPC, podremos dotar a nuestros protocolos de transparencia, solidez y seguridad.

La unión de los protocolos MPC y ECS la hemos plasmado en la sección final con el protocolo de votación electrónica. Pretenderemos destacar las ventajas en el protocolo de votación electrónica frente a los sistemas tradicionales que todos conocemos. Se

intentará asegurar la fiabilidad y eficiencia, de este protocolo frente a la corrupción pasiva, es decir, aquella situación en que los participantes corruptos siguen el protocolo pero comparten la información.

1.1. Motivación

Este trabajo consta de una descripción teórica de todos los conceptos matemáticos y criptográficos. Se presentará los *ECS* y el protocolo *MPC*, con el objetivo de ofrecer una mejor comprensión sobre cómo poder dotar de seguridad y al mismo tiempo, ofrecer una serie de pautas eficientes para la utilidad de estas técnicas en la vida real.

El documento presenta una descripción de todos los conceptos matemáticos como criptográficos. Intentaremos centrarnos en la teoría que existe detrás de la compartición de secretos *ECS* y del protocolo *MPC*. Siempre hemos tenido como finalidad crear una aplicación capaz de garantizar la seguridad de la información, como la votación electrónica.

Para entender todos los conceptos desarrollados en la votación electrónica es necesario comprender previamente los esquemas de compartición de secretos y el protocolo *MPC*. La votación electrónica se ha convertido en la actualidad en un recurso muy utilizado, cada vez son más las organizaciones que están buscando soluciones para estos protocolos. La necesidad de eliminar desplazamientos y dotar de sencillez al proceso, implica que importantes empresas han conseguido lanzar prototipos totalmente seguros de votación electrónica.

Es fundamental, por sus beneficios, un conocimiento integral tanto de la justificación como del proceso e implementación de los protocolos, aunque esto haya supuesto un mayor tiempo en el desarrollo del trabajo de investigación. Se pretende combinar los conocimientos adquiridos durante el grado con la investigación teórica llevada a cabo durante la revisión bibliográfica. En este sentido se tratará de llevar a cabo los objetivos que se proponen a continuación.

1.2. Objetivos

Ahora mostramos una serie de objetivos que se pretenden alcanzar tras la realización del presente trabajo fin de grado.

1. En un primer momento, se ofrece un estudio sobre la Teoría de códigos correctores, que tendrá como objetivo introducirse en el mundo de la codificación y que servirá para explicar los protocolos posteriormente.

2. Realizar un seguimiento sobre distintos resultados de los ECS. Estas explicaciones tendrán como objetivo principal la ejemplificación y las justificaciones matemáticas de todos los conceptos para entender la importancia de los esquemas de compartición de secretos como medida de seguridad de la información.

3. Como ya se expuso, siempre que existe información confidencial, esta puede ser atacada por adversarios ajenos. Por este motivo, es fundamental conocer las diversas formas de ataques y los distintos tipos de atacantes que existen para poder anticiparse a ellos. Con este objetivo, hablamos de la creación de protocolos fiables a través de la computación segura entre múltiples partes, (MPC). Esta sección nos servirá para explicar como agentes externos pueden tener el control de partes para intentar entorpecer el protocolo, y como con el uso del protocolo MPC diseñaremos posteriormente protocolos totalmente seguros.

4. Aprender los conceptos del protocolo de votación electrónica. Este punto se encuentra dividido en tres partes diferenciadas, para entender y justificar el protocolo de la mejor manera. En la primera parte se tiene como objetivo explicar de manera general el protocolo. Posteriormente, en la segunda parte emplearemos los esquemas ECS y el protocolo MPC para explicar un ejemplo de votación electrónica seguro frente a la corrupción pasiva. Por último se ejemplifica numéricamente el protocolo que se ha desarrollado previamente.

Capítulo 2

Teoría de los códigos correctores de errores

2.1. Introducción

En esta sección, empezaremos definiendo el código y sus parámetros fundamentales. Además, explicaremos otros conceptos que nos servirán para posteriores capítulos del documento. Por último, al final del capítulo definiremos la entropía, concepto básico de la teoría de la información. Para la descripción de todo esto, hemos utilizado las siguientes referencias: [2], [9], [13], [22] y [24].

Los códigos lineales son códigos que poseen una estructura algebraica, en ellos cualquier combinación lineal de palabras es también una palabra del código. Estos códigos permiten que la codificación y decodificación sea menos costosa computacionalmente. Los códigos que emplearemos a partir de este momento serán lineales.

Definición 2.1. Un código lineal C de dimensión k, es un subespacio vectorial de $\mathbb{F}_q^n$. Los códigos lineales se suelen denotar con los llamados parámetros fundamentales, $[n, k]$, n indica la longitud de los elementos del subespacio vectorial y k indica la dimension del subespacio vectorial, es decir, del código lineal.

Nota 2.2. Nosotros consideraremos códigos de bloque. Los códigos de bloque permiten que las palabras del código, que denotaremos por c, tengan todas la misma longitud y hagan referencia al cuerpo finito $\mathbb{F}_q$, de q símbolos con el que operamos. La cantidad de palabras de un código viene dada por q^k.

Nota 2.3. El soporte de un vector $a = (a_1, \cdots, a_n) \in \mathbb{F}_n^q$ esta definido por $sop(c) = \{i \mid a_i \neq 0,\ 1 \leq i \leq n\}$.
Una palabra c cubre a una palabra c', si el soporte de c contiene al de c'.
Una palabra c es llamada mínima si solo cubre a múltiplos distintos de cero.

Definición 2.4. Llamaremos matriz generatriz G de un código lineal C, a una matriz $k \times n$ cuyas filas forman una base de C, es decir, estas filas forman una base del código lineal como espacio vectorial.
La matriz generatriz G se puede relacionar con una aplicación lineal biyectiva:

$$f : \mathbb{F}_q^k \longrightarrow C \subset \mathbb{F}_q^n.$$

Nota 2.5. Las bases de un subespacio vectorial no son únicas, entonces las matrices generatrices no son únicas. Siempre existirá una matriz invertible con la que podremos llegar a partir de una matriz generatriz a otra. Además, distintas matrices generatrices producirán distintas codificaciones, ya que $C = \{aG \mid a \in \mathbb{F}_q^k\}$.

Definición 2.6. Un código lineal C es un código sistemático si y solo si su matriz generatriz G es de la forma $G = (I_k|A)$, donde I_k es la matriz identidad de tamaño $k \times k$. Sea una información $m \in \mathbb{F}_q^k$ y la codificamos $mG = m(I_k|A) = (m|mA) \in \mathbb{F}_q^n$, donde mA es la redundancia.

Nota 2.7. Las palabras de un código lineal C $[n, k]$ sistemático tienen k símbolos de información y la redundancia es $r = n - k$ símbolos. Los símbolos redundantes permiten validar que los k símbolos de información obtenidos son correctos y no contienen errores.
Para un n fijo nuestro objetivo será buscar que la dimensión k sea alta, ya que esto producirá baja redundancia, lo que mejorará la capacidad de transmisión del código.

Definición 2.8. Dado un código lineal C $[n, k]$, llamamos matriz de control H, a una matriz $(n - k) \times n$ que cumple:

$$c \in C \Leftrightarrow Hc^t = 0, \forall c \in \mathbb{F}_q^n.$$

En consecuencia de esta definición, si tenemos un código lineal C, con una matriz generatriz G y H como matriz de control entonces:

$$GH^t = HG^t = 0.$$

Al principio de la sección hablamos de los parámetros fundamentales en el cual incluíamos n y k. A continuación, añadiremos como parámetro la distancia mínima d, por lo tanto, los códigos a partir de ahora serán de tipo $[n, k, d]$. Este nuevo parámetro, d, nos servirá para calcular la cantidad de errores que los códigos lineales son capaces de corregir, pero para hablar de este concepto es necesario introducir los conceptos de peso y distancia Hamming.

Definición 2.9. Dado un vector $x = (x_1, \ldots, x_n) \in \mathbb{F}_q^n$, llamaremos peso de Hamming al número de posiciones que son distintas de cero, $\omega(x)$, de tal manera que:

$$\omega(x) = \#\{i \mid x_i \neq 0,\ 1 \leq i \leq n\}.$$

Nota 2.10. El peso mínimo de un código lineal C es el mínimo de los pesos de Hamming de las palabras del código no nulas, es decir, $\omega(C) = min\{\omega(x) \mid x \in C \setminus \{0\}\}$.

Definición 2.11. La distancia de Hamming que existe entre dos vectores $x, y \in \mathbb{F}_q^n$, es denotada por $d_H(x, y)$, y es el número de posiciones en las que x_i difiere de y_i:

$$d(x, y) = \#\{i \,|x_i \neq y_i,\, 1 \leq i \leq n\} = \omega(x - y).$$

Definición 2.12. La distancia mínima de un código lineal C es el mínimo de las distancias de Hamming de las palabras del código dos a dos:

$$d(C) = min\{d(x, y) \mid x,\ \ y \in C,\ \ x \neq y\}.$$

Proposición 2.13. *Sea un código lineal C $[n, k, d]$, entonces el peso mínimo del código C es igual a la distancia mínima de C.*

Demostración. Sea $c \in C$ una palabra de peso mínimo, tenemos por definición $\omega(c) = d(0, c)$, $0 \in C$ por ser un subespacio vectorial. Entonces obtenemos que $d(C) \leq \omega(C)$.

Por otra parte, si cogemos dos palabras del código de distancia mínima $a, b \in C$. Entonces $d(a, b) = d(a - b, 0) = d(0, a - b) = \omega(a - b)$ y $a - b \in C$, dado que C es un subespacio vectorial. Por lo tanto, $\omega(C) \leq d(C)$ y llegamos a la igualdad $\omega(C) = d(C)$. □

Nota 2.14. La explicación de la existencia de la porposición 2.13, viene dada por la linealidad del código, ya que la distancia entre dos palabras hemos visto que es igual al peso de $x - y$. Además, al ser lineal el mínimo de los pesos es igual al mínimo de las distancias.

La definición de distancia mínima 2.12, nos indica que tenemos que comparar cada par de palabras del código o el peso de cada palabra del código. Esto es complejo cuando tengamos códigos con una gran cantidad de palabras. La matriz de control del código, nos facilita este trabajo como observaremos en la siguiente proposición:

Proposición 2.15. *Sea un código lineal C de tipo $[n, k, d]$ denotemos a H como su matriz de control. Dado un número entero positivo r, tenemos que $d > r$ si y sólo si cualquier combinación de r columnas de la matriz de control H son linealmente independientes.*

Demostración. $\Rightarrow$) Demostraremos la contraposición: Si la matriz de control H del código C tiene r columnas linealmente dependientes, entonces existe un vector $c \in C$ que proporcionan dicha combinación lineal. Esto implica que $Hc^t = 0$, es decir, $c \in C$ y como $\omega(c) \leq r$, entonces $d \leq r$.

$\Leftarrow$) Si la matriz de control H del código C tiene r columnas linealmente independientes, entonces no existe ningún vector c que proporcionan dicha combinación lineal, lo que implica que $Hc^t \neq 0$ y $c \notin C$. Debido a esto, si $c \in C$, necesariamente $\omega(c) > r$.

□

Corolario 2.16. *Sea un código lineal C que tiene a H como una matriz de control, entonces si $d(C)$ es la distancia mínima del código C, esta sería igual al mínimo numero de columnas linealmente dependientes de H.*

Nota 2.17. Todos los conceptos vistos previamente de peso mínimo y distancia mínima nos permitirán detectar los errores que contiene el código. Ahora llamaremos e al vector error que tiene la misma longitud que las palabras del código. Si enviamos una palabra $c \in C$ recibiremos $x = c + e$, donde e es el vector error, de tal manera que $e = x - c$.

Es posible que en ocasiones no nos percatemos del error producido, por eso es esencial preguntarnos cuantos errores somos capaces de corregir:

Definición 2.18. Si tenemos dos palabras c, c' de un código lineal C, con vectores de errores e y e', de tal manera que sus pesos $\omega(e), \omega(e') \leq t$, diremos que es corrector de t errores si $c + e \neq c' + e'$, para todo c y $c' \in C$.

Teorema 2.19. *Un código lineal C $[n, k, d]$ es corrector de t errores $\Leftrightarrow t < \frac{d}{2}$, es decir, $t = \left\lfloor \frac{d-1}{2} \right\rfloor$.*

Demostración. Para realizar esta prueba realizaremos dos reducciones al absurdo:

$\Rightarrow$) Si suponemos que $t \geq \frac{d}{2}$, entonces sea una palabra $c \in C$ de manera que su peso sea igual a la distancia d, es decir, $\omega(c) = d$. Si tomamos $\left\lfloor \frac{d-1}{2} + 1 \right\rfloor$ posiciones no nulas y las cambiamos a nulas obtenemos x, tal que:

$$\omega(x) = d(0, x) \leq d - \left\lfloor \frac{(d-1)}{2} + 1 \right\rfloor = \left\lfloor \frac{(d-1)}{2} \right\rfloor < \frac{d}{2} \leq t,$$

además $\omega(x - c) = d(c, x) \leq \frac{d}{2} \leq t$.
Entonces como nuestro código C es corrector de t errores, llegamos a la contradicción porque 0 y c pertenecen a C $(0 + x = c + (x - c))$. Esto no puede ocurrir porque para dos palabras distintas de C a las que introducimos errores, con peso $\omega(e) < t$, estas palabras deberían seguir siendo diferentes.

$\Leftarrow$) Ahora suponemos $t < \frac{d}{2}$ y el código C no es corrector de t errores. Si cogemos dos palabras de código, c, c', con errores respectivos e, e' y $\omega(e) < t$, $\omega(e') < t$, tal que $c + e = c' + e'$.
Obtenemos que $c - c' = e - e'$ y esto produce que $\omega(c - c') = \omega(e' - e) \leq 2t < d$. Aquí llegamos a la contradicción, dado que $a = c - c'$ sería otra palabra del código C cuyo peso es inferior al peso mínimo del código y esto no puede ocurrir.

□

Nosotros siempre trataremos, que para un n fijo, la dimensión k y la distancia d sean lo más altas posibles. Si el parámetro k es muy alto, esto provocará que nuestro código tenga poca redundancia. Sí la d es grande, esto nos permitirá corregir una gran cantidad de errores. Esta idea tiene una serie de restricciones una de ellas, la cota superior que nos proporcionaría la proposición 2.15, conocida como cota de Singleton:

Corolario 2.20. ***(Cota de Singleton)*** *Si tenemos un código lineal C de tipo $[n, k.d]$, se cumple:*

$$d \leq n - k + 1.$$

Demostración. Sea un código C, con matriz de control H. Como el rango de H es $n - k$, por lo tanto, $n - k + 1$ columnas de H son linealmente dependientes. Entonces por la proposición 2.15 llegamos a la conclusión de que $d \leq n - k + 1$. □

Nota 2.21. Una vez introducido los conceptos anteriores de distancia podemos corregir errores en el código, si dado un $x \in C$ $Hx^{\perp} \neq 0$, ya que como hemos descrito anteriormente $x \in C \Leftrightarrow Hx^t = 0$.

2.1.1. Códigos MDS

Llamamos *MDS*, o códigos de máxima distancia separable, a los códigos que cumplen la igualdad de la Cota de Singleton:

$$d = n - k + 1.$$

Estos códigos son muy interesantes ya que nos dan el mayor valor del par (k, d) posible para dicha cota, es decir, son códigos que tienen la capacidad de codificar un mayor número de errores.
Los códigos Reed-Solomon son los códigos paradigmáticos dentro de los códigos *MDS*. En la actualidad, los códigos Reed-Solomon son códigos lineales que se utilizan para corregir errores en varios sistemas incluyendo los dispositivos de almacenamiento (DVD, código de barras, discos compactos, etc), así como también en comunicaciones inalámbricas, móviles o satélites. Definiremos este tipo de códigos y posteriormente veremos las relación que tienen con los *ECS*.

Definición 2.22. Dado un vector $x = (x_1, \ldots, x_n) \in \mathbb{F}_q^n$, con $x_i \neq x_j \, \forall i \neq j$, y un número entero n tal que $0 < n \leq q$. Denotamos por $RS_k(x)$ al código lineal Reed-Solomon asociado al vector x, formado por los vectores obtenidos al evaluar los polinomios de L_k en las coordenadas del vector x:

$$RS_k(x) = \{(f(x_1), \ldots, f(x_n)) \mid f \in L_k\},$$

donde L_k es el espacio vectorial formado por los polinomios $\mathbb{F}_q[x]$ de grado menor que k, donde k es un entero con $0 < k \leq n$.

Es evidente que la dimensión de L_k es k, debido a que cualquier polinomio de grado menor que k se puede generar como combinación lineal de los elementos $\{1, x, \ldots, x^{k-1}\}$ sobre el cuerpo $\mathbb{F}_q$. Como estos elementos son linealmente independientes, forman una base de L_k.
La aplicación ψ:

$$\begin{aligned} \psi : \; L_k &\longrightarrow \; \mathbb{F}_q^n \\ f &\longmapsto \; (f(x_1), \ldots f(x_n)) \end{aligned}$$

es lineal e inyectiva, ya que $f \in Ker \Rightarrow f(x_i) = 0 \;\; \forall i \Rightarrow f = 0$ y el $deg(f) < k \leq n$.

$RS_k(x)$ es un código lineal, dado que si tomamos dos palabras c_1 y $c_2 \in \mathbb{F}_q^n$, que se codifican de la siguiente manera $c_1 = (f(x_1), \ldots, f(x_n))$ y $c_2 = (g(x_1), \ldots, g(x_n))$.
Sean $b, b' \in \mathbb{F}_q$ y $h(x) = bf(x) + b'g(x)$, de tal manera que $h(x) \in L_k$ y tenemos que $b'c_1 + bc_2 = (g(x_1), \ldots, g(x_n))$

Proposición 2.23. *Un código Reed-Solomon que tiene como parámetros $[n, k, n+1-k]$ sobre $\mathbb{F}_q$, con $n \leq q$ y es por tanto un código MDS.*

Demostración. Una vez probado que ψ es inyectiva y de dimension k entonces f es un polinomio no nulo cuyo grado es menor que k, entonces este polinomio no puede tener más de $k-1$ raíces.Si tomamos la definición de distancia mínima $d(C) = min\{\omega(c) \mid c \in C \setminus \{0\}\}$, el peso será al menos n menos el número de ceros. Por lo tanto, $d(C) \geq n - (k-1)$.

Por la cota de Singleton $d \leq n-k+1$ llegamos a la igualdad $d = n-k+1$, entonces el código $RS_k(a)$ sería un código MDS.

□

Al ser la aplicación ψ lineal e inyectiva, si tomamos como base de L_k, $\{1, X, \ldots, X^{k-1}\}$, entonces $\{\psi(1), \ldots, \psi(X^{k-1})\}$ también será base del código $RS_k(x)$. Por tanto, si se evalúan los polinomios de la base $\{1, X, \ldots, X^{k-1}\}$ en las posiciones del vector $x = (x_1, \ldots, x_n)$.
La matriz generatriz de un código Reed-Solomon $G \in \mathbb{F}_q^{k \times n}$ es:

$$G = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ x_1 & x_2 & \cdots & x_n \\ \vdots & \vdots & & \vdots \\ x_1^{k-1} & x_2^{k-1} & \cdots & x_n^{k-1} \end{pmatrix}$$

Cada una de las filas de la matriz G corresponde a la imagen $\psi(1), \psi(x), \ldots, \psi(x^{k-1})$.

Definición 2.24. Sea α un elemento primitivo de $\mathbb{F}_q$ y sea $x = (\alpha^0, \alpha^1, \ldots, \alpha^n)$ el vector de puntos de evaluación. La matriz de tipo Vandermonde asociada sino evaluamos en cero es:

$$V(\alpha, k) = \begin{pmatrix} 1 & \alpha^0 & \cdots & \alpha^{n-1} \\ 1 & \alpha^1 & \cdots & \alpha^{n-1} \\ \vdots & \vdots & & \vdots \\ 1 & \alpha^{k-1} & \cdots & \alpha^{(k-1)(n-1)} \end{pmatrix}$$

2.2. Decodificación

Ahora hablaremos de una manera de decodificación de errores, que tiene en cuenta tanto la estructura algebraica como las propiedades de grupo que tienen los códigos lineales.

Definición 2.25. Dado un código lineal C de tipo $[n, k]$, cuya matriz de control es H. Si tenemos el vector $r \in \mathbb{F}_q^n$ entonces el síndrome asociado a r es $s(r) = Hr^t$, por tanto, $s(r) \in \mathbb{F}_q^{n-k}$. Además $s(r) = Hr^t = 0$ si y solo si $r \in C$.

También podemos definir el síndrome como una aplicación lineal:

$$s : \mathbb{F}_q^n \longrightarrow \mathbb{F}_q^{n-k}$$

$$r \longmapsto Hr^t$$

Por eso si enviamos una palabra $c \in C$ y recibimos $r = c + e$, donde e es el vector error. Se tiene que r tendrá el mismo síndrome que el vector e, ya que $s(r) = Hr^t = s(c + e) = H(c + e)^t = Hc^t + He^t = He^t = s(e)$.

Definición 2.26. Dado un código lineal C de tipo $[n, k]$ y la clase de equivalencia $\mathbb{F}_q/C$, dada por $a \backsim b \Leftrightarrow a - b \in C$. Si tenemos un elemento $b \in \mathbb{F}_q^n$, llamamos cogrupo de b al conjunto $b + C = \{b + c \mid c \in C\}$. Las clases de equivalencia de $\mathbb{F}_q/C$ forman una partición del código C.

Proposición 2.27. *Dos elementos de $\mathbb{F}_q^n$ tienen el mismo síndrome si y solo si están en el mismo cogrupo.*

Demostración. $\Rightarrow$) Sean $x, y \in C$ con el mismo síndrome, esto implica que $Hx^t = Hy^t$ y entonces $H(x-y)^t = 0$, con $x - y \in C$. Por último, como $y = x + (y - x)$ y $x = x + 0$ llegamos a la conclusión que x, y están en el mismo cogrupo.

$\Leftarrow$) Sean $x, y \in C$ pertenecientes al mismo cogrupo $(b + C)$, esto implica por la definición de clase de equivalencia que $x = a + c$ y $y = a + c'$ con c' y $c \in C$. Además, x, y tendrán el mismo síndrome ya que $s(x) = Hx^t = H(a+c)^t = Ha^t = H(a+c')^t = s(y)$.

□

Definición 2.28. Llamaremos líder de la clase de un cogrupo al elemento de peso mínimo dentro del subconjunto. Algunos cogrupos pueden no poseer líder, si existen varios elemento de peso mínimo.

Con estas definiciones llegamos a una decodificación en la que si nosotros enviamos la palabra c y se recibe $r = c + e$, siendo $e \in \mathbb{F}_q$ el vector error. Entonces si $e_i = 0$ todo se encontraría correcto, pero si $e_i \neq 0$ existirían errores y $\omega(e)$ =número de errores.
Tanto r como e pertenecen al mismo cogrupo, por tanto, tienen el mismo síndrome. Entonces calculamos el síndrome de r y en el caso de que tuviera líder podríamos realizar la decodificación, ya que esta es posible restando a r el líder de la clase. Esta resta se puede realizar debido a que $c \in C$ y tiene síndrome 0, $Hc^t = 0$. Entonces el síndrome de r y e es el mismo, y el vector e es de la misma clase de equivalencia que r.

Proposición 2.29. *Cada cogrupo tiene como máximo un elemento de peso menor o igual que $t = \left\lfloor \frac{d-1}{2} \right\rfloor$.*

Demostración. Por reducción al absurdo, si tomamos un x, y distintos y no nulos, pertenecientes al mismo cogrupo, con $\omega(x), \omega(y) \leq t$. Entonces sabemos que $x - y \in C$ y como $\omega(x - y) \leq 2t < d$, es decir, tendríamos una palabra del código no nula con peso inferior a la distancia mínima del código, lo cual es absurdo. □

Nota 2.30. Siempre existe un líder y es único para todos los errores de peso menor o igual que $\left\lfloor \frac{d-1}{2} \right\rfloor$, pero para errores de peso mayor que $\left\lfloor \frac{d-1}{2} \right\rfloor$ puede haber varios de peso mínimo y podríamos asumir distintas opciones:

La primera opción sería decir que no podemos realizar la decodificación, otro camino que podemos tomar sería realizar la decodificación con una de las palabras de peso mínimo de la clase. Nosotros optaremos por la opción de no realizar la decodificación en este caso.

Proposición 2.31. *Dado un código lineal C de tipo $[n, k, d]$, corregirá t errores si y solo si todas las palabras de peso menor o igual que t son lideres de cogrupos.*

Demostración. $\Rightarrow$) Supongamos que C corrige t errores, entonces $d = w_{min} > 2t$. Si tomamos dos palabras c_1 y c_2 del mismo cogrupo y de peso menor o igual que t. Entonces su diferencia esta en C y tiene un peso como máximo de $2t$, contradicción.

$\Leftarrow$) Vamos a suponer que todas las palabras de peso menor o igual que t son líderes del cogrupo. Si tenemos las palabras c_1 y c_2 codificadas con una distancia, $d \leq 2t$. Entonces existen e_1 y e_2 tales que $c_1 + e_1 = c_2 + e_2$ con $w(e_1) \leq t$ y $w(e_2) \leq t$. Esto significa que $s(e_1) = s(e_2)$ y llegamos a la contradicción, dado que e_1 y e_2 están en diferentes cogrupos.

□

Ejemplo 2.1. Gracias a todos los conceptos vistos hasta el momento, daremos un ejemplo de un código lineal C de tipo $[6, 3, 4]$ sobre $\mathbb{F}_2$.
$\mathbb{F}_2$ es un cuerpo finito formado únicamente por los elementos 0 y 1, por ello, también se le llama cuerpo binario.

Sea G la matriz generatriz sistemática del código C:

$$G = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{pmatrix}$$

La aplicación lineal asociada a esta matriz generatriz viene definida de la siguiente forma:

$$f : \mathbb{F}_2^3 \longrightarrow \mathbb{F}_2^6$$
$$a \longmapsto Ha^t$$
$$a = (a_1, a_2, a_3) \longmapsto c = (a_1, a_2, a_3, a_1 + a_2 + a_3, a_1 + a_3, a_1 + a_2)$$

Esto es debido a que cualquier palabra del código, se forma multiplicando el bloque de mensaje por la matriz generatriz:

$$aG = c \in C.$$

La matriz de control H del código C es:

$$H = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}$$

La cantidad de palabras del código C es $2^3 = 8$ y serán vectores de longitud 6.
El código C esta constituido por:

$$C = \{(0,0,0,0,0,0), (1,0,0,1,1,1), (0,1,0,1,0,1), (1,1,0,0,1,0),$$

$$(0,0,1,1,1,0), (1,0,1,0,0,1), (0,1,1,0,1,1), (1,1,1,1,0,0)\}$$

El síndrome pertenece a el cuerpo finito $\mathbb{F}_2^3$. Entonces existen un total de $2^3 = 8$ síndromes.
Ahora mostraremos una tabla con cada líder para cada posible síndrome de este código C.

Síndrome	Líder
$(0,0,0)$	$(0,0,0,0,0,0)$
$(1,1,1)$	$(1,0,0,0,0,0)$
$(1,0,1)$	$(0,1,0,0,0,0)$
$(1,1,0)$	$(0,0,1,0,0,0)$
$(1,0,0)$	$(0,0,0,1,0,0)$
$(0,1,0)$	$(0,0,0,0,1,0)$
$(0,0,1)$	$(0,0,0,0,0,1)$
$(0,1,1)$	No tiene líder

Tabla 2.1: Lideres y sindromes.

En la tabla hemos observado que para el síndrome $(0,1,1)$ no hemos obtenido líder, ya que no existe un único elemento de peso mínimo dentro del subconjunto, dado que tienen el mismo peso tanto el $(0,0,0,0,1,1)$ como el $(1,0,0,0,1,1)$.

Si introducimos ahora un error en una de las palabras del código C, podemos corregir dicho error debido a lo visto en la proposición 2.31, ya que el código tiene una capacidad correctora de $t = 1$. Entonces si tomamos una palabra $c = (1,0,1,0,0,1)$ del código C y añadiendo un error en la tercera coordenada obtenemos $r = (1,0,0,0,0,1)$. El síndrome es $s(r) = Hr^t = (1,1,0)$ y si miramos en la tabla el líder de ese síndrome es $(0,0,1,0,0,0)$, que coincide con el error introducido.

Esta decodificación es sencilla, ya que simplemente habría que buscar tanto el síndrome como el líder de la clase. Pero tiene inconvenientes, dado que si el código tiene una gran dimensión puede presentar problemas de almacenamiento y velocidad de transmisión. Se tendrían que almacenar q^{n-k} síndromes y lideres.

2.3. Código dual

Sea un código lineal C $[n,k]$ sobre $\mathbb{F}_q$, con matriz de control H de rango máximo. Esta matriz puede coincidir al tener rango máximo con la matriz generatriz G de otro código lineal C' sobre $\mathbb{F}_q$. Como C tiene dimension k, C' tiene dimension $n-k$ y una matriz de control de C' coincidirá con la matriz generatriz G de C, ya que $GH^T = 0$.

Definición 2.32. El código dual del código lineal C es el código C', y lo denotaremos por $C^\perp = \{x \in \mathbb{F}_q^n \mid x \cdot c = 0 \ \forall c \in C\}$, donde ' $\cdot$ ' denota el producto escalar usual.

Corolario 2.33. *Dado un código lineal C, se cumple que el dual del código dual coincide con el código lineal de partida, es decir, $(C^\perp)^\perp = C$.*

Proposición 2.34. [24]. *Si el código lineal C es un código MDS, entonces el código lineal $C^\perp$ es también un código MDS.*

Demostración. Sea C un código lineal *MDS* cuya matriz generatriz es G. Al ser un código MDS si tomamos k columnas, estas son linealmente independientes en la matriz G. Además, la matriz generatriz de C coincide con la matriz de control de $C^\perp$, entonces se cumple que $d > k$. Además, como la dimensión del código $C^\perp$ es $n-k$, por la cota de Singleton $d \leq n-(n-k)+1$ y $d \leq k+1$. Luego hemos llegado a la igualdad y $C^\perp$ es un código *MDS*.

□

Definición 2.35. Un código lineal C es auto ortogonal $C \subseteq C^\perp$ y auto-dual si $C = C^\perp$.

Nota 2.36. Una condición necesaria para que un código C sea auto-dual es que su longitud sea par ya que $k = n-k \Leftrightarrow n = 2k$.

2.4. Entropía

Para entender el concepto de entropía empezaremos comentando la importancia que tiene esta en la teoría de la información y como está relacionada con la estadística. Se puede entender la entropía como una función de probabilidad que surge del proceso de comunicación y que basa sus propiedades en la intuición de como se debería comportar un tipo de información.

Definición 2.37. La entropía de una variable aleatoria discreta X se define por:

$$H(X) = -\sum p(x) \ \ log_2 \ \ p(x)$$

La entropía corresponde con el número de bits necesarios para describir una variable aleatoria X, es decir, es la incertidumbre de una sola variable aleatoria que es medida en bits.

Por la definición de entropía obtenemos las siguientes consecuencias:

- $H(X) \geq 0$: Dado que $0 \leq p(x) \leq 1$, entonces $\log\ (\frac{1}{p(x)}) \geq 0$.

- La entropía puede cambiar de una base a otra multiplicando por el factor adecuado: $H_b(X) = (log_b\ a)H_a(X)$. Esto puede observarse fácilmente: $log_b\ p = (log_b\ a)\ \ (log_a\ p)$.

Nota 2.38. Si la base del log es b, denotaremos la entropía como $H_b(X)$. Aunque depende del cuerpo que usemos, normalmente todos los logaritmos serán base 2, dado que condiciona que la medición se realice en bits. Además, cambiar de base es sencillo como hemos observado.

Ejemplo 2.2. Vamos a suponer que acudimos al hipódromo a ver carreras de caballos y en la primera carrera que observamos participan 8 caballos y las probabilidades que cada uno gane es la siguiente:

$$(\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{64}, \frac{1}{64}, \frac{1}{64}, \frac{1}{64})$$

Por lo tanto, es posible calcular la entropía:

$$H(X) = -\frac{1}{2}\ \ (log_2\ \ (\frac{1}{2})) - \frac{1}{4}\ \ (log_2\ \ (\frac{1}{4})) - \frac{1}{8}\ \ (log_2\ \ (\frac{1}{8}))-$$

$$\frac{1}{16}\ \ (log_2\ \ (\frac{1}{16})) - 4(\frac{1}{64}\ \ (log_2\ \ (\frac{1}{64}))) = 2\ \ bits$$

Ejemplo 2.3. Un ejemplo muy ilustrativo en el que se pueden observar las propiedades de la entropía es el siguiente:

Definimos X de la siguiente manera:

$$X = \begin{cases} 1 & con\ \ probabilidad\ \ p, \\ 0 & con\ \ probabilidad\ \ 1-p. \end{cases}$$

Entonces,

$$H(X) = -p\ \ log(p) - (1-p)\ \ log(1-p) = H(p).$$

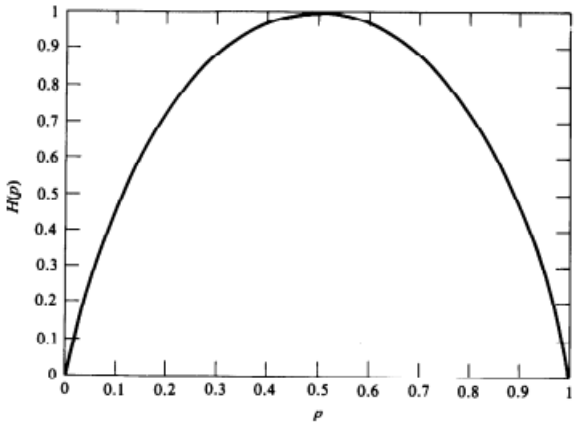

Imagen 2.1: H(p) versus p.

En la anterior figura podemos observar propiedades básicas de la entropía, ya que es una función de distribución que es igual a 0 cuando $p = 0$ o 1. Esto tiene sentido porque en estos casos no existe incertidumbre y se tiene una certeza total acerca del proceso.

Si $p = 1/2$ entonces $H(X) = 1$ bit, entonces en este punto tanto la incertidumbre como la entropía es máxima y no tenemos ninguna razón para decir que algo tiene mayor probabilidad de ocurrir o no.

Definición 2.39. Si tenemos una variable aleatoria X y otra variable aleatoria Y llamamos entropía condicional, a la incertidumbre de X dado un valor particular de Y. Podemos denotar a la entropía condicional de un par de variables aleatorias (X, Y), como $H(X, Y)$, con una distribución conjunta $p(x, y)$:

$$H(X,Y) = -\sum_{x \in X} \sum_{y \in Y} p(x,y) \ \ log_2 \ \ p(x,y).$$

Definición 2.40. La reducción de la incertidumbre respecto a otra variable aleatoria se llama información mutua. La información mutua $I(X;Y)$ es una medida de dependencia que se da entre las dos variables aleatorias X e Y que son siempre no negativas y se denota por:

$$I(X;Y) = H(X) - H(X|Y) = \sum_{(x,y)} p(x,y) \ \ log_2 \ \ \frac{p(x,y)}{p(x)p(y)}.$$

Definición 2.41. Para un canal de comunicación con entrada X y salida Y, definimos la capacidad C por:

$$C = \max_{p(x)} \ I(X,Y)$$

Ejemplo 2.4. El ejemplo más sencillo es suponer que intentamos transmitir un mensaje por un canal binario sin ruido. En este canal, cualquier bit es transmitido sin error y la entrada binaria se reproduce exactamente en la salida. Entonces podemos enviar 1 bit en cada transmisión al receptor y la capacidad sería:

$$C = \max_{p(x)} \ I(X,Y) = 1 \ \ bit.$$

Capítulo 3

Esquemas de Comparticion de secretos.

3.1. Introducción

En esta sección explicaremos los esquemas de compartición de secretos (ECS). Se relacionará los ECS con los códigos lineales vistos en la sección anterior. Para este punto, hemos realizado el estudio de diferentes referencias: [3], [4], [5], [8], [19], [20], [21], [26] y [27].

En la antigüedad la forma de compartir un secreto, como la combinación de tres dígitos $17 - 14 - 92$ para una cerradura de combinación con 100 posiciones en su dial, sería dar parte del secreto a cada usuario, por ejemplo, 17 a Ana, 14 a Bob y 92 a Eva, indicando a cada usuario la posición de su parte del secreto. Esta forma de compartir el secreto tiene dos inconvenientes: el primero es que el secreto tiene más posibilidades de ser descubierto a medida que se aprenden más particiones.
En el ejemplo, cada partición ya reduce las posibilidades de recuperar la clave a sólo 10^4 de los 10^6 combinaciones posibles y el conocimiento de dos particiones más reduce esta probabilidad a sólo 10^2 combinaciones.
En segundo lugar, todas las particiones son necesarias para determinar el secreto. Si Eva pierde su parte, Ana y Bob, después de combinar sus partes, deben experimentar para determinar cuál de las 10^2 combinaciones posibles restantes es la correcta.

Los esquemas de compartición de secretos, surgen de la importancia de proteger y preservar la información, de esta manera se hace necesario el uso de métodos robustos para tener acceso a ella.
Los ECS se encargan de controlar el acceso a la información, distribuyendo la responsabilidad entre varios usuarios. Además, fragmentan en particiones un dato o "secreto" (como puede ser una clave numérica en el ejemplo anterior) y lo distribuyen entre los participantes de la comunicación, de forma que solo los participantes

autorizados pueden reconstruir el secreto a partir de sus particiones.
Queremos destacar que cuando hablamos de repartir o dividir un secreto, no entendamos este concepto como partir el secreto en distintas partes de información, si no hacemos referencia a que tomando el secreto podemos construir las particiones que posteriormente se repartirán entre los distintos participantes (esto implica que las particiones no tienen porque ser parte del secreto). Estas particiones se realizan con el fin de proteger la información y si logramos juntar un número lo suficientemente grande de particiones, esto nos permitirá reconstruir el secreto por completo. Entonces nosotros realizaremos el estudio de los esquemas de compartición de secretos, teniendo clara la idea de que a lo largo del presente documento o bien sabemos todo sobre el secreto o no sabemos nada sobre él.

Existen distintas metodologías a la hora de repartir el secreto, [23][11]. Nosotros basaremos nuestro documento en la estructura de acceso y los esquemas de umbral:

- **ECS de estructura de acceso general**: sólo los miembros pertenecientes a la estructura de acceso (EA) pueden reconstruir el secreto. (Sección 3.3).
- **ECS de umbral**: son básicamente los introducidos por Shamir (mediante Interpolación de Polinomios) y Blakley (el cual se basa en la Geometría Proyectiva). En los esquemas de umbral el número de particiones es mayor o igual al número de posibles secretos. En concreto, podemos representar a los esquemas de umbral como una EA general. (Sección 3.4).
- **ECS perfectos**: son esquemas en los que existe el mismo número de participantes que de posibles particiones. Para los esquemas perfectos, el tamaño de cada partición debe ser al menos tan grande como el tamaño del secreto. En este tipo de esquemas o se tiene plena información sobre el secreto o no se tiene conocimiento alguno sobre él. Por ejemplo, el esquema umbral de Shamir o Blakley son esquemas perfectos. (Sección 3.4).
- **ECS no perfectos**: son esquemas en el que el tamaño de cada partición puede ser menor que el tamaño del secreto, esto implica que exista una alta tasa de información. En este tipo de esquemas se puede tener información parcial sobre el secreto. (Sección 3.6.3).

3.2. Parametrización

La compartición de secretos es muy relevante a día de hoy en la criptografía. En el modelo básico de compartición de secretos podemos distinguir entre dos protocolos a seguir:

(i) **Un protocolo de distribución** en el que existe un distribuidor D que se encarga de distribuir particiones del secreto S entre el conjunto finito de participantes P, de tal manera que $D \notin P$.

(ii) **Un protocolo de reconstrucción** que permite recuperar el secreto, agrupando las particiones de un subconjunto autorizado de los participantes. Esto se vera reflejado en los esquemas de umbral 3.4.

Antes de definir formalmente el esquema de compartición de secretos, fijaremos la siguiente notación que será utilizada a lo largo del capitulo:

Un vector de variables aleatorias, es un vector $X = (X_i)_{i \in I}$ tal que el conjunto de índices I es finito y no vacío. Mientras que X_i son variables aleatorias definidas en el mismo espacio finito de probabilidad, X es un vector de variables aleatorias con conjuntos de índices I y $A \subset I$, con $A \neq 0$. Además, X_A denota el vector de variables aleatorias $X = (X_i)_{i \in A}$.

Llamaremos P al conjunto finito de participantes y S a la información, que en este caso corresponde al secreto. Sea $H(X)$ la entropía de una variable aleatoria X.

Definición 3.1. Un esquema de compartición de secretos es un vector de variables aleatorias S con conjunto de índices $I = \{0, \ldots, n\}$ tales que:

- El conjunto S_i debe satisfacer $\mid S_i \mid > 1$.

- La uniformidad del secreto esta dada por S_0, es decir que el secreto será destinado en dicha variable S_0. Se determina S_0 por el resto de variables de S, $H(S_0 \mid S_{I \setminus \{0\}}) = 0$. Esto significa que no se sabe nada sobre el secreto en este momento pero es posible obtener su posterior reconstrucción.

Las variables aleatorias restantes se definen como un conjunto de índices $I \setminus \{0\} = \{1, \ldots, n\}$, que nombraremos como I^*. Esta notación se denota como un conjunto de índices I en el cual, el índice $I = 0$ esta reservado para el secreto, ya que este indica su destino. La cardinalidad $I \setminus \{0\} = \{1, \ldots, n\}$, la llamaremos $n(S)$.
Sea S_0 el secreto, y las variables restantes S_i, $i \in I \setminus \{0\}$ las particiones.
El encargado de repartir la información al conjunto de variables S_i se denomina distribuidor y se denota como D, de tal manera que $D \notin P$, es decir, D no se encuentra dentro del conjunto de participantes P.

Los esquemas de compartición de secretos se pueden utilizar para distribuir cierta información secreta. Si alguna de las partes (un distribuidor) quiere distribuir cierta

información $s \in S_0$ entre un conjunto de $n(S)$ partes (las que asumimos como I^*) lo hará de la siguiente manera:

El distribuidor muestrea un vector $a = (s, a_1, \ldots, a_n)$ según la distribución de la variable condicionada $S \mid S_0 = s$, luego para cada $i \in I^*$ el distribuidor envía el valor $a_i \in S_i$, siendo la variable S_i la $i-$ésima parte. El vector $a = (a_1, \ldots, a_n)$ recoge las particiones para llegar a reconstruir la información.

Nota 3.2. Las particiones a_i, $i = 1, \ldots, n$, en las que divide D el secreto S, tienen que ser distribuidas de manera confidencial. Ningún participante tiene acceso a la información de otro participante. Esto implica que la posterior reconstrucción del secreto solo se hace posible si contenemos un mínimo de participantes con sus respectivas particiones.

3.3. Estructuras de acceso

La finalidad del concepto de estructura de accesos (EA), es definir que conjuntos de participantes pueden o no llegar a reconstruir el secreto, una vez que se agrupan. Estos participantes capaces de reconstruir el secreto son llamados participantes autorizados y el conjunto de participantes autorizados, agrupación autorizada.

A lo largo del presente capítulo realizaremos construcciones, de tal manera que si se reúnen todas las particiones de una agrupación de participantes autorizados se les permita reconstruir íntegramente el secreto. Por el contrario, si se junta una agrupación de participantes no autorizados estos no podrán descubrir ninguna información sobre el secreto. Este tipo de esquemas se denominan esquemas perfectos.
En la última sección de este punto veremos los esquemas ECS no perfectos en los que observaremos que agrupaciones de particiones conjuntas dan información parcial sobre el secreto.

Definición 3.3. Dado un conjunto de participantes $\mathbb{P} = \{P_1, \ldots, P_n\}$, una estructura de acceso consiste en un conjunto Γ tal que $\Gamma \subseteq P(\mathbb{P})$. En estos términos, se dice que el esquema realiza la estructura de acceso Γ y cumple:

- El distribuidor $D \notin \mathbb{P}$.
- S es el conjunto de secretos.
- $\Gamma \subseteq P(\mathbb{P})$: subconjunto de partes de $\mathbb{P}$ que agrupándose pueden reconstruir el secreto.

- $\triangle \subseteq P(\mathbb{P})$: subconjunto de partes de $\mathbb{P}$ que agrupándose no pueden reconstruir el secreto.
- $\Gamma \cap \triangle = \emptyset$ y que $\Gamma \cup \triangle = P(\mathbb{P})$.
- a: conjunto de las particiones a repartir.
- $a_i \subseteq S_i, \ \forall i \in \{1, \ldots, n\}$: conjunto de las particiones que recibe cada participante P_i.

Cuando se desea distribuir un secreto $s \in S$ a partir de un esquema, el distribuidor D enviará a cada participante $P_i \in \mathbb{P}$ una partición $a_i \in S_i$. Se puede formular un esquema de estructura de acceso para la compartición del secreto en la que un secreto se transforma en particiones de tal manera que:

- El conocimiento de las particiones en cualquier conjunto en Γ determina el secreto.

- El conocimiento de las particiones de un conjunto que no está en Γ y que no tiene ningún subconjunto en Γ no revela ninguna información sobre el secreto. Según lo mostrado en la sección 2.4, si t participantes no tienen ninguna información sobre el secreto nos encontramos con que la entropía es 1. Por el contrario, en caso de que podamos reconstruir el secreto la entropía será 0. Cuanto más cercanos nos encontremos a una entropía igual a 0, más probabilidad tendremos de conseguir reconstruir el secreto.

Nota 3.4. Si un grupo de partes puede recuperar el secreto combinando sus particiones, entonces cualquier grupo de partes que este conteniendo en este grupo puede también recuperar el secreto.

Definición 3.5. Sea Γ una estructura de acceso, se dice que es una estructura de acceso monótona si:

$$\emptyset \in \Gamma, \ \textit{dado} \ \ B \in \Gamma \quad y \quad B \subseteq C \subseteq P \Longrightarrow C \in \Gamma.$$

Esta condición implica que si añadimos nuevos participantes a cualquiera de los subconjuntos autorizados de Γ, estos seguirán siendo agrupaciones de autorizados. En cambio, si eliminamos participantes de los subconjuntos no autorizados estas seguirán siendo agrupaciones no autorizadas.

Ejemplo 3.1. Sea $\mathbb{P} = \{P_1, P_2, P_3, P_4\}$:

- $\Gamma_1 = \{\{P_1, P_2\}, \{P_2, P_3\}, \{P_1, P_2, P_3\}, \{P_1, P_2, P_4\}, \{P_2, P_3, P_4\}, \mathbb{P}\}$ es una estructura de acceso monótona.
- $\Gamma_2 = \{\{P_1, P_2\}, \{P_2, P_4\}, \{P_1, P_2, P_5\}\}$ no es una agrupación monótona, dado que $\{P_2, P_4, P_5\}$ no es una agrupación autorizada y $\{P_2, P_4\}$ si lo es.

Se considerará que si una agrupación es capaz de recuperar el secreto es lógico pensar que esta agrupación más otro participante también será capaz de recuperar el secreto.

Nota 3.6. La estructura de acceso de un esquema de compartición de secretos, son los conjuntos de participantes que determinan el secreto.
Un grupo de participantes es llamado conjunto mínimo de acceso, si cualquiera participante n puede recuperar el secreto con sus particiones, pero cualquier subgrupo propio de este no lo puede realizar.

Nota 3.7. La tasa de información de un esquema ECS, es la relación entre el tamaño del secreto y el tamaño máximo de la partición. En los esquemas perfectos el tamaño de cada partición es al menos tan grande como el tamaño del secreto. Una ventaja de los esquemas no perfectos es que el tamaño de cada partición puede ser menor que el tamaño del secreto, esto implica que exista una alta tasa de información.

Definición 3.8. Un esquema de compartición de secretos que contiene una estructura de accesos $(\Gamma, \triangle)$ se dice que es perfecto si:

- Sea un conjunto autorizado de participantes $A \in \Gamma$, este puede determinar el secreto $s \in S$, si computa sus particiones $a_{iA} \in a$.
- Sea un conjunto no autorizado de participantes $B \in \triangle$, si unen en conjunto sus particiones no puede saber ningún tipo de información del secreto.

Como hemos explicado en el ejemplo que introduce la sección, es claro que a partir de un conjunto P existen diferentes estructuras de acceso, en concreto $\mathbb{P}(P)$. Estas estructuras de acceso son equivalentes si no se tiene en cuenta la diferencia entre que formen parte de ellas unos participantes u otros.

Definición 3.9. Sean Γ_1, Γ_2 dos estructuras de acceso definidas sobre un mismo P. Si definimos la permutación σ en el conjunto de índices $\{1, \ldots, n\}$ de forma que:

$$\{P_{i1}, \ldots, P_{ir}\} \in \Gamma_1 \Leftrightarrow \{P_{\sigma(i_1)}, \ldots, P_{\sigma(i_r)}\} \in \Gamma_2$$

Estas estructuras son equivalentes y además Γ_2 puede ser generada por Γ_1 renombrando a sus participantes.

Ejemplo 3.2. Sea $P = P_1, P_2, P_3$ y las estructuras de acceso:

$$\Gamma_1 = \{\{P_1, P_3\}, \{P_1, P_2, P_3\}\}$$
$$\Gamma_2 = \{\{P_1, P_2\}, \{P_1, P_2, P_3\}\}$$

son equivalentes, ya que podemos renombrar $P_{\sigma_{(3)}} = P_2$ y $P_{\sigma_{(2)}} = P_4$.

En el momento que un esquema de compartición de secretos se construye a partir de una estructura de accesos Γ, en la que sus conjuntos son los subconjuntos de $P(\mathbb{P})$ compuestos por al menos $t+1$ participantes, se dice que es un esquema umbral (t, n). A continuación, veremos como t particiones no tienen ningún conocimiento del secreto y $t+1$ particiones pueden llegar a reconstruir el secreto.

3.4. Esquema umbral

Existen diferencias entre los diferentes esquemas de compartición de secretos, *ECS*. Tales consideraciones llevaron a Shamir, mediante la Interpolación de Polinomios [21] y a Blakely que se basa en la Geometría Proyectiva [1] a formular, de forma independiente, en 1979, los llamados esquemas umbral para la compartición de secretos.

Para poder acceder a todo el contenido del secreto, necesitamos conocer el concepto de umbral. En este sentido, se entiende que si tenemos un número determinado de particiones del secreto S que denotaremos por umbral de privacidad $t(S)$, del que existen $n(S)$ particiones, no tenemos ningún conocimiento sobre el secreto. Pero si conocemos un número de particiones que denominaremos umbral de reconstrucción y denotaremos por $r(S)$, podemos llegar a la reconstrucción del mismo en su totalidad.

Nosotros consideraremos los esquemas de umbral (t, n) en el que el secreto se transforma en una lista de particiones n de tal manera que:

El conocimiento del umbral de reconstrucción $r(S) = t+1$ particiones (donde por conocimiento de una partición siempre se entenderá conocimiento tanto de su valor como de su posición en la lista de particiones) revela el secreto, pero el conocimiento del umbral de privacidad $t(S)$ o menos particiones no da ninguna información sobre el secreto.

Definición 3.10. Un esquema (t, n) umbral es una disposición de acceso sobre un conjunto P que contiene n participantes, con sus respectivas particiones del secreto S ($n(S)$), de manera que un conjunto de al menos $t+1$ participantes, con sus respectivas particiones del secreto S, son capaces de recuperar el secreto. Sin embargo, en el

caso que se junten t participantes con sus respectivas particiones del secreto S ($t(S)$), no se obtendrá ninguna información sobre el secreto.

Definición 3.11. Sea $A \subset I \setminus \{0\} = \{1, \ldots, n\}$ con $A \neq \emptyset$. Luego:

- A representa un conjunto de reconstrucción si $H(S_0 \mid S_A) = 0$. Esto significa que las particiones del conjunto A de forma conjunta pueden determinar el secreto con probabilidad 1. El umbral de reconstrucción se representa como $r(S)$.

- También, A es un conjunto de privacidad si $H(S_0|S_A) = H(S_0)$. Esto significa que la incertidumbre sobre el secreto después de que se otorguen la particiones para A, es igual a la incertidumbre sobre el secreto antes de que se otorguen las particiones.

3.5. Esquema de Shamir

El esquema de Shamir es el ejemplo más famoso de un esquema de compartición de secretos. La estructura de acceso está formada por los subconjuntos con al menos t participantes de un conjunto con n participantes. Este esquema es como hemos definido anteriormente un esquema $(t, n)-$umbral.

3.5.1. Introducción

El esquema de Shamir tiene como base la interpolación de polinomios para la distribución del secreto S entre n participantes. Necesitaremos $t + 1$ participantes para la reconstrucción de S.

Este esquema consiste en ocultar información en un polinomio aleatorio. La utilización de este polinomio nos permite realizar particiones basadas en evaluaciones del polinomio. Dada cierta información parcial del polinomio, podemos llegar a recuperar el secreto que estaba escondido en él. En el caso de que un participante recupera las suficientes particiones se podrá entonces recuperar el secreto completo.

Explicaremos a continuación el protocolo que propone Shamir para la distribución del secreto.

Sea un secreto S y un número entero t tal que $t \leq n$. Sea un numero primo $q \geq \max\{S + 1, n + 1\}$. Se considera $q > n$, entonces el distribuidor D elige n elementos

independientes y distintos de cero de $\mathbb{F}_q$, que se denotan como x_i, con $1 \leq i \leq n$. Para $1 \leq i \leq n$, D asigna el valor x_i a P_i, de forma pública.

Cuando D comparte el secreto $S \in \mathbb{F}_q$, toma al azar t elementos de $\mathbb{F}_q$, denotados como $a_1, \ldots, a_t$, estos elementos corresponden a los coeficientes que forman el polinomio, el cual debe ser de un grado menor o igual a t, de esta manera el coeficiente que contendrá el secreto siempre será a_0, tal que $a_0 = S$.

El polinomio requerido se muestra a continuación

$$f(x) = a_0 + \sum_{j=1}^{t} a_j x^j \in \mathbb{F}_q[X]$$

D calcula $y_i = f(x_i)$, y otorga la partición y_i a P_i, es obvio que $a_0 = f(0)$.

Como hemos observado el esquema de Shamir utiliza el siguiente principio matemático:

Si tenemos k puntos del plano $(x_0, y_0), \ldots (x_k, y_k)$, donde los x_i son diferentes, de tal manera que siempre existe un único polinomio f que tenga grado menor o igual que $k-1$ y $f(x_i) = y_i$, para $i = 1, \ldots, t+1$.
En este tipo de esquemas tener t particiones, es equivalente a no conocer nada. Esto implica que el umbral la privacidad es $t(S) = t$ y que el umbral de reconstrucción sería $r(S) = t+1$. Esta explicación gana sentido gracias al siguiente Teorema, ya que ahora nos preguntaremos que ocurre si nos encontramos con que $k \leq t$.

Teorema 3.12. *Si se unieran menos de t participantes estos no tendrían ninguna posibilidad de saber nada sobre el secreto S, siendo $k \leq t$.*

Demostración. Tomamos $g(X)$ como el polinomio interpolador de las k particiones. El grado de $g(X)$ es menor o igual que $k \leq t$. Por lo tanto los polinomios,

$$q(X) = h(X) \prod_{j=1}^{k-1} (X - x_j) + g(X)$$

Indiferentemente del polinomio $h(X)$ de grado $t-k$ satisfacen que $deg\ q(X) = t$ $q(x_i) = g(x_i) = y_i, 1 \leq i \leq k$.

Si tomáramos en particular $h_0(X)$, obtendremos el polinomio $f(X)$:

$$S = a_0 = f(0) = h_0(0) \prod -x_j + g(0)$$

y como se desconoce $h_0(0)$, la coalición de los k usuarios no podría recuperar S. □

La principal desventaja que existe en este tipo de esquemas es que solo son posibles si $q > n$, ya que no si esto no se cumple no tendríamos los suficientes puntos donde evaluar el polinomio.
Entonces es necesario que conozcamos k puntos ($t + 1$ particiones) para reconstruir f usando la fórmula de interpolación de Lagrange que introduciremos a continuación.

3.5.2. Interpolación de Lagrange

Para la reconstrucción del secreto se utiliza la interpolación de Lagrange, debido a que es posible reconstruir polinomios utilizando varios puntos en los que se evalúa el polinomio.

Cada participante obtiene una serie de puntos. Se pueden juntar un conjunto de participantes lo suficientemente grande, pudiendo llegar a reconstruir el secreto. La fórmula de la interpolación de Lagrange permite la reconstrucción del polinomio.

Nota 3.13. La reconstrucción integra del polinomio no es necesaria para la reconstrucción del secreto, ya que podemos llegar de la siguiente manera:

Calculamos $S_i' = \prod_{i \neq j}^{k} \frac{-x_j}{x_i - x_j} \mod p$.

Realizamos las reconstrucción:

$$S = \sum_{i=1}^{t+1} y_i \ S_i' \mod p$$

Definición 3.14. Interpolación polinomial de Lagrange:

Dado un conjunto de k puntos

$$(x_0, y_0), \ldots, (x_k, y_k)$$

donde todos los x_j se asumen distintos, el polinomio interpolador en la forma de Lagrange es la combinación lineal:

$$L(x) = \sum_{j=0}^{k} y_j * \ell_j(x)$$

De bases polinómicas de Lagrange:

$$\ell_j(x) = \prod_{i=1, i\neq j}^{k} \frac{x - x_i}{x_j - x_i} = \frac{x - x_0}{x_j - x_0} * \ldots * \frac{x - x_{j-i}}{x_j - x_{j-i}} * \frac{x - x_{j+i}}{x_j - x_{j+i}} * \ldots * \frac{x - x_k}{x_j - x_k}$$

Para todo $i \neq j$, $\ell_j(x)$ incluye en el numerador el término $(x - x_i)$, de modo que el producto completo valdrá cero en $x = x_i$.

$$\forall (j \neq i) : \ell_j(x_i) = \prod_{m\neq j}^{k} \frac{x_i - x_m}{x_j - x_m} = \frac{x_i - x_0}{x_j - x_0} * \ldots * \frac{x_i - x_i}{x_j - x_i} * \ldots * \frac{x_i - x_k}{x_j - x_k} = 0$$

Por otro lado,

$$\ell_j(x_j) = \prod_{m\neq j}^{k} \frac{x_j - x_m}{x_j - x_m} = 1$$

Todas las bases polinómicas de Lagrange valen cero en $x = x_i$, excepto $\ell_j(x)$, para el que aplica $\ell_j(x_j) = 1$, puesto que carece del término $(x - x_j)$ en el numerador.

Por tanto, se deriva que $y_j \ell_j(x_j) = y_j$, en cada punto x_j,

$$L(x_j) = y_j + 0 + 0 + \ldots + 0 = y_j,$$

demostrando que L interpola la función de forma exacta.

Ejemplo 3.3. Explicaremos este procedimiento con un pequeño ejemplo numérico. Tomaremos $\mathbb{F}_q = 11$ y un polinomio L que tiene los siguientes valores $y_1 = 2,\ y_2 = 4,\ y_3 = 3,\ y_4 = 4$ en los primeros 4 puntos y 0 en los demás puntos.

Tomamos $\ell_1, \ell_2, \ell_3, \ell_4$, uno por cada punto del polinomio L. Para reconstruir el polinomio, mostramos lo que haríamos con ℓ_1 y se seguirá el mismo procedimiento para $\ell_2,\ \ell_3,\ \ell_4$. Los polinomios tienen como valor 1 en un solo punto y en los demás puntos el valor 0. De tal manera que tomaremos $\ell_1(1) = 1, \ell_1(2) = 0, \ell_1(3) = 0$ y $\ell_1(4) = 0$.

Para la reconstrucción del polinomio L tomamos cada polinomio y le multiplicamos por el coeficiente correspondiente. Si sumamos todos nos dará como resultado:

$$2 \cdot \ell_1 + 4 \cdot \ell_2 + 3 \cdot \ell_3 + 4 \cdot \ell_4.$$

Podemos observar que la suma en el primer punto da como resultado $2 + 0 + 0 + 0$, lo mismo sucede en los demás puntos.

Entonces tenemos el polinomio $(x-2)(x-3)(x-4)$, y se visualizará que las evaluaciones de los ℓ_1 en los puntos son correctas y nos dan como resultado $\ell_1(2) = 0, \ell_1(3) = 0$ y $\ell_1(4) = 0$. Para que $\ell_1(1) = 1$, tenemos que dividir entre -6 ya que $(1-2)(1-3)(1-4) = -6$.

La fórmula para la construcción de ℓ_1 sería

$$\ell_1 = \prod_{i=1, i\neq j}^{k} \frac{x-x_i}{x_j-x_i} = \frac{x-2}{1-2} \cdot \frac{x-3}{1-3} \cdot \frac{x-4}{1-4} = \frac{(x-2)(x-3)(x-4)}{-6},$$

dando como resultado $\ell_1(1) = 1, \ell_1(2) = 0, \ell_1(3) = 0$ o $\ell_1(4) = 0$. Y se puede construir ℓ_1, ℓ_2, ℓ_3, ℓ_4, de la misma manera.

$$\ell_2 = \prod_{i=2, i\neq j}^{k} \frac{x-x_i}{x_j-x_i} = \frac{x-1}{2-1} \cdot \frac{x-3}{2-3} \cdot \frac{x-4}{2-4} = \frac{(x-1)(x-3)(x-4)}{-6}$$

$$\ell_3 = \prod_{i=3, i\neq j}^{k} \frac{x-x_i}{x_j-x_i} = \frac{x-1}{3-1} \cdot \frac{x-2}{3-2} \cdot \frac{x-4}{3-4} = \frac{(x-1)(x-2)(x-4)}{-6}$$

$$\ell_4 = \prod_{i=3, i\neq j}^{k} \frac{x-x_i}{x_j-x_i} = \frac{x-1}{4-1} \cdot \frac{x-2}{4-2} \cdot \frac{x-3}{4-3} = \frac{(x-1)(x-2)(x-3)}{-6}$$

Si tomamos los valores y_1, y_2, y_3 e y_4 podemos construir la fórmula de $L(0)$.

Por la interpolación de Lagrange $L(x) = \sum_{j=1}^{k} y_j * \ell_j(x)$ y la evaluación en 0 de los polinomios.

$$L(0) = y_1 \cdot \ell_1(0) + y_2 \cdot \ell_2(0) + y_3(0) \cdot \ell_3(0) + y_4 \cdot \ell_4(0) =$$

$$2 \cdot \ell_1(0) + 4 \cdot \ell_2(0) + 3 \cdot \ell_3(0) + 4 \cdot \ell_4(0) = 2 \cdot 4 + 4 \cdot 6 + 3 \cdot 4 + 4 \cdot 1 = 48 \quad \text{mód } 11 = 4.$$

En los (ECS) sabemos que el grado del polinomio está acotado. El distribuidor compartirá el secreto eligiendo un polinomio de coeficientes aleatorios pero a los sumo de grado t. Entonces necesitaremos de $t+1$ puntos para reconstruir $L(x)$.

3.5.3. Ejemplo Esquema de Shamir

Como hemos comentado al principio de la sección, las particiones dadas a cada participante se encuentran en $\mathbb{F}_q$. El distribuidor D construye un polinomio aleatorio $f(x) \in \mathbb{F}_q[x]$ que tiene como grado máximo t. Cada participante P_i obtiene un par (x_i, y_i) del polinomio.

Para explicar mejor este esquema trataremos un ejemplo numérico muy sencillo que nos permitirá entender el protocolo a la perfección.

Ejemplo 3.4. Enunciado: Supongamos que queremos compartir un dato que corresponde a $S = 10$ y tomamos $\mathbb{F}_q = 11$. El secreto S se desea dividir en 6 partes $(n = 6)$; de forma que cualquier subconjunto $(t + 1 = 3)$ sea suficiente para reconstruir el secreto, entonces estamos describiendo un esquema umbral $(2, 6)$.

Al azar se obtienen t valores, por ejemplo, 120 y 89. Se usan coeficientes que definen estos valores ($a_1 = 120 \mod 11 = 10$; $a_2 = 89 \mod 11 = 1$), mientras que el coeficiente que corresponde al secreto se define como $a_0 = S = 10$.
Se describe el polinomio en un orden especifico:

$$f(x) = a_0 + a_1 x + a_2 x^2$$

$$f(x) = 10 + 10x + x^2$$

Posteriormente dado que $n = 6$, se calculan seis puntos a partir del polinomio, es decir, evaluamos el polinomio cuando $x = 1, 2, \ldots, 6$ de manera que se obtienen puntos de la forma $(x_i, f(x_i))$.
$(1, f(1)); (2, f(2)); (3, f(3)); (4, f(4)); (5, f(5)); (6, f(6))$, evaluando estos valores en el polinomio, se obtienen los puntos:

$$(1, 10); (2, 1); (3, 5); (4, 0); (5, 8); (6, 7)$$

A cada participante le corresponde un único punto x y $f(x)$.

Reconstrucción: como se describió anteriormente, para reconstruir el secreto bastará con tres puntos $(t + 1 = 3)$. Considere 3 puntos cualesquiera

$$(x_0, y_0) = (2, 1); (x_1, y_1) = (3, 5); (x_2, y_2) = (5, 8)$$

Usando la interpolación polinómica de Lagrange:

$$\ell_0 = \frac{(x - x_1)}{(x_0 - x_1)} * \frac{(x - x_2)}{(x_0 - x_2)} = \frac{(x - 3)}{(2 - 3)} * \frac{(x - 5)}{(2 - 5)} = (\frac{1}{3}x^2 - \frac{8}{3}x + 5) \mod 11 = \frac{1}{3}x^2 + x + 5$$

$$\ell_1 = \frac{(x-x_0)}{(x_1-x_0)} * \frac{(x-x_2)}{(x_1-x_2)} = \frac{(x-2)}{(3-2)} * \frac{(x-5)}{(3-5)} = (-\frac{1}{2}x^2 + \frac{7}{2}x - 5) \quad \text{mód } 11 = 5x^2 + \frac{7}{2}x + 6$$

$$\ell_2 = \frac{(x-x_0)}{(x_2-x_0)} * \frac{(x-x_1)}{(x_2-x_1)} = \frac{(x-2)}{(5-2)} * \frac{(x-3)}{(5-3)} = (\frac{1}{6}x^2 - \frac{5}{6}x + 1) \quad \text{mód } 11 = \frac{1}{6}x^2 + x + 1$$

Por lo tanto,

$$f(x) = \sum_{j=0}^{k} y_j * \ell_j(x)$$

$$= 1 * (\frac{1}{3}x^2 + x + 5) + 5 * (5x^2 + \frac{7}{2}x + 6) + 8 * (\frac{1}{6}x^2 + x + 1)$$

$$= 10 + 10x + x^2$$

Teniendo en cuenta que el secreto es el coeficiente a_0, entonces el secreto original es $S = 10$.

3.6. Esquemas de Compartición de Secretos lineales

Los esquemas de compartición de secretos lineales son los más comunes en el campo de la compartición de secretos.

Definición 3.15. Un esquema de compartición de secretos lineal $(LSSS)$ sobre $\mathbb{F}_q$, es un esquema de compartición de secretos donde se verifica que dado un secreto S con particiones $(y_1, \ldots, y_n)$ y un secreto S' con particiones $(y'_1, \ldots, y'_n)$ se tiene un vector de particiones de $\lambda S + \lambda S'$ con $\lambda, \lambda' \in \mathbb{F}_q$, tal que: $\lambda(y'_1, \ldots, y'_n) + \lambda'(y'_1, \ldots, y'_n)$.

Proposición 3.16. *El esquema de Shamir es un esquema de compartición de secretos lineal, LSSS:*

Si los secretos $s_1, \ldots, s_e$ se comparten con el vector de particiones $(y_1^j, \ldots, y_n^j)$, para cada $j = 1, \ldots, e$. Entonces para $\lambda_1, \ldots, \lambda_e \in \mathbb{F}_q$:

$$(\sum_{j=1}^{e} \lambda_j y_1^j, \ldots, \sum_{j=1}^{e} \lambda_j y_n^j)$$

Ahora es un vector de particiones para $\sum_{j=1}^{e} \lambda_j s_j$ en el mismo esquema.

Demostración. Si tenemos las siguientes particiones:

$(y_1, \ldots, y_n)$ particiones de S.

$(y'_1, \ldots, y'_n)$ particiones de $S + S'$.

Entonces $(y_1 + y'_1, \ldots, y_n + y'_n)$ son particiones de $S + S'$.

Nosotros sabemos que $f(x_i) = S + \sum y_i x_i$ y $f'(x_i) = S' + \sum y'_i x_i$, con $i = 1, \ldots, n$. Además $f(0) = S$ y $f'(0) = S'$. Por lo tanto, tenemos que:

$$f(x_i) + f'(x_i) = (f + f')(x_i) = S + S' + \sum (y_i + y'_i)x_i,$$

$$(f + f')(0) = S + S'.$$

□

3.6.1. Método de Massey

En la sección 3.5 vimos como el esquema de Shamir solo era posible si $q > n$, ya que si esto no se cumple no se podría recuperar el secreto, dado que no tendríamos los suficientes puntos donde evaluar el polinomio. La aplicación del método de Massey, nos dará la ventaja de trabajar sobre el cuerpo que deseemos.

La idea de Massey es una de las formas más habituales para a partir de un código lineal formar un esquema de compartición de secretos. El código lineal nos permite "dividir"secretos en particiones de igual tamaño. Massey es un método para la construcción de un esquema de compartición de secretos a partir de un código de corrección de errores, [12, 26].

Nosotros nos centraremos a partir de ahora en los *LSSSs* en el que todos los espacios compartidos son el cuerpo $\mathbb{F}_q$ y el secreto que esta en $\mathbb{F}_q^\ell$, para $\ell \geq 1$. Es útil describir estos *ECS* desde el punto de vista de los códigos lineales. El soporte de la variable S es un código lineal C sobre $\mathbb{F}_q$ de longitud $\ell + n$, es decir, un subespacio lineal de $\mathbb{F}_q^{\ell+n}$.

Definición 3.17. Sea $1 \leq \ell < m$ números enteros. Dado un código lineal C subespacio lineal de $\mathbb{F}_q^{\ell+n}$, definimos el vector de variables aleatorias $S^{(\ell)}(C) = (S_i)_{i=0,\ldots,n}$, donde $n = m - \ell$, cuya distribución viene dada por la selección de $c = (c_{0,1}, c_{0,2}, \ldots, c_{0,\ell}, c_1, c_2, \ldots, c_n))$ uniformemente al azar en C y definiendo el secreto S_0 como $(c_{0,1}, c_{0,2}, \ldots, c_{0,\ell})$ y la partición $i-$ésima S_i como c_i para $i = 1, \ldots, n$.

Existen varías maneras para a partir de los códigos lineales construir un esquema de compartición de secretos, en esta sección mostraremos uno de los métodos más conocidos de este tipo de construcciones de esquemas, el método de Massey.

Teorema 3.18. *Sea C un código $[n+1,k,d]$ sobre un cuerpo finito $\mathbb{F}_q$ y sea la matriz generatriz $G=(g_0,g_1,\ldots,g_n)$ de C. Dado un valor secreto $s\in\mathbb{F}_q$, si seleccionamos una palabra clave al azar $a=(a_0,a_1,\ldots,a_n)\in C$, tal que $a_0=s$, se define el vector de particiones como $(a_1,\ldots,a_n)$. Si tomamos un vector aleatorio $u\in\mathbb{F}_q^n$, tal que $ug_0=a_0=s$ Entonces $LSSS(C)$ es un esquema lineal de compartición de secretos.*

Demostración. Probaremos la linealidad del código $LSSS(C)$. Si tomamos el vector aleatorio $u\in\mathbb{F}_q^n$:

$$uG=u(g_0,g_1,\ldots,g_n)=(ug_0,ug_1,\ldots,ug_n)=(S,c_1,\ldots,c_n),$$

Y el vector aleatorio $u'\in\mathbb{F}_q^n$:

$$u'G=u'(g_0,g_1,\ldots,g_n)=(u'g_0,u'g_1,\ldots,u'g_n)=(S',c_1',\ldots,c_n').$$

Entonces,

$$\begin{aligned}(u+u')G=(u+u')(g_0,g_1,\ldots,g_n)&=((u+u')g_0,(u+u')g_1,\ldots,(u+u')g_n)\\&=(S+S',c_1+c_1',\ldots,c_n+c_n')=uG+u'G.\end{aligned}$$

□

Massey

Dado el esquema lineal descrito en el teorema 3.18, al que nombramos como $LSSS(C)$, con estructura de acceso $\Gamma(C)$. Sea $C^\perp$ un código $[n+1,n+1-k,d^\perp]$ Considerando el conjunto V_0 de todo $a'\in C^\perp$ tal que $a_0'=1$. Entonces para un vector x y por la definición de soporte 2.3 $\Gamma(C)=\{sop(a')\ 0:a'\in V_0\}$.

Sea la matriz generatriz $G=(g_0,g_1,\ldots,g_n)\in C$, que tiene distintos de cero los vectores columna. El secreto es un elemento del cuerpo finito $\mathbb{F}_q$. Dado un conjunto de participantes $\mathbb{P}=\{P_1,\ldots,P_n\}$ y un distribuidor D.
Para construir las particiones de un secreto $s\in S$, el distribuidor D elige un vector aleatorio $u=(u_0,u_1,\ldots,u_n)\in\mathbb{F}_q^n$ tal que $s=ug_0$.
El elemento u es el vector de información y es el encargado de la codificación de tal manera que:

$$uG=u(g_0,g_1,\ldots,g_n)=(ug_0,ug_1,\ldots,ug_n)=(a_0,a_1,\ldots,a_n)=a.$$

Entonces D asigna a_i a cada P_i como su partición para cada $i \geq 1$

El siguiente teorema relaciona las particiones con las columnas de la matriz generatriz del código lineal.

Teorema 3.19. *Las particiones*

$$(a_{i_1}, a_{i_2}, \ldots, a_{i_m}) = u(g_{i_1}, \ldots, g_{i_m}),$$

donde $G = (g_0, g_1, \ldots, g_n)$ *es la matriz generatriz de un código lineal, se encargan de determinar el secreto si y sólo si* g_0 *es una combinación lineal de* $\{g_{i_1}, \ldots, g_{i_m}\}$.

Demostración. $\Leftarrow$) Si g_0 es una combinación lineal de $gi_1, \ldots, g_{i_m}$, entonces:

$$g_0 = c_{i_1} g_{i_1} + \cdots + c_{i_m} gi_m,$$

además,

$$ug_0 = uc_{i_1} g_{i_1} + \cdots + uc_{i_m} g_{i_m}.$$

Y llegamos a:

$$s = c_{i_1} a_{i_1} + \cdots + c_{i_m} a_{i_m}.$$

$\Rightarrow$) Por otro lado si $s = c_{i_1} a_{i_1} + \cdots + c_{i_m} ci_m$, entonces

$$ug_0 = c_{i_1} ug_{i_1} + \cdots + c_{i_m} ug_{i_m},$$

y

$$s = ug_0 = u(c_{i_1} g_{i_1} + \cdots + c_{i_m} g_{i_m}).$$

Por lo tanto,

$$g_0 = c_{i_1} g_{i_1} + \cdots + c_{i_m} g_{i_m},$$

dado que esta igualdad surge $\forall u \in \mathbb{F}_q^n$ tal que $s = ug_0$, obtenemos dos funciones lineales iguales. □

Con la explicación de este Teorema obtenemos el siguiente resultado sobre el código dual de C.

Corolario 3.20. *Dada una matriz generatriz* G *del código lineal* C $[n, k]$*, decimos que el conjunto de particiones* $\{a_{i_1}, a_{i_2}, \ldots, a_{i_m}\}$ *determina el secreto si y solo si existe una palabra* c *en el código dual* $C^{\perp}$ *tal que:*

$$(1, 0, \ldots, 0, c_{i_1}, 0, \ldots, 0, c_{i_m}, 0, \ldots, 0) \qquad (1),$$

donde $c_{i_j} \neq 0$ *para al menos una* j, $1 \leq i_1 < \cdots < i_m \leq n \quad y \quad 1 \leq m \leq n$.

Demostración. La matriz G es una matriz generatriz del código dual $C^{\perp}$. Si cogemos una palabra c tal que (1) en $C^{\perp}$ tenemos:

$$G(1,0,\ldots,0,c_{i_1},0,\ldots,0,c_{i_m},0,\ldots,0)^{\perp}$$

$$=(g_0,g_1,\ldots,g_n)(1,0,\ldots,0,c_{i_1},0,\ldots,0,c_{i_m},0,\ldots,0)^{\perp}=0$$

si y sólo si,

$$g_0+c_{i_1}g_{i_1}+\cdots+c_{i_m}g_{i_m}=0,$$
$$g_0=-c_{i_1}gi_1-\cdots-c_{i_m}g_{i_m}.$$

Utilizando el Teorema anterior se comprueba el resultado.

□

Para el siguiente corolario nos ayudará recordar los conceptos explicados en las notas 3.6 y 2.3.

Corolario 3.21. *Sea C un código lineal $[n,k]$, si creamos un esquema de compartición de secretos de C existirá una correlación entre todos los conjuntos de mínimo acceso y el conjunto de palabras mínimas cuya primera entrada es 1 del código dual $C^{\perp}$.*

Demostración. Si tomamos A como un conjunto de mínimo acceso y $a = \{a_{i_1},a_{i_2},\ldots,a_{i_m}\}$ su conjunto de particiones que determina el secreto, entonces existirá por el Corolario anterior una palabra en el código dual $C^{\perp}$ tal que:

$$c=(1,0,\ldots,0,c_{i_1},0,\ldots,0,c_{i_m},0,\ldots,0)$$

y

$$s=-c_{i_1}a_{i_1}-\cdots-c_{i_m}a_{i_m}$$

siendo $c_{ij}\neq 0$ para todo j tal que $1\leq j\leq m$. Por otro lado, c es una palabra mínima del código $C^{\perp}$, ya que los coeficientes de las particiones a_i correspondientes a conjuntos mínimos de access son todos distintos de cero, dado que si tenemos un coeficiente c_{ij} que es cero, se puede eliminar la parte correspondiente a ese coeficiente. Ahora si c es una palabra mínima de $C^{\perp}$ con 1 en la primera entrada:

$$c=(1,0,\ldots,0,c_{i_1},0,\ldots,0,c_{i_m},0,\ldots,0),\ \ c_{ij}\neq 0,\ \ \forall j\in\{1,\ldots,m\},$$

por el Corolario anterior sabemos que existe un número de particiones $a = \{a_{i_1},a_{i_2},\ldots,a_{i_m}\}$ que determina el secreto s y que a su vez le corresponde un conjunto de mínimo acceso, ya que si llegamos a eliminar una de estas partes, otra vez por el mismo corolario se tiene una palabra contenida propiamente en c. □

Teorema 3.22. *Sea un código lineal C de tipo $[n+1,k,d]$ sobre $\mathbb{F}_q$. Entonces $LSSS(C)$ tiene como umbral de privacidad $t(s)=(d^{\perp}-2)$ y como umbral de reconstrucción $r(S)=(n-d+2)$.*

Demostración. Sea $LSSS(C)$ un esquema de compartición de secretos lineal que se forma a partir del método de Massey, que tiene como estructura de acceso $\Gamma(C) = (\Gamma(C^{\perp}))'$, es decir, la estructura de acceso de $LSSS(C)$ es el dual de la estructura de acceso de $LSSS(C^{\perp})$.

Un conjunto $A \in \Gamma(C)$ si y solo si hay un $a' \in C^{\perp}$ con $a'_0 = 1$ y $a_i = 0$ para todo $i \in \{1, \ldots, n\} \setminus A(:= \overline{A})$, entonces $\overline{A} \notin (\Gamma(C^{\perp}))'$.
$LSSS(^{\perp})$ rechaza todos los conjuntos de tamaño $d^{\perp} - 2$ debido a la caracterización de $\Gamma(C)$ y dado que $\Gamma(C) = (\Gamma(C^{\perp}))'$, $LSSS(C)$ acepta todos los conjuntos de tamaño $n - d + 2$.

El umbral de privacidad exacto es igual a:

$$t(S) = -2 + min\ \{w_H(a') : a' \in C^{\perp} : a'_0 = 1\} = (d^{\perp} - 2)$$

y el umbral de reconstrucción:

$$r(S) = n + 2 - min\ \{w_H(a) : a \in C : a_0 = 1\} = (n - d + 2).$$

□

Nota 3.23. Una vez explicado el método de Massey, notemos que una forma equivalente de presentar el esquema de Shamir, es gracias a la idea de Massey aplicada a los códigos Reed Salomon:

Sea $s \in \mathbb{F}_q$, tal que $n \leq q$. Fijamos $0, \alpha_1, \ldots, \alpha_n$ elementos distintos dos a dos en $\mathbb{F}_q$. Denotemos por $\mathbb{F}_q[X]_{\leq t}$ el conjunto de polinomios de grado menor que t y con coeficientes en $\mathbb{F}_q$. Para compartir un secreto $s \in \mathbb{F}_q$ entre n partes con este esquema, se elige un polinomio f en $\mathbb{F}_q[X]_{\leq t}$ al azar pero claro con la condición de que $f(0) = s$. Entonces la partición $i-$ésima se define como la evaluación $f(\alpha_i) \in \mathbb{F}_q$. Este esquema de compartición de secretos se construye como $S^{(\ell)}(C)$, con $\ell = 1$, a partir del código Reed Solomon:

$$C = \{(f(0), f(\alpha_1), \ldots, f(\alpha_n)) : f \in \mathbb{F}_q[X]_{\leq t}\}.$$

En el siguiente ejemplo ilustraremos como hemos definido la idea de Massey a lo largo de la sección.

Ejemplo 3.5. Sea el código lineal C $[6, 3, 3]$ sobre $\mathbb{F}_2$, que tiene como umbral de reconstrucción $r(S) = 5 - 3 + 2 = 4$ y con matriz generatriz:

$$G = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix}$$

La aplicación lineal asociada a esta matriz generatriz viene definida de la siguiente forma:

$$f : \mathbb{F}_2^3 \longrightarrow \mathbb{F}_2^6$$
$$a \longmapsto Ha^t$$
$$a = (a_1, a_2, a_3) \longmapsto c = (a_1, a_2, a_3, a_2 + a_3, a_1 + a_2, a_1 + a_3)$$

y matriz de control:

$$H = \begin{pmatrix} 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}$$

Consideremos el esquema $C^\perp$, si elegimos $u = (1, 0, 1)$. Como H es una matriz generatriz para $C^\perp$, entonces $s = (1, 0, 1) \cdot (0, 1, 1) = 1$.

Para conocer los conjuntos de mínimo acceso, tomamos todas las palabras mínimas del código C con uno en la primera entrada. Todas las palabras del código C son:

$$(0,0,0,0,0,0), (0,1,0,1,1,0), (0,0,1,1,0,1), (0,1,1,0,1,0),$$
$$(1,0,0,0,1,1), (1,1,1,0,0,0), (1,1,0,1,1,0) \ \ y \ \ (1,0,1,1,1,0)$$

Todas las palabras con uno en la primera entrada son palabras mínimas. Sean cinco participantes $\{P_1, P_2, P_3, P_4, P_5\}$ con sus respectivas particiones $\{a_1, a_2, a_3, a_4, a_5\}$. Tenemos que:

$$s = 1 = a_4 + a_5 = a_1 + a_2 = a_1 + a_3 + a_4 = a_2 + a_3 + a_4$$

serían las combinaciones de todos los conjuntos de mínimo acceso. Como $u = (1, 0, 1)$, se tiene que:

$$a_1 = 1, a_2 = 0, a_3 = 0, a_4 = 0, a_5 = 1.$$

3.6.2. Estructura privilegiada

En ocasiones se desea compartir un secreto de forma que algunos usuarios tengan mayor privilegio de acceso a él que otros. Por ejemplo, supongamos que Ana es presidenta de un banco en el que Bob, Eva y David son cajeros. Se puede desear que Ana junto con cualquiera de los cajeros tenga acceso al dinero de la caja fuerte del banco que en nuestro ejemplo es considerado el secreto, pero que ni siquiera los tres cajeros juntos puedan acceder al dinero sin Ana. Para el ejemplo que acabamos de mencionar, la estructura de acceso consiste en los siguientes conjuntos de particiones:

$\{A, B\}, \{A, C\}$, y $\{A, D\}$, donde A, B, C y D denotan las particiones de Ana, Bob, Eva y David, respectivamente.

Explicaremos como dar forma a esta idea de estructura de accesos. Sea un código lineal C, de dimensión k, un subespacio vectorial de $\mathbb{F}_q^n$ y teniendo en cuenta la definición de palabra mínima, 2.3:

Proposición 3.24. *Dada un código lineal C $[n, k, d]$, cada palabra del código puede escribirse como una combinación lineal de las palabras clave mínimas que cubre.*

Demostración. Sea c una palabra clave no mínima, entonces c cubre una palabra clave mínima c_1 y, por lo tanto, hay una constante b_1 tal que la palabra clave $c - b_1c_1$ tiene un peso Hamming menor que c. Como c cubre a $c - b_1c_1$, se deduce que si $c - b_1c_1$ es 0 o una palabra clave mínima, la prueba está completa.
En caso contrario, $c - b_1c_1$ cubre una palabra clave mínima c_2, entonces hay una constante b_2 tal que la palabra clave cubierta por c, $c - b_1c_1 - b_2c_2$, tiene un peso Hamming menor que $c - b_1c_1$.

Este proceso se puede seguir continuamente, por lo tanto, todo código lineal C $[n, k, d]$ sin componentes que sean 0 en sus palabras clave, determina un esquema de intercambio de secretos de estructura de acceso (S, Γ). Sin pérdida de generalidad, el secreto se toma como el primer dígito c_0 de la palabra clave. A continuación, se calcula la palabra clave $c = [c_1, c_2, \ldots, c_n]$ y se toma la lista de particiones como $c_2, c_3, \ldots, c_n$ de modo que $S = n - 1$. □

Ejemplo 3.6. Ahora intentamos ejemplificar numéricamente el ejemplo que vimos al principio de la sección, en el que deseábamos que Ana junto con cualquiera de los cajeros tenga acceso al dinero de la caja fuerte del banco.

Dado un código lineal C de tipo $[5, 3, 3]$ sobre $\mathbb{F}_2$. Sea G la matriz generatriz sistemática del código C:

$$G = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \end{pmatrix}$$

Se puede comprobar que $[1, 1, 1, 0, 0]$, $[1, 1, 0, 1, 0]$, $[1, 1, 0, 0, 1]$, $[0, 0, 1, 1, 0]$, $[0, 0, 1, 0, 1]$ y $[0, 0, 0, 1, 1]$ son todas y sólo las palabras clave mínimas de este código.

Si la matriz G es una matriz de comprobación de paridad para el código C, es decir, una matriz generadora para el código dual $C^{\perp}$, entonces las palabras clave

mínimas con el primer componente 1 en este código dual son $[1,1,1,0,0]$, $[1,1,0,1,0]$ y $[1,1,0,0,1]$.

Según el corolario 3.21, la estructura de acceso está formada por los conjuntos de particiones $\{c_2, c_3\}$, $\{c_2, c_4\}$ y $\{c_2, c_5\}$. Si c_2, c_3, c_4 y c_5 son las particiones de Ana, Bob, Eva y David respectivamente, vemos que hemos realizado la estructura de acceso deseada para el banco en el que Ana es presidenta.

3.6.3. Esquemas en Rampa

Hasta el momento, hemos hablado solo de los esquemas de compartición de secretos perfectos. En estos esquemas como ya hemos comentado las particiones que poseen los participantes no revelan ninguna información sobre el secreto, sólo se encargan de determinar conjuntamente de manera única el secreto. Además, el tamaño de cada partición debe ser al menos tan grande como el tamaño del secreto.

En esta sección veremos los *ECS* no perfectos. En este tipo de *ECS* se tienen subconjuntos cuyas particiones conjuntas dan alguna información parcial del secreto. La ventaja de este tipo de esquemas de compartición de secretos es que es posible una alta tasa de información, es decir, el tamaño de la partición puede ser mucho menor que el tamaño del secreto. Esto implica que hay un camino en el medio entre lo conocido y lo desconocido.

Nota 3.25. En los *ECS* no perfectos si no llegamos al umbral de reconstrucción $r(S)$ puede que no se consiga determinar el secreto, pero si se puede tener información parcial sobre él.

3.6.3.1. Esquema de Shamir en Rampa

El esquema de Shamir visto en la sección 3.5 tiene como problema el alto grado de almacenamiento y costo computacional que tiene llegar a recuperar el secreto.
Ahora hablaremos del esquema de Shamir en rampa que nos permitirá reducir el tamaño de las particiones. Mientras que en el esquema de Shamir para distribuir ℓ elementos de $\mathbb{F}_q$ hay que enviar ℓ elementos de $\mathbb{F}_q$, en Shamir en rampa es suficiente con mandar un elemento de $\mathbb{F}_q$.

En la sección 3.5 se describe una versión que admite secretos en $\mathbb{F}_q$, ahora hablaremos de secretos en $\mathbb{F}_q^\ell$. Definamos $x_1, x_2, \ldots, x_n$ con n elementos distintos de cero de $\mathbb{F}_q$ y distintos dos a dos. El valor x_i está asociado con P_i, para todo i, $1 \leq i \leq n$.
Cuando el distribuidor D comparte el secreto $S \in \mathbb{F}_q^\ell$, toma al azar $t - \ell$ elementos de

$\mathbb{F}_q$. Sea el secreto $S = a = (a_0, \ldots, a_{t-\ell}) \in (\mathbb{F}_q)^{t-\ell}$. Si en el esquema tenemos n participantes se escogen $f_\ell, f_{\ell+1} \ldots, f_{k-1} \in \mathbb{F}_q$ aleatorios. Estos elementos corresponden a los coeficientes que forman el polinomio unidos. El polinomio requerido se muestra a continuación:

$$f(x) = a_0 + a_1 x + \ldots + a_{\ell-1} x^{\ell-1} + f_\ell x^\ell + \ldots + f_{k-1} x^{k-1} \in \mathbb{F}_q[x].$$

Las particiones son: $f(x_1), \ldots, f(x_n)$, con $x_i \in \mathbb{F}_q$ y $x_i \neq x_j$.

Mientras que en el esquema de Shamir el secreto $s \in \mathbb{F}_q$, en el esquema de Shamir en rampa el secreto $s \in \mathbb{F}_q^\ell$ para $\ell > 1$. La privacidad y la reconstrucción se deducen de la interpolación de Lagrange y la principal desventaja en ambos esquemas es que $q > n$, ya que sino no podríamos evaluar el polinomio.

Nota 3.26. En el esquema de Shamir en rampa se puede saber parte de la información del secreto, pero si se juntan $k - \ell$ participantes necesitan ℓ parámetros para reconstruir el secreto. Por lo tanto, el umbral de privacidad es igual a $t(S) = k - \ell$ y el umbral de reconstrucción $r(S) = k$.

Entonces si tenemos menos que $k - \ell$ particiones estamos seguros de que no se sabe nada sobre el secreto y nos encontramos con que la entropía es 1. Por el contrario, en caso de que tengamos k particiones, estamos seguro de que conocemos todo acerca del secreto y la entropía será 0.

Pero que se sabe acerca del secreto en el intervalo formado entre el umbral de privacidad y reconstrucción. En este caso igual sabemos nada, igual algo o igual todo, depende de cada caso en particular. Entonces si añadimos un participante al umbral de privacidad, podemos ir adquiriendo información acerca del secreto:

Si tenemos $k - \ell + 1$ participantes estos necesitarían $\ell - 1$ parámetros para reconstruir el secreto y nos iremos acercando cada vez más a una entropía igual a 0 y tendremos mayor probabilidad de conseguir reconstruir el secreto. Los atacantes tienen el conocimiento de que el secreto $s \in V \subset \mathbb{F}_q^\ell$, siendo V un espacio vectorial en la que la dim $V = \ell - 1$.
Realizando el proceso de sumar cada vez más participantes llegamos a que si se juntaran $k - 1$ participantes, necesitarían solo un parámetro para reconstruir el secreto, con $s \in V \subset \mathbb{F}_q^\ell$. Entonces $s \in V \subset \mathbb{F}_q^\ell$ con dim $V = 1$ y el secreto pertenece a una recta y tenemos q posibilidades de determinar el secreto. Al principio del proceso teníamos q^ℓ posibilidades de que la reconstrucción fuera satisfactoria, esto quiere decir que hemos ido ganando información acerca del secreto.

A continuación, ilustraremos un ejemplo numérico del esquema de Shamir en rampa.

Ejemplo 3.7. Suponemos que queremos compartir un dato $S = (a_0, a_1) = (S_0, S_1) = (2, 3) \in \mathbb{F}_5^2$ con el esquema de compartición de secretos de Shamir en rampa $(4, 4)$. El secreto S se desea dividir en 4 partes $(n = 4)$, de forma que cualquier subconjunto $(t = k = 4)$ sea suficiente para reconstruir el secreto.

Al azar se obtienen $t - l = 4 - 2$ valores, por ejemplo, 59 y 6. Se usan coeficientes que definen estos valores ($a_2 = 59 \mod 5 = 4$; $a_3 = 6 \mod 5 = 1$), mientras que el coeficiente que corresponde al secreto se define como $S = (a_0, a_1) = (S_0, S_1) = (2, 3)$. Se describe el polinomio en un orden especifico:

$$f(x) = S_0 + S_1 x + a_2 x^2 + a_3 x^3$$

$$f(x) = 2 + 3x + 4x^2 + x^3$$

Posteriormente dado que $n = 4$, se calculan cuatro puntos a partir del polinomio, es decir, evaluamos el polinomio cuando $x = 1, 2, \ldots, 4$ de manera que se obtienen puntos de la forma $(x_i, f(x_i))$.
$(1, f(1)); (2, f(2)); (3, f(3)); (4, f(4))$, evaluando estos valores en el polinomio, se obtienen los puntos:

$$(1, 0); (2, 2); (3, 4); (4, 2)$$

Reconstrucción: Para reconstruir el secreto bastará con cuatro puntos $(t = k = 4)$. Este ejemplo es el más robusto posible ya que se necesitan todas las particiones para recuperar el secreto:

$$(1, 0); (2, 2); (3, 4); (4, 2)$$

Usando la interpolación polinómica de Lagrange:

$$\ell_0 = \frac{(x-2)}{(1-2)} * \frac{(x-3)}{(1-3)} * \frac{(x-4)}{(1-4)} = (\frac{x^3 - 9x^2 + 26x - 24}{-6}) \mod 5 = \frac{x^3 - 4x^2 + x - 4}{4}$$

$$\ell_1 = \frac{(x-1)}{(2-1)} * \frac{(x-3)}{(2-3)} * \frac{(x-4)}{(2-4)} = (\frac{x^3 - 8x^2 + 19x - 12}{2}) \mod 5 = \frac{x^3 - 3x^2 + 4x - 2}{2}$$

$$\ell_2 = \frac{(x-1)}{(3-1)} * \frac{(x-2)}{(3-2)} * \frac{(x-4)}{(3-4)} = (\frac{x^3 - 7x^2 + 14x - 8}{-2}) \mod 5 = \frac{x^3 - 2x^2 + 4x - 3}{3}$$

$$\ell_3 = \frac{(x-1)}{(4-1)} * \frac{(x-2)}{(4-2)} * \frac{(x-3)}{(4-3)} = (\frac{x^3 - 6x^2 + 11x - 6}{6}) \mod 5 = x^3 - x^2 + x - 1$$

Por lo tanto,

$$f(x) = \sum_{j=0}^{k} y_j * \ell_j(x)$$

$$= 0 * (\frac{x^3 - 4x^2 + x - 4}{4}) + 2 * (\frac{x^3 - 3x^2 + 4x - 2}{2}) + 4 * (\frac{x^3 - 2x^2 + 4x - 3}{3}) +$$

$$+2 * (x^3 - x^2 + x - 1) = x^3 - 3x^2 + 4x - 2 + 8x^3 - x^2 + 2x - 4 + 2x^3 - 2x^2 + 2x - 2 =$$

$$= 11x^3 - 6x^2 + 8x - 8 = x^3 + 4x^2 + 3x + 2$$

Entonces hemos llegado a la reconstrucción del secreto original es $S = (2, 3)$.

Hemos visto tanto para el esquema de Shamir como para el esquema de Shamir en rampa los ejemplos cuando se tiene pleno conocimiento sobre el secreto. A continuación, veremos que sabemos en ambos casos si nos encontramos entre el intervalo del umbral de privacidad y de reconstrucción.

Ejemplo 3.8. Esquema de Shamir

Sea el esquema de Shamir con $t = 3$, $n = 6$ en el cuerpo $\mathbb{F}_q$ de tal manera que:

$$f(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3,$$

entonces se necesitarán 4 particiones para reconstruir el secreto $S = a_0$, $r(S) = 4$.
Imaginaremos que sólo tenemos el conocimiento de tres particiones:

$$(1, f(1)); (2, f(2); (3, f(3)),$$

necesitaríamos de una última partición $(4, \lambda)$ para realizar la interpolación de Vandermonde $f(x)[\lambda]$ y llegar a la reconstrucción del secreto. Entonces $(4, \lambda)$ es nuestra partición desconocida con $\lambda \in \mathbb{F}_q$ y el secreto $s \in \mathbb{F}_q$.
En este caso o no conocemos **nada** sobre el secreto o lo conocemos **todo**.

Esquema de Shamir en rampa

Sea el esquema de Shamir en rampa con $t = 3$, $n = 6$ en el cuerpo $\mathbb{F}_q^\ell$ de tal manera que:

Sea el secreto $S = (S_0, S_1) \in \mathbb{F}_q^\ell$, entonces tenemos:

$$f(x) = S_0 + S_1 x + a_2 x^2 + a_3 x^3.$$

Al igual que el ejemplo anterior se necesitan 4 particiones para reconstruir el polinomio y llegar al secreto S. Si tenemos las particiones: $(1, f(1)); (2, f(2)); (3, f(3)); (4, \lambda)$ y realizamos la interpolación de Vandermonde de tal manera que nos queda:

$$(f(\lambda))(x) = \lambda + (2\lambda)x + 3\lambda x^2 + \lambda x^3$$

como $\lambda \in \mathbb{F}_q$ y $S = (\lambda, 2\lambda)$, es decir, $S \in \langle (1, 2) \rangle$ y solo existen ℓ posibles soluciones. Si particularizamos en este caso $S \in \mathbb{F}_5^2$ tenemos 5^2 posibles secretos:

$$(1, 2); (2, 4); (3, 1); (4, 3); (0, 0)$$

.

Ahora iremos un paso más allá si tenemos las siguientes particiones: $(1, f(1)); (2, f(2)); (3, \lambda), (4, \mu)$.

Solo conocemos dos particiones y en este caso tenemos $(\lambda, \mu) \in \mathbb{F}_5^2$ y no tenemos ninguna información sobre el secreto. Entonces si nos encontramos $\mathbb{F}_5^2$ hemos obtenido los siguientes resultados con un esquema de Shamir en rampa con $t = 3$ y $l = 2$:

- **1 partición**: No se sabe nada acerca del secreto
- **2 particiones**: No se sabe nada acerca del secreto. Encontramos el umbral de privacidad, $t(S) = k - l = 4 - 2 = 2$.
- **3 particiones**: Zona gris se sabe información sobre el secreto. Es una recta en este caso.
- **4 particiones**: Se sabe todo y se recupera S, como visualizamos en el ejemplo 3.7. Encontramos el umbral de reconstrucción, $r(S) = k = 4$.

3.6.3.2. Extensión a la idea de Massey

El esquema de Shamir en rampa y el esquema de Shamir tienen la desventaja de no poder usar un cuerpo finito arbitrario. La idea de Massey desarrollada en la sección 3.6.1, subsana este problema. La extensión de esta idea facilitará una mejor tasa de información.
Mientras que en el método de Massey para compartir n bits es necesario enviar n bits, ahora con la extensión de Massey para distribuir n bits es suficiente con mandar uno solo. Esto provoca que este tipo de esquemas sea más eficiente para ficheros grandes.

Sea C un código $[n + \ell, k, d]$ sobre un cuerpo finito $\mathbb{F}$. Sea ℓ un número entero no negativo tal que $\ell < d^{\perp}$ y un $s \in \mathbb{F}^{\ell}$. Si tomamos una palabra c del código C, tal que

$c = (s_1, \ldots, s_{\ell-1}, c_1, \ldots, c_n) \in C$. Este es un esquema en rampa lineal que tiene como umbral de privacidad, $t(S) \geq d^{\perp} - \ell - 1$ y de reconstrucción $r(S) \leq n + 1 - d + 1$.

La reconstrucción se puede demostrar dado que si existieran dos palabras clave del código C estas coincidirían en $n + l - d + 1$ posiciones, la diferencia daría una palabra del código C con peso de Hamming menor que d.
El umbral de privacidad se puede demostrar tomando la matriz generatriz G del código C, en la que cualquier colección de $m < d^{\perp}$ filas son linealmente independientes. Entonces las columnas correspondientes abarcan $\mathbb{F}_m$, ya que el código se genera por columnas. Por lo tanto, para cada $j \in \{0, \ldots, \ell - 1\}$ y para cada $A \subset \{1, \ldots, n\}$ con $\mid A \mid \leq d^{\perp} - \ell - 1$ existe una palabra clave c tal que $c'_j = 1$ y $c'_i = 0$ para todos $i \in \{0, \ldots, \ell - 1\} \setminus \{j\}$ y $c_u = 0$ para todo $u \in A$, entonces el umbral de privacidad es $t(S) \leq d^{\perp} - \ell - 1$.

Nota 3.27. Este tipo de esquema de compartición de secretos cumple la idea de Massey descrita anteriormente para $\ell = 1$.

Nota 3.28. Una forma equivalente de presentar el esquema de Shamir en rampa, es gracias a la extensión de la idea de Massey aplicada a los códigos Reed Salomon:

Sea $s \in \mathbb{F}_q^{\ell}$, tal que $1 \leq t + \ell - 1 \leq n$ y $n + \ell \leq q$. Fijamos $\alpha_1, \ldots, \alpha_n, \beta_1, \ldots, \beta_\ell$ elementos distintos dos a dos en $\mathbb{F}_q$. Denotemos por $\mathbb{F}_q[X]_{\leq t+\ell-1}$ el conjunto de polinomios de grado menor que $t + \ell - 1$ y con coeficientes en $\mathbb{F}_q$. Para compartir un secreto $s \in \mathbb{F}_q^{\ell}$ entre n partes con este esquema, se elige un polinomio f en $\mathbb{F}_q[X]_{\leq t+\ell-1}$ al azar pero claro con la condición de que $f(\beta_j) = s_j$ para $j = 1, \ldots, \ell$. Entonces la partición $i-$ésima se define como la evaluación $f(\alpha_i) \in \mathbb{F}_q$. Este esquema de compartición de secretos se construye como $S^{(\ell)}(C)$ a partir del código Reed Solomon:

$$C = \{(f(\beta_1), \ldots, f(\beta_\ell), f(\alpha_1), \ldots, f(\alpha_n)) : f \in \mathbb{F}_q[X]_{\leq t+\ell-1}\}.$$

Ejemplo 3.9. Sea el código lineal C $[7, 3, 3]$ sobre $\mathbb{F}_2^2$, que tiene como umbral de reconstrucción $r(S) \geq 5 + 1 - 3 + 2 = 5$ y con matriz generatriz:

$$G = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \end{pmatrix}$$

La aplicación lineal asociada a esta matriz generatriz viene definida de la siguiente forma:

$$f : \mathbb{F}_2^3 \longrightarrow \mathbb{F}_2^7$$
$$a \longmapsto Ha^t$$

$$a = (a_1, a_2, a_3) \longmapsto c = (a_1, a_2, a_3, a_3, a_2 + a_3, a_1 + a_2, a_2)$$

y matriz de control:

$$H = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Consideremos el esquema $C^{\perp}$, si elegimos $u = (1, 0, 1)$. Como H es una matriz generatriz para $C^{\perp}$, entonces $s_1 = (1, 0, 1) \cdot (1, 0, 0) = 1$ y $s_2 = (1, 0, 1) \cdot (0, 1, 1) = 1$. Entonces $S = (s_1, s_2) = (1, 1)$.

Capítulo 4

Computación Segura entre múltiples partes (MPC)

4.1. Introducción

La computación segura entre múltiples partes (MPC) es un concepto de seguridad criptográfica que puede ayudarse de los esquemas de compartición de secretos (ECS), permitiendo que varias partes que no confían entre si puedan calcular conjuntamente resultados. Los resultados (output) si pueden ser puestos en común, pero no las aportaciones de las demás partes (input). El objetivo es calcular de forma segura un problema acordado, en el que cooperan n participantes. Cada participante realiza una tarea de cálculo basada en los datos privados que cada uno proporciona y ningún otro participante puede calcular. En este punto se han seguido los siguientes artículos de referencia: [7], [14], [20], [29] y [30].

El concepto *MPC* fue introducido por Andrew C. Yao. en 1982 y ejemplificado por el "Problema de los millonarios [Yao]": "Dos millonarios quieren saber quién es más rico; sin embargo, no quieren revelar cualquier información adicional sobre la riqueza de cada uno. ¿Cómo pueden llevar a cabo una conversación así? ". Se quiere saber quien es el más rico de los dos millonarios, sin descubrir en ningún momento el dinero que tiene cada uno.

Este protocolo no usa medios convencionales, en la que partes comunes podrían acumular información convencional, sino que existe un participante de confianza que se encarga de recopilar las informaciones secretas de los demás participantes. Este juez confidencial calcula el resultado de todas las entradas y revela únicamente una salida. Pero no siempre tiene que existir un participante de confianza, y mediante el uso computacional entre distintas partes pueden lograr el mismo nivel de confianza.

Si no existe una figura de confianza tenemos que tener en cuenta que la capacidad para validar las entradas por el protocolo *MPC* se basa en la honestidad de los participantes.

4.2. Corrupción de la información (Activa y Pasiva)

Es importante saber que no se puede confiar en todos los participantes en el protocolo *MPC*, dado que algunos participantes pueden transformar la información o intentar interrumpir el proceso. Si en el cálculo del protocolo existen n partes involucradas, algunas pueden comportarse sin entorpecer el protocolo (partes honestas) y algunas pueden querer romper la privacidad de las entradas de las otras partes (partes corruptas).
Esto da lugar a dos tipos de corrupción de la información:

- **Corrupción Pasiva**: Los participantes corruptos siguen el protocolo pero comparten la información. Cuando la corrupción es pasiva se obtiene acceso a información que debería ser inaccesible. En este caso podemos poner como ejemplo, un votante que pregunta y contrasta su voto con el de otro votante obteniendo de esa manera más información sobre el protocolo. Esta corrupción se puede prevenir con la utilización de los (ECS), porque el esquema garantiza que si t participantes no son honestos, entonces no podrán obtener ninguna información. En el presente documento solo trataremos protocolos resistentes frete a la corrupción pasiva.

- **Corrupción Activa**: Los participantes corruptos se comportan de forma arbitraria y no siguen el protocolo. En la corrupción activa el adversario intenta enviar datos, para que las partes se desvíen del protocolo. Esto puede ser resuelto si la validez de las particiones es verificada por cualquier parte que interviene en la comunicación.

Nota 4.1. El protocolo no sabe qué partes son corruptas y que partes son honestas. Por tanto, el objetivo será asegurarse de que las partes corruptas no se enteren de las partes honestas. Cualquiera de las partes puede ser corrupta, por eso en primera instancia no se puede confiar en ninguna. Además, no se puede asegurar completamente que un adversario deshonesto no pueda corromper el protocolo, pero si intentaremos que la probabilidad de que esto se produzca sea aproximadamente nula.

Número de partes corruptas.

Los adversarios se pueden visualizar como una entidad maestra que controla un subconjunto de las partes, cuantas más partes pueda corromper el adversario, mayor

poder tendrá y le resultará más sencillo romper la privacidad de los demás participantes honestos.

Nota 4.2. El número de partes corruptas y su comportamiento son los parámetros más relevantes para tener consciencia del poder que puede tomar el adversario.

Si denotamos a n como el número total de particiones, y t la mayor cantidad de partes que pueden estar corrompidas, cuya identidad no es conocida. Por tanto, tenemos diversas opciones que se pueden dar en el protocolo:

- $t < n$. Es el escenario más fuerte posible, mantiene la privacidad del protocolo. Se conoce este escenario como mayoría deshonesta.

- $t < n/2$. Se conoce este escenario como mayoría honesta. El adversario solo puede corromper a una minoría de las partes. En este caso, la privacidad del protocolo se rompe tan pronto como el adversario corrompa a la mitad o más de las partes. La mayoría honesta permite protocolos más eficientes con garantías de seguridad más sólidas, asumiendo que las partes tienen acceso a los canales de comunicación.

- $t < n/3$. El adversario solo puede corromper a un tercio de las partes. Este escenario garantiza una seguridad aún mejor y una mayor eficiencia. Además, puede tener sentido cuando n es muy grande, asumiendo que los canales son privados y autenticados.

4.3. Protocolo MPC

La pregunta que nos tenemos que hacer cuando pensamos en el protocolo *MPC*, es la siguiente: ¿cómo es posible que las partes calculen una función sin filtrar los datos y preservando la privacidad?.

Consideremos n partes $P_1, \ldots, P_n$ que tienen como entradas privadas $x_1, \ldots, x_n$ (input), que desconfían entre si. Si se quiere calcular una función $y = f(x_1, \ldots, x_n)$ solo filtrando su salida (output), esta debe mantener siempre la privacidad de las entradas (input). El objetivo de *MPC* es permitir esto de forma que solamente se precise la comunicación entre las partes. Para un protocolo *MPC* seguro tendremos que garantizar que aunque existan participantes deshonestos nos sea posible calcular correctamente $f(x_1, x_2, \ldots, x_n)$. Por lo tanto, debe de cumplirse que se pueda calcular $f(x_1, x_2, \ldots, x_n)$ solo filtrando su salida (output), incluso si un pequeño número t de los participantes se corrompe y hace trampas.

Ejemplo 4.1. Ahora mostraremos de manera general como funciona el protocolo *MPC*.
Consideremos que trabajamos sobre $\mathbb{F}_5$. Existen dos participantes Ana y Bob que tienen como entrada x e y respectivamente y quieren calcula la función $z = f(x, y) = (x - y) * (x + y) \mod 5$. Los valores x e y se encuentran en el conjunto $\{0, 1, \cdots, 4\}$.
Los participantes no tienen ningún conocimiento de los datos, pero si se juntan ambos podrían reconstruir z. Así la privacidad está garantizada, siempre que un adversario no obtenga las suficientes particiones para reconstruir el secreto.

El objetivo es realizar esto de manera que Bob no aprenda la entrada x de Ana, y a su vez Ana no aprenda la entrada y de Bob.
Cualquier participante puede notar que $z = x^2 - y^2 \mod 5$. Como Bob sabe y, aprender z le permite aprender $x^2 \mod 5$, que filtra bastante sobre x, lo mismo ocurriría con Ana y la entrada y.

Tenemos que tener en cuenta en primer lugar, que esta operación no proporciona x en su totalidad pero si sabe que cada cuadrado módulo 5 tiene dos raíces, una en el conjunto $\{1, 2\}$ y otra en el $\{3, 4\}$. En segundo lugar, no nos importa lo que filtre z sobre las aportaciones de las partes, dado que la definición de seguridad para *MPC* requiere que el adversario no aprende nada acerca de las entradas de las partes honestas, más allá de lo que ya se filtró desde la salida.

Ana y Bob quieren reconstruir z.
Sea la entrada de Ana $x = 2$. Ana muestra un número aleatorio en el conjunto $\{0, \ldots, 4\}$, por ejemplo 3, y lo resta ($\mod 5$) de su entrada para obtener 4. Ana envía el número aleatorio 3 a Bob y se queda con el número 4, ya que si calculamos $3 + 4 \mod 5 = 2 = x$. Bob solo tiene conocimiento del número aleatorio que recibió, por lo que no puede saber cual es la entrada de Ana. Bob realiza el mismo procedimiento con su entrada $y = 1$, de esta manera se garantiza que ningún participante tendrá conocimiento sobre la entrada del otro.

Necesitamos diseñar un protocolo que permita calcular $z = (x - y) * (x + y) \mod 5$ y que no filtre nada más. Si observamos la función z parece que necesitamos conocer las entradas x e y para calcularla. Las operaciones módulo 5 que se realizan en este protocolo son:

1. Calcular $u = x + y$ y $v = x - y$.

2. Multiplicar $z = u * v$ y devuelve z.

Entonces este protocolo no filtrará nada sobre las entradas x e y siempre que utilicemos un cifrado para estas entradas (input) , que nos permita operar tanto con **suma, resta y multiplicación** estos valores cifrados.

Cifrar los datos permitirá transformarlos en una representación oculta y aún así realizar las operaciones descritas anteriormente sin revelar ningún dato. Si ciframos las entradas y las llamamos $Enc(x)$ y $Enc(y)$, entonces se pueden realizar los pasos anteriores, sin revelar ningún tipo de información.

1. Ana cifra x como $Enc(x)$ y Bob cifra y como $Enc(y)$.

2. Calculamos $Enc(u) = Enc(x) + Enc(y)$ y $Enc(v) = Enc(x) - Enc(y)$.

3. Multiplicamos $Enc(z) = Enc(u) * Enc(v)$.

4. Desciframos z y devolvemos z.

El objetivo de este diseño es calcular una función de forma segura, sin que se filtre nada sobre los valores x e y mientras se están procesando (input). Aunque Ana y Bob en este ejemplo aprenden z (output), intentaremos ir un paso más allá.
La utilización de los (ECS) (capitulo 3) en este protocolo nos permite en lugar de obtener una representación oculta de los datos, distribuir los datos entre Ana y Bob de tal manera que ninguno de ellos pueda saber realmente cuáles son los datos. Es decir, nos puede servir para que las partes puedan comunicarse y ayudarse mutuamente a realizar el cálculo sin filtrar las entradas (input) y obteniendo sólo como resultado final las salidas (output).

4.3.1. Funciones y polinomios

Antes de explicar el protocolo MPC unido a los esquemas ECS, veremos como cualquier función sobre cuerpos finitos es un polinomio. Para ello utilizaremos la interpolación de Vandermonde vista en el capítulo 3.5.
La sustitución de los puntos en los polinomios deseados produce un sistema de ecuaciones lineales en los coeficientes del polinomio. Este sistema se puede resolver mediante la Eliminación Gaussiana.

Tenemos una serie de puntos $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$ y queremos encontrar un polinomio que pase por todos ellos. Entonces si tenemos n puntos podemos plantear n ecuaciones y necesitamos n incógnitas que serán los coeficientes del polinomio. Por

lo tanto:

$$P(x) = a_0 + a_1 x + \ldots + a_{n-1} x^{n-1},$$

el polinomio tiene como máximo grado $n - 1$.

Entonces si tenemos una función f definida tal que:

$$f : \mathbb{F}_q \longrightarrow \mathbb{F}_q$$

Por la interpolación de Vandermonde podemos llegar a un polinomio P de tal manera que el grado de $P \leq q - 1$. Además, se pueden interpolar q valores $\{(a, f(a)) \mid a \in \mathbb{F}_q\}$, tal que:

$$f(a) = P(a) = b, \ \ \forall a \in \mathbb{F}_q.$$

Una vez visto como cualquier función sobre cuerpos finitos es un polinomio. A continuación, nos centraremos en como realizar a partir de un polinomio la computacion multiparte con sumas y productos.

4.3.2. MPC con los esquemas de compartición de secretos

El protocolo *MPC* es totalmente seguro siempre que garanticemos que un adversario no puede corromper a más de una minoría de las partes. Para conseguir que el protocolo se mantenga totalmente firme frente adversarios deshonestos, es necesario utilizar diferentes métodos para evitar todo tipo de ataques.
Nosotros trataremos en este documento, métodos que hagan resistente al protocolo *MPC* frente a la corrupción pasiva, en la que todas las partes se comportan de acuerdo al protocolo y solo exista fuga de información. El método que nos permitirá tener un protocolo *MPC* seguro frente a la seguridad pasiva son los esquemas de compartición de secretos.

Los *ECS* permiten preservar la información de las entradas (input) totalmente ocultas, sólo filtrando las salidas (output). Pero como comentamos en el ejemplo 4.1, antes de utilizar un método criptográfico para preservar la información en el protocolo *MPC*, es necesario observar si este permite operar con los datos sin presentar ningún inconveniente al respecto.

La utilización de los *ECS* en el protocolo *MPC* permite que la operación de adición no presente ningún problema como veremos. La multiplicación de particiones por el contrario si será mucho más compleja de ejecutar.

Los esquemas lineales de compartición de secretos cumplen la siguiente propiedad que es de importancia clave para que la adicción de particiones no presente ningún

problema:
Suponemos varios secretos $s^{(1)}, s^{(2)}, \ldots, s^{(\ell)}$ que se comparten con S, donde la partición para $s^{(j)}$ es $(a_1^{(j)}, \ldots, a_n^{(j)})$. Entonces dado $\alpha^{(1)}, \ldots, \alpha^{(\ell)} \in \mathbb{F}_q$, las combinaciones lineales, $(\sum_{j=1}^{\ell} \alpha^{(j)} a_i^{(j)})_{i=1,\ldots,n}$, son particiones de $\sum_{j=1}^{\ell} \alpha^{(j)} s^{(j)}$ en el mismo esquema S.

Por este motivo la suma de secretos es igual a la suma de las particiones, o incluso una combinación lineal de secretos sera una combinación lineal de particiones.

Sería útil que se mantuviera una propiedad similar a la que acabamos de mencionar para la multiplicación de particiones, es decir, dadas las particiones s y s', los productos de las coordenadas de las particiones formaran una partición en el mismo esquema, pero desafortunadamente esto no se puede deducir de la linealidad de un esquema de compartición de secretos.

Sea $f : \mathbb{F}^n \longrightarrow \mathbb{F}$ una función lineal. Por linealidad, el conjunto de valores resultante es un reparto de $f(x_1, \ldots, x_n)$. Entonces los participantes pueden transmitir las particiones y reconstruir la salida (output).
Sería deseable poder trasladar esta idea a cualquier función f. Pero si tenemos un $LSSS\ S := S^{(1)}(C)$, y cada participante multiplica sus particiones, el conjunto de valores resultante no es una partición en el mismo esquema $S^{(1)}(C)$, sino en $S^{(1)}(C^2)$.

Definición 4.3. Potencia $m-$ésima de un código lineal. Sea $C \subseteq \mathbb{F}_q^n$ un código lineal sobre $\mathbb{F}_q$, con $d > 0$ entero, tal que:

$$C^{*m} := \mathbb{F}_q\langle\{c^{(1)} * c^{(2)} * \ldots * c^{(m)} : (c^{(1)}, c^{(2)}, \ldots, c^{(m)}) \in C^m\}\rangle$$

Definición 4.4. Un $LSSS\ S^{(\ell)}(C)$ ideal, tiene una reconstrucción del producto r, si para cada conjunto $A \subseteq I$ de tamaño al menos r, existe alguna función lineal ρ_A:

$$\mathbb{F}_q^{|A|} \longrightarrow \mathbb{F}_q^{\ell}$$

tal que para cada partición $(c_0, c_1, \ldots, c_n), (a_0, a_1, \ldots, a_n) \in C$, se sostiene que $\rho_A(c_A * a_A) = c_0 * a_0$.

Esta definición explica como el producto de dos secretos se reconstruirá a partir del conjunto de los productos de las particiones de los participantes en A. Pero encontrar la función ρ_A no es tan sencillo.
Nosotros intentaremos construir un protocolo MPC seguro contra un adversario pasivo que corrompe a t participantes a partir de cualquier $LSSS$ con privacidad t.
La idea en el caso pasivo es la siguiente:

Sea $f : \mathbb{F}^n \longrightarrow \mathbb{F}$ una función lineal. Cada entrada se encuentra en $\mathbb{F}$ y $S := S^{(1)}(C)$ es un $LSSS$ sobre $\mathbb{F}$ con privacidad t. Los participantes comparten en secreto su entrada x_i (input) con S. Después cada participante aplica f a las particiones recibidas.

Si permitimos la interacción entre los participantes llegaremos a una mejor alternativa.
Suponemos que $S^{(1)}(C)$ es multiplicativo, entonces por definición, existe una función lineal tal que $ss' = \rho(c_1c'_1, \ldots, c_nc'_n)$ donde c_i, c'_i son las particiones de s' y s respectivamente. El siguiente protocolo permite que cada participante i comparta el producto $c_ic'_i$ de sus particiones con $S^{(1)}(C)$. Entonces los participantes tienen particiones de $c_1c'_1, c_2c'_2, \ldots, c_nc'_n$ y pueden calcular localmente las particiones de $ss' = \rho(c_1c'_1, \ldots, c_nc'_n)$ ya que ρ es lineal.

Esta construcción es segura contra un adversario pasivo que corrompe a lo sumo t partes si el esquema tiene como umbral de privacidad t, dado que cualquier función puede ser calculada de forma segura por n participantes en esta situación.

En el siguiente punto veremos como podemos aplicar la multiplicación de particiones en un ejemplo con el esquema de Shamir $2t < n$.

4.3.2.1. MPC con el esquema de Shamir

El método que utilizaremos unido al protocolo MPC para mantener la seguridad frente a la corrupción pasiva, es el esquema de Shamir. Esta unión la trataremos en la siguiente sección con el protocolo de votación electrónica, pero antes veremos como explicar numéricamente el protocolo MPC con la multiplicación.

Suponemos que distintos participantes tienen particiones tanto de $[x]$ como de $[y]$. Entonces cada participante P_i obtendrá los polinomios $f(i)$ y $g(i)$ como las particiones de x e y, de tal manera que $f(0) = x$ y $g(0) = y$.

Cada participante P_i puede realizar la operación aditiva $f(i) + g(i)$ para obtener particiones de $x + y$. Esto es posible ya que si tomamos un polinomio $q(X) = f(X) + g(X)$ y lo evaluamos en el valor i. Este polinomio tiene grado como máximo t y cumple que $q(0) = x + y$, pero si multiplicamos las particiones de P_i, entonces $h(X)$ es el polinomio tal que $f(X) * g(X)$ y $h(0) = f(0) * g(0) = x * y$. Por lo tanto, el grado de $h(x)$ es lo que falla, ya que f y g tienen un grado como máximo t, lo que implica que h tiene un grado como máximo $2t$.

Las partes pueden obtener localmente particiones $x * y$, basándose en el esquema de compartición de secretos de Shamir con grado $2t$ en lugar de t. Entonces si

tenemos $2t + 1$ partes estas podrán reconstruir el secreto, sin embargo esto no es una simple multiplicación, sino que tenemos que conseguir las particiones $x * y$ bajo el esquema de compartición de secretos de Shamir de grado t. Esto será posible gracias a la linealidad y multiplicidad de este esquema, ya que la reconstrucción a partir de $2t + 1$ particiones es "lineal". Es decir, existen constantes $a_1, \cdots, a_n$ tal que $h(0) = a_1 * h(1) + \cdots + a_n * h(n)$ y nos permitirán obtener las particiones $[x * y]$ del grado correcto, dado que $h(0) = x * y$.

Ejemplo 4.2. En este ejemplo veremos como se puede llevar a cabo la computación multiparte con productos a partir de un polinomio y utilizando el esquema de Shamir:

Respecto a lo comentado anteriormente la función ρ para el caso de Shamir es trivial y siempre podremos calcular la multiplicación con $t < \frac{n}{2}$.
Para ejemplificar lo descrito calculamos el producto de los secretos $S_1 \cdot S_2$ pertenecientes al cuerpo $\mathbb{F}_{11}$. Para S_1 tomaremos los datos del Ejemplo 3.4 con $t = 2$, descritos en la sección Esquema de Shamir. Entonces si S_1 y S_2 se esconden con Shamir de grado t, la multiplicación de ambos nos dará un esquema de Shamir de grado $2t$:

$$f_1(x) = S_1 + a_1 x + a_2 x^2.$$
$$f_2(x) = S_2 + b_1 x + b_2 x^2.$$
$$h(x) = f_1 \cdot f_2 = S_1 \cdot S_2 + (S_1 \cdot b_1 + S_2 \cdot a_1)x^1 + (a_1 \cdot b_1 + S_1 \cdot b_2 + S_2 \cdot a_2)x^2 +$$
$$+(a_1 \cdot b_2 + a_2 \cdot b_1)x^3 + (a_2 \cdot b_2)x^4$$

Nos encontramos en el cuerpo $\mathbb{F}_q = 11$ con $n = 6$. Sean $S_1 = 10$, $a_1 = 10$, $a_2 = 1$, $b_1 = 5$, $b_2 = 2$ y $S_2 = 6$, entonces tenemos:

$$f_1(x) = 10 + 10x + x^2$$
$$f_2(x) = 6 + 5x + 2x^2$$

Por lo tanto, $h(x) = f_1(x) \cdot f_2(x) = 5 + 10x^2 + 3x^3 + 2x^4$.

Las particiones del primer esquema de Shamir $(x_i, f_1(x_i))$ son:

$$(1, 10); (2, 1); (3, 5); (4, 0); (5, 8); (6, 7)$$

y para el segundo $(x_i, f_2(x_i))$:

$$(1, 2); (2, 2); (3, 6); (4, 3); (5, 4); (6, 9)$$

Si realizamos la multiplicación de particiones $(x_i, f_1(x_i) \cdot f_2(x_i))$ tenemos:

$$(1,9); (2,2); (3,8); (4,0); (5,10); (6,8)$$

Para reconstruir el secreto es necesario $2t + 1$ particiones, es decir, 5 particiones. Consideremos 5 puntos cualesquiera:

$$(1,9); (2,2); (3,8); (5,10); (6,8)$$

Usando la interpolación polinómica de Lagrange para llegar a la reconstrucción del secreto:

$$\ell_0 = \frac{(x-2)}{(1-2)} * \frac{(x-3)}{(1-3)} * \frac{(x-5)}{(1-5)} * \frac{(x-6)}{(1-6)} = \frac{x^4 - 16x^3 + 91x^2 - 216x + 180}{40} \quad \text{mód } 11 =$$

$$\frac{x^4 - 5x^3 + 3x^2 - 7x + 4}{7}$$

$$\ell_1 = \frac{(x-1)}{(2-1)} * \frac{(x-3)}{(2-3)} * \frac{(x-5)}{(2-5)} * \frac{(x-6)}{(2-6)} = \frac{x^4 - 15x^3 - 77x^2 - 153x + 90}{-12} \quad \text{mód } 11 =$$

$$\frac{x^4 - 4x^3 - 10x + 2}{10}$$

$$\ell_2 = \frac{(x-1)}{(3-1)} * \frac{(x-2)}{(3-2)} * \frac{(x-5)}{(3-5)} * \frac{(x-6)}{(3-6)} = \frac{x^4 - 14x^3 + 65x^2 - 112x + 60}{12} \quad \text{mód } 11 =$$

$$x^4 - 3x^3 + 10x^2 - 2x + 5$$

$$\ell_3 = \frac{(x-5)}{(5-1)} * \frac{(x-2)}{(5-2)} * \frac{(x-3)}{(5-3)} * \frac{(x-6)}{(5-6)} = \frac{x^4 - 12x^3 + 47x^2 - 72x + 36}{-24} \quad \text{mód } 11 =$$

$$\frac{x^4 - x^3 + 3x^2 - 6x + 3}{9}$$

$$\ell_4 = \frac{(x-1)}{(6-1)} * \frac{(x-2)}{(6-2)} * \frac{(x-3)}{(6-3)} * \frac{(x-5)}{(6-5)} = \frac{x^4 - 11x^3 + 41x^2 - 61x + 30}{60} \quad \text{mód } 11 =$$

$$\frac{x^4 + 8x^2 - 6x + 8}{5}$$

Por lo tanto,

$$f(x) = \sum_{j=0}^{k} y_j * \ell_j(x) = 9 * (\frac{x^4 - 5x^3 + 3x^2 - 7x + 4}{7}) + 2 * (\frac{x^4 - 4x^3 - 10x + 2}{10}) +$$

$$8*(x^4-3x^3+10x^2-2x+5)+10*(\frac{x^4-x^3+3x^2-6x+3}{9})+8*(\frac{x^4+8x^2-6x+8}{5})=$$

$$=\frac{9x^4-x^3+5x^2-8x+3}{7}+\frac{2x^4-8x^3-9x+4}{10}+8x^4-2x^3-3x^2-5x+7+$$

$$\frac{10x^4-10x^3+8x^2-5x+8}{9}+\frac{8x^4+9x^2-4x+9}{5}=$$

$$7684x^4-2554x^3+4034x^2-5291x+6626=\frac{6x^4-2x^3+8x^2+4}{3}=$$

$$2x^4-\frac{2}{3}x^3+\frac{8}{3}x^2+\frac{4}{3}=2x^4+3x^3+10x^2+5$$

Y como queríamos demostrar es posible llegar a concluir con el resultado:

$$S_1 \cdot S_2 = 5.$$

A continuación, nos preguntaremos si esto funciona con todos los esquemas que hemos visto en la sección de compartición de secretos 3.

Esquema de Shamir en Rampa

Veremos que pasa si tomamos dos esquemas de Shamir en rampa para realizar la multiplicación anterior $S_1 \cdot S_2$.
Sean los secretos (S_1, S_2) y (S_1', S_2') pertenecientes a $\mathbb{F}_q^l$. Entonces

$$f_1(x) = S_1 + S_2 x + a_t x^t.$$

$$f_2(x) = S_1' + S_2' x + b_t x^t.$$

$$h(x) = f_1 \cdot f_2 = S_1 \cdot S_1' + (S_1 \cdot S_2' + S_2 \cdot S_1'))x.....$$

En este caso sería imposible la reconstrucción del secreto, dado que tras la multiplicación de particiones llegaríamos a un polinomio h en el que el secreto se encuentra tanto en el termino independiente como acompañando a la variable x y se compone con multiplicaciones cruzadas de tal manera que la única información que tenemos sobre el secreto es:

$$(S_1 \cdot S_1', S_1 \cdot S_2' + S_2 \cdot S_1')$$

Lo cual sería insuficiente para que podamos reconstruir el secreto $S_1 \cdot S_2$.

Idea de Massey y extensión a la idea de Massey:

Ahora nos preguntamos que sucede si vemos este mismo ejemplo con el método de Massey o la extensión a la idea de Massey. Pues en este caso mostraremos que si se puede realizar al igual que sucedía en el esquema de Shamir, dado que la funión ρ en este caso tampoco añade complejidad.

Utilizando la extensión a la idea de Massey para ello tomamos $(s_1, \ldots, s_l, c_1 \ldots c_n) \in C$ y $(s'_1, \ldots, s'_l, c'_1 \ldots c'_n) \in C$. La multiplicación de las particiones por la definición 4.3 sería:

$$(s_1 \cdot s'_1, \ldots, s_l \cdot s'_l, c_1 \cdot c'_1, \ldots, c_n \cdot c'_n) \in C^{*2},$$

calcular el código de C^{*2} es bastante laborioso, pero factible:

$$C^{*2} := \mathbb{F}_q\langle\{c * c' \mid c \in C, c' \in C\}\rangle = \langle v_i \neq w_j \mid \forall i, j\rangle,$$

con v_i base de c y w_j base de c'.

A continuación, tomemos la idea de Massey aplicada a dos códigos Reed-Salomon $C\ [n, k.d]$, la multiplicación generará un código Reed-Salomon $C^{*2}\ [n, 2k-1, d]$. Este ejemplo es similar a la multiplicación vista con dos esquemas de Shamir de grado t:

Suponemos que nos encontramos al igual que antes en $\mathbb{F}_{11}$ y tenemos dos códigos Reed-Salomon $C\ [6, 3, 4]$.

El primer código Reed-Salomon viene dado por la evaluación del polinomio $f_1(x) = 10 + 10x + x^2$ en los puntos $P = \{0, 1, 2, 3, 4, 5\}$, es decir, $RS_{k_1}(a) = \{S_1, f_1(1), f_1(2), f_1(3), f_1(4), f_1(5)\}$.

El segundo código Reed-Salomon viene dado por la evaluación del polinomio $f_2(x) = 6 + 5x + 2x^2$ en los puntos $P = \{0, 1, 2, 3, 4, 5\}$ es decir $RS_{k_2}(b) = \{S_2, f_2(1), f_2(2), f_2(3), f_2(4), f_2(5)\}$.

$$f_1(P) = (10, 10, 1, 5, 0, 8)$$
$$f_2(P) = (6, 2, 2, 6, 3, 4)$$

Entonces el código Reed-Salomon formado de la multiplicación es un código Reed-Salomon $C^{*2}\ [6, 5, 5]$:

$$f_1(P) \cdot f_2(P) = (5, 9, 2, 8, 0, 9)$$

En el que el secreto es la primera coordenada, $S_1 \cdot S_2 = 5$.

Capítulo 5

Votación electrónica

Una vez explicados los esquemas de compartición de secretos y *MPC* seguro frente la corrupción pasiva, buscaremos la aplicación de estos conceptos en el protocolo de votación electrónica segura frente a la corrupción pasiva: [30], [31], [32] y [33].

Todos nos hemos encontrado en la tesitura de tener que emitir un voto para una elección, ya sea un tema simple y rápido, o un tema mucho más complejo como unas elecciones gubernamentales.

Una visión muy general del protocolo que vamos a describir es permitir que los participantes puedan votar de manera conforme (SÍ= 1) o desconforme (NO= 0) a una consulta y publicar la suma de dichos resultados en un canal público. Cuando la votación haya finalizado, cualquier grupo de participantes autorizado tomará los votos y calculará el resultado final. Usaremos *MPC* y el esquema de Shamir para que ningún participante tenga información de los votos emitidos por los demás participantes. Además, estos protocolos dotarán de gran consistencia y seguridad a la votación electrónica.

Podemos dividir en tres partes el protocolo que vamos a explicar. Una primera fase de incialización, una segunda de elección y por último el recuento de los votos. Detallaremos algo más estas fases.

1. Inicialización: En esta primera fase se publican los protocolos de seguridad y el votante tendrá que acreditarse como participante con sus credenciales para poder votar.

2. Emisión de votos: En esta fase ya estamos en la elección real. Los votantes emiten el voto, cada votante vota 1 o 0 y genera un secreto aleatorio. Además los participantes en la votación crean particiones del secreto para que su voto no sea descubierto por los demás participantes.

3. Recuento: Es la fase que se produce después de la elección. El conteo de votos se realiza por las autoridades. Estos suman los resultados y los publican. Al menos $t+1$ participantes acumulan y descifran sus particiones. Por último se presenta el cómputo final del total de votos.

Para entender mejor el protocolo de votación electrónica, le explicaremos desde el punto en el que todo el mundo colabora y se obtiene como resultado final la suma de todos los votos de los participantes. Este ejemplo es un método resistente frente a la corrupción pasiva de t participantes y no frente a la corrupción activa:

Si consideramos el esquema de Shamir umbral (t, n) en el que existen n participantes, nuestro ejemplo será resistente frente a t participantes, es decir, la resistencia del protocolo frente a este tipo de corrupción depende del esquema de Shamir (t, n) utilizado.

Paso 1. En este primer paso, los participantes emitirán un voto cuyo valor sera 0 si no están de acuerdo y 1 si su voto es conforme a la consulta realizada. Por lo tanto, $S_i = 0$ o 1.
Un participante j que es apto para votar, prepara su papeleta creando el vector de particiones $s_j : (y_1^j, \ldots, y_n^j)$, usando el esquema de Shamir (t, n).

Paso 2. El participante j guarda la información correspondiente a su voto y envía la partición y_i^j al participante i.

Paso 3. Cada participante tiene su propio vector de particiones formado con la información que le han enviado otros participantes. Al jugador j le pertenece el vector de particiones $(y_j^1, \ldots, y_j^n)$. El participante j calcula $d_j = y_j^1 + \ldots + y_j^n = \sum_{i=1}^{n} y_j^i$. Todos los participantes se encargan de realizar sus propios cálculos.

Paso 4. Los participantes ponen en común d_j y forman el vector $(d_1, \ldots, d_n)$. Además, se recupera el secreto asociado: $S = \sum_{i=1}^{n} S_i$ mediante la interpolación de Lagrange asociada al esquema de Shamir (t, n).

Paso 5. Por último, se da el resultado final de la votación tras el recuento de cada uno de los votos.

En este ejemplo solo se filtra la salida (output) del secreto y como el esquema de

Shamir es lineal, la suma de secretos es igual a la suma de las particiones, como vimos en la sección 4.3.2. Entonces se puede realizar la suma de particiones tal que:

$$f(S_1, \ldots, S_n) = \sum_{i=1}^{n} S_i = S_1 + \ldots + S_n \in \mathbb{F}_q, \ \ con \ \ q > n.$$

El esquema de Shamir (t, n) marca el umbral de reconstrucción del protocolo $r(S) = t + 1$ y el de privacidad $t(S) = t$. Entonces es posible recuperar S con $t + 1$ participantes, $1 \leq t \leq n$.

Ejemplo 5.1. A continuación, daremos un ejemplo para mostrar de manera numérica este protocolo.
Suponemos que existe un jurado popular constituido por 5 personas en un juicio. Los 5 participantes se reúnen después de la realización del juicio para tomar la decisión de sí el acusado es culpable o por el contrario inocente y posteriormente dar el veredicto.

- **Paso 1.**

  Nos encontramos en el cuerpo $\mathbb{F}_7$ con el conjunto de participantes $P_1, \ldots, P_5$. El esquema de Shamir que utilizaremos sera el esquema umbral $(4, 5)$, es decir, el umbral de privacidad en este caso sería $t(S) = 4$ y el de reconstrucción $r(S) = 5$.

  Este ejemplo es el más robusto posible ya que se necesita de los 5 participantes para recuperar el secreto, pero podrían existir otros casos dependiendo el esquema de Shamir que utilicemos.

  Los votantes se encargan de emitir sus votos, ya sea 0 o 1. Además construyen el polinomio aleatorio de grado como máximo $n - 1 = t = 4$.

Votante 1	
Voto S_1	1
Polinomio aleatorio $f(x)$	$1+2x+2x^2$
Votante 2	
Voto S_2	0
Polinomio aleatorio $f(x)$	$0+3x+2x^2$
Votante 3	
Voto S_3	0
Polinomio aleatorio $f(x)$	$0+x+2x^2$
Votante 4	
Voto S_4	1
Polinomio aleatorio $f(x)$	$1+x+x^2$
Votante 5	
Voto S_5	1
Polinomio aleatorio $f(x)$	$1+2x+x^2$

Tabla 5.1: Voto, secreto y polinomio de cada votante

Los votantes crean sus particiones f (x)					
Voto/punto	f(1)	f(2)	f(3)	f(4)	f(5)
$f_1(x)=1+2x+2x^2$	5	6	4	6	5
$f_2(x)=0+3x+2x^2$	5	0	6	2	2
$f_3(x)=0+x+2x^2$	3	3	0	1	6
$f_4(x)=1+x+x^2$	3	0	6	0	3
$f_5(x)=1+2x+x^2$	4	2	2	4	1

Tabla 5.2: Se calculan $f_1(1) = 5$ (*mod* 7), $f_1(2) = 13$ (*mod* 7), $f_1(3) = 25$ (*mod* 7)...

- **Paso 2.**

  Cada votante distribuye el cifrado de las particiones descritas en la tabla anterior y recibe las siguientes particiones cifradas.

Cada participante recibe las particiones					
Votante	Partición 1	Partición 2	Partición 3	Partición 4	Partición 5
Votante 1	5	5	3	3	4
Votante 2	6	0	3	0	2
Votante 3	4	6	0	6	2
Votante 4	6	2	1	0	4
Votante 5	5	2	6	3	1

Tabla 5.3: Los participantes reciben las particiones encriptadas.

- **Paso 3.**

  Cada participante se encarga de calcular $\sum_{i=1}^{n} y_j^i = d_j$ y ponen en común $(d_1, \ldots, d_n)$.

Se calcula d_j		
Votante 1	$d_1 = 5+5+3+3+4$	mód $7 = 6$
Votante 2	$d_2 = 6+0+3+0+2$	mód $7 = 4$
Votante 3	$d_3 = 4+6+0+6+2$	mód $7 = 4$
Votante 4	$d_4 = 6+2+1+0+4$	mód $7 = 6$
Votante 5	$d_5 = 5+2+6+3+1$	mód $7 = 3$

Tabla 5.5: Se calcula $(d_1, \ldots, d_n) = (6, 4, 4, 6, 3)$.

- **Paso 4.**

  Una vez calculado $(d_1, \ldots, d_n) = (6, 4, 4, 6, 3)$ se recupera el secreto asociado: $S = \sum_{i=1}^{n} S_i$ mediante la interpolación de Lagrange, nota 3.13.

  Calculamos $S_i' = \prod_{i \neq j}^{k} \frac{-x_j}{x_i - x_j}$ mód p.

$$S_1' = \frac{-x_2}{(x_1 - x_2)} \frac{-x_3}{(x_1 - x_3)} \frac{-x_4}{(x_1 - x_4)} \frac{-x_5}{(x_1 - x_5)} \quad \text{mód } 7 =$$

$$\frac{-2}{(1-2)}\frac{-3}{(1-3)}\frac{-4}{(1-4)}\frac{-5}{(1-5)} = 5$$

$$S'_2 = \frac{-x_1}{(x_2-x_1)}\frac{-x_3}{(x_2-x_3)}\frac{-x_4}{(x_2-x_4)}\frac{-x_5}{(x_2-x_5)} \quad \text{mód } 7 =$$
$$\frac{-1}{(2-1)}\frac{-3}{(2-3)}\frac{-4}{(2-4)}\frac{-5}{(2-5)} = 4$$

$$S'_3 = \frac{-x_1}{(x_3-x_1)}\frac{-x_2}{(x_3-x_2)}\frac{-x_4}{(x_3-x_4)}\frac{-x_5}{(x_3-x_5)} \quad \text{mód } 7 =$$
$$\frac{-1}{(3-1)}\frac{-2}{(3-2)}\frac{-4}{(3-4)}\frac{-5}{(3-5)} = 3$$

$$S'_4 = \frac{-x_1}{(x_4-x_1)}\frac{-x_2}{(x_4-x_2)}\frac{-x_3}{(x_4-x_3)}\frac{-x_5}{(x_4-x_5)} \quad \text{mód } 7 =$$
$$\frac{-1}{(4-1)}\frac{-2}{(4-2)}\frac{-3}{(4-3)}\frac{-5}{(4-5)} = 2$$

$$S'_5 = \frac{-x_1}{(x_5-x_1)}\frac{-x_2}{(x_5-x_2)}\frac{-x_3}{(x_5-x_3)}\frac{-x_4}{(x_5-x_4)} \quad \text{mód } 7 =$$
$$\frac{-1}{(5-1)}\frac{-2}{(5-2)}\frac{-3}{(5-3)}\frac{-4}{(5-4)} = 1$$

- **Paso 5.**

  Realizamos las reconstrucción: $(6, 4, 4, 6, 3)$

  $$S = \sum_{i=1}^{t} d_i \;\; S'_i \quad \text{mód } p$$

  Se calcula el resultado final:

  $$S = d_1 \cdot S'_1 + d_2 \cdot S'_2 + d_3 \cdot S'_3 + d_4 \cdot S'_4 + d_5 \cdot S'_5 =$$

  $$6 \cdot 5 + 4 \cdot 4 + 4 \cdot 3 + 6 \cdot 2 + 3 \cdot 1 = 73 \quad \text{mód } 7 = 3$$

  Una vez realizado el recuento existen 3 votos a favor, por lo tanto, el acusado es declarado culpable por el jurado popular.

Nota 5.1. Observamos que el protocolo no es seguro frente a la corrupción activa, dado que si el participante P_5 miente y dice que $d_5 = 0$ en lugar de 3 que era el resultado correcto tenemos que:

$$S = 6 \cdot 5 + 4 \cdot 4 + 4 \cdot 3 + 6 \cdot 2 + 0 \cdot 1 = 70 \quad \text{mód } 7 = 0$$

En este caso aunque el protocolo no es seguro frente a la corrupción activa, el sentido común del jurado popular se daría cuenta de que el resultado no es correcto, ya que cualquiera de los participantes de la votación que voto como culpable al acusado, indicaría que es imposible que existieran 0 votos a favor de la condena.

Entonces ahora veremos que si el participante P_5 miente y da el valor $d_5 = 2$:

$$S = 6 \cdot 5 + 4 \cdot 4 + 4 \cdot 3 + 6 \cdot 2 + 2 \cdot 1 = 72 \quad \text{mód } 7 = 2$$

En este caso el acusado sería declarado inocente y nadie se daría cuenta del error en el resultado de la votación.

También es posible que existan privilegios como vimos en la sección 3.6.2 con el ejemplo en el que Ana como presidenta del banco era la única que podía acceder al dinero unida a uno de los cajeros, pero los cajeros juntos no podían acceder al dinero sin Ana.
En el siguiente ejemplo de votación electrónica mostraremos que los votos tienen distinto valor dependiendo del votante:

Ejemplo 5.2. Consideramos que nos encontramos en una votación entre los accionistas de una empresa que quiere decidir sobre el futuro de negocio de la misma. El comité de la votación esta formado al igual que antes por 5 participantes, pero ahora el segundo y tercer votante tienen ambos el $\frac{2}{7}$ de las acciones y los otros dos participantes el $\frac{1}{7}$.

Tomando los datos del ejemplo anterior hasta el **Paso 3**, es decir, los votos de cada uno de los participantes son: $S_1 = 1$, $S_2 = 0$, $S_3 = 0$, $S_4 = 1$ y $S_5 = 1$ en $\mathbb{F}_7$. Cada participante se encarga de calcular $d_j = \sum_{\{j=1,i=1\}}^{5} \lambda_j y_j^i$ y ponen en común $(d_1, \ldots, d_5)$. Tomando los datos de la tabla 5.2 y el ejemplo planteado de los accionistas, los datos que tenemos que calcular en este paso son la combinación lineal: $(5,6,4,6,5) + 2(5,0,6,2,2) + 2(3,3,0,1,6) + (3,0,6,0,3) + (4,2,2,4,1)$.

Se calcula d_j		
Votante 1	$d_1 = 5 + 2 \cdot 5 + 2 \cdot 3 + 3 + 4$	mód $7 = 0$
Votante 2	$d_2 = 6 + 2 \cdot 0 + 2 \cdot 3 + 0 + 2$	mód $7 = 0$
Votante 3	$d_3 = 4 + 2 \cdot 6 + 2 \cdot 0 + 6 + 2$	mód $7 = 3$
Votante 4	$d_4 = 6 + 2 \cdot 2 + 2 \cdot 1 + 0 + 4$	mód $7 = 2$
Votante 5	$d_5 = 5 + 2 \cdot 2 + 2 \cdot 6 + 3 + 1$	mód $7 = 4$

Tabla 5.5: Se calcula $(d_1, \ldots, d_n) = (0, 0, 3, 2, 4)$.

Si seguimos usando los datos calculados en el ejemplo 5.1 :$S'_i = \prod_{i \neq j} \frac{-x_j}{x_i - x_j}$ mód p: $S'_1 = 5$, $S'_2 = 4$, $S'_3 = 3$, $S'_4 = 2$ y $S'_5 = 1$.

Se calcula el resultado final :

$$S = d_1 \cdot S'_1 + d_2 \cdot S'_2 + d_3 \cdot S'_3 + d_4 \cdot S'_4 + d_5 \cdot S'_5 =$$

$$= 0 \cdot 5 + 0 \cdot 4 + 3 \cdot 3 + 2 \cdot 2 + 4 \cdot 1 = 17 \quad \text{mód } 7 = 3$$

Una vez realizado el recuento existen 3 votos a favor. Por lo tanto, como ahora nos encontramos con una votación en la que existen 7 votos en función de las acciones, la votación ha sido considerada como negativa ante la consulta practicada.
A pesar de que sólo 2 votantes de 5 votaron de forma negativa en la votación, su peso en el consenso es superior a la suma de los otros 3 y el resultado de la votación es no favorable.

Los proyectos de voto electrónico se cuentan por millares desde hace años, la votación electrónica ha ido en aumento en los últimos tiempos. La importancia de este protocolo en la actualidad, hace que integrantes de empresas pioneras en el mundo del blockchain, tengan equipos completos trabajando sobre el voto electrónico, intentando crear un proyecto consolidado y totalmente seguros.

Aunque el voto electrónico esta instaurado de lleno en el mundo real, destaca ver cómo siguen siendo aún casos aislados en votaciones de decisiones políticas o excepciones como el voto de personas que se encuentran viviendo en el extranjero. El problema es que se necesita primero el cambio de la ley electoral para permitir que este protocolo se siga desarrollando.

La pandemia del Covid-19 ha reabierto el debate sobre todas las ventajas que puede aportar la implantación de este protocolo. Actualmente ya se puede asegurar que la votación electrónica sea totalmente segura. La instauración de este protocoles es complicada, dado al coste que conlleva y a las leyes electorales en las distintas naciones,

pero lo que es claro es que la votación electrónica ira aumentando de manera lenta pero estará cada vez más presente en nuestros días.

Capítulo 6

Conclusiones

El proyecto es un estudio teórico de conceptos y protocolos tanto criptográficos como matemáticos. Un buen entendimiento de la parte teórica ha permitido el desarrollo de ejemplos prácticos en todas las secciones. Siempre hemos prestado una principal atención a la seguridad de la información. El objetivo final ha sido la creación del protocolo de votación electrónica, seguro frente a la corrupción pasiva.

Este proyecto nos ha permitido poner en práctica diferentes conocimientos vistos durante el Grado como puede ser la seguridad informática, criptografía y teoría de grupos. Sin embargo, la importancia fundamental del proyecto ha sido la adquisición de madurez y nuevas competencias, así como los objetivos prefijados en la sección 1.2, los cuales se han alcanzado en su totalidad.

1. El estudio de la Teoría de Códigos nos permitió entender mejor las secciones posteriores y un primer contacto con la codificación y decodificación.

2. El estudio de nuevas ramas criptográficas como los diferentes esquemas de compartición de secretos, nos permitirán preservar la información y mantenerla segura contra ataques externos.

3. El estudio de estos ataques externos de adversarios, ha hecho posible crear protocolos totalmente seguros con el uso de la computación segura entre múltiples partes *MPC*, consiguiendo diseños sólidos y eficientes.

4. El estudio general, justificación matemática y ejemplos numéricos realizados acerca del protocolo de votación electrónica han conllevado la adquisición de plenos conocimientos sobre este protocolo. Pensamos que esta estructura es una manera óptima de aprender el protocolo mientras se prueban los conceptos en la práctica.

6.1. Ampliación de futuro

Para terminar el documento profundizaremos posibles mejoras que pueden contemplar un proyecto mucho más completo. Consideramos que este documento unido al trabajo que hemos realizado en el Grado de Infórmatica [28], es un estudio teórico de diferentes ramas criptográficas y de teoría de grupos, que puede servir para una buena adquisición de conceptos.

Al existir ejemplos detallados en todas las secciones, este estudio nos puede servir como manual en futuras ampliaciones de personal. Si queremos ampliar el estudio teórico a un aplicativo práctico, podemos contratar desarrolladores de software, que pueden aprender los conceptos con una curva de aprendizaje poco pronunciada. Además, los conocimientos previos de análisis y de diseño sobre los protocolos, suponen que podamos desarrollar aplicaciones que lleven a cabo los algoritmos y nos permita digitalizar el proceso, manteniendo la información de forma segura en todo momento.
El aplicativo ha de ser escalable y aunque se tenga experiencia práctica en programación, nos podremos enfrentar a situaciones en las que la teoría nos ayudará a aclarar cómo resolver distintas situaciones. Por este motivo es totalmente beneficioso adquirir un conocimiento detallado de los conceptos matemáticos y criptográficos.

Hemos desarrollado en la sección 4.3.2 un protocolo *MPC* seguro frente a la corrupción pasiva, en futuras ampliaciones podemos extender esta idea para que funcione para un protocolo *MPC* de seguridad activa.
La construcción de un protocolo resistente a la corrupción activa es mucho más compleja, dado que no sabemos como actúan los adversarios corruptos de manera activa. Es importante identificar porque fallan estos puntos para un protocolo *MPC* de seguridad activa:

La detección de errores 2.2 es posible para reconstruir las particiones de grado t por lo que este paso no es un problema tampoco en la seguridad activa. El problema existe en las particiones de grado $2t$, ya que si tenemos $n = 2t + 1$ cualquier conjunto de n particiones es consistente un un polinomio de grado $2t$ y será imposible saber si los participantes corruptos cambiaron o no las particiones.

Por último, en la votación electrónica descrita en 5 solo se contemplan los votos $\{0, 1\}$. Se puede ampliar este concepto a votos nulos y a diversas elecciones, además de mejorar la seguridad frente a la corrupción activa, al igual que para el protocolo *MPC*.
El tema de la seguridad y el tema legal son la principal piedra que hace que estos sistemas electrónicos tarden en instaurase en la realidad. Pero ya existen diseños seguros

de este protocolo frente a todo tipo de ataques. En el mundo real, la votación electrónica es una solución para sistemas de votación tradicionales que presentan muchos problemas con el censo, el registro y con elecciones nuevas y libres.

Bibliografía

[1] BLAKLEY, G. R.. *Safeguarding cryptographic keys*. Proceedings AFIPS 1979 National Computer Conference, 48, 313-317, 1997.

[2] CASCUDO, I., *Secret Sharing Schemes with Algebraic Properties and Applications*, Department of Mathematics, Aalborg University, Aalborg, Denmark 2016.

[3] CHEN, H. R. CRAMER, R. DE HAAN, R., et al. *Strongly multiplicative ramp schemes from high degree rational points on curves*. N. Smart (Ed.), EUROCRYPT 2008, LNCS, vol. 4965, Springer, Heidelberg (2008), pp. 451 − 470.

[4] CHEN, H., CRAMER, R., GOLDWASSER, S., DE HAAN, R AND VAIKUNTANATHAN, V., *" Secure Computation from Random Error Correcting Codes,"* In Proceedings of 26th Annual IACR EUROCRYPT, Barcelona, Spain, Springer Verlag LNCS, vol. 4515, pp. 329-346, May 2007.

[5] DELGADO, F., AND NÚÑEZ, A, *Esquemas para compartir secretos*, Universidad de Valladolid 2018.

[6] GONZÁLEZ M. I., *Criptografía Avanzada*, Universidad Rey Juan Carlos 2018.

[7] HAIFA, I., *Un curso intensivo sobre MPC*, Medium , 2020, https://medium.com/applied-mpc/a-crash-course-on-mpc-part-2-fe6f847640ae.

[8] HAO CHEN, RONALD CRAMER, SHAFI GOLDWASSER, ROBBERT DE HAAN AND VINOD VAIKUNTANATHAN, *Secure Computation from Random Error Correcting Codes*, Department of Computing and Information Technology, School of Information Science and Engineering, Fudan University, Shanghai, China 2007.

[9] JUSTESEN J. AND, HØHOLD, T., *A Course in Error-Correcting Codes*, European Mathematical Society Publishing House 2004.

[10] KATZ, J. AND LINDELL, Y., *Introduction to Modern Cryptography: Principles and Protocols*, Chapman y Hall, 2014.

[11] MALKHI, D., *An advance course in computer and network security. Lecture Notes*, 2002, Disponible en: http://www.cs.huji.ac.il/ns/SS.doc.

[12] MASSEY, J. L, *"Some applications of coding theory in cryptography,"* in P.G Farrell (ed.), Codes and Ciphers, Cryptography and Coding IV, Formara Lt, Esses, England, pp. 33-47, 1995.

[13] MCELIECE, R. AND SARWATE, D., *"On Sharing Secrets and Reed-Solomon Codes,"* Communications of the ACM, vol. 24, pp. 583-584, 1981.

[14] SCHOENMAKERS, B., *A Simple Publicly Verifiable Secret Sharing Scheme and Its Application to Electronic Voting*, Department of Mathematics and Computing Science Eindhoven University of Technology 1999.

[15] MEIJERING, E., *«A chronology of interpolation: from ancient astronomy to modern signal and image processing»*, Proceedings of the IEEE 90 (3): 319-342, 2002.

[16] MUNUERA, C. AND TENA, J., *Codificación de la Información*, Universidad de Valladolid 1997.

[17] PADRÓ, C., *Lecture Notes in Secret Sharing*, Nanyang Technological University, Singapore 2013.

[18] PENG, KUN, COLIN, DAWSON EDWARD, *Optimization of Electronic First-Bid Sealed-Bid Auction Based on Homomorphic Secret Sharing*, Queensland University of Technology 2005.

[19] QI CHEN, DINGYI PEI, CHUNMING TANG, QIANG YUE, TONGKAI JIA. *Note on ramp secret sharing schemes from error-correcting codes*, 2011.

[20] RUANO, D., *Esquemas de compartición de secretos en rampa*, IMUVA, 2018.

[21] SHAMIR, A. (1979). *How to share a secret*. Communications of the ACM, 22, 612-613.

[22] THOMAS M. COVER, JOY A. THOMAS, *Elements of Information Theory* , City College of New York 2006.

[23] TRAPPE, W. *Introduction to Cryptography*, Prentice Hall, 2002.

[24] MCWILLIAMS, F.J., SLOANE, N.J.A.*The Theory of Error-Correcting Codes*, North Holland, 1988.

[25] ROMAR DELA CRUZ, ANNIKA MEYER AND PATRICK SOLÉ *'An extension of Massey scheme for secret sharing'*- Division of Mathematical Sciences, SPMS, Nanyang Technological University, Singapore.

[26] MASSEY, J. L. . *Minimal codewords and secret sharing*. Proceedings 6th Joint Swedish-Russian International Workshop on InformationTheory, 276 – 279, (1993).

[27] ABASCAL, P., TENA, J.*Algoritmos de búsqueda de un Código Corrector de Errores realizando una Estructura de Acceso para Compartir Secretos*. Actas de la V Reunión Española sobre Criptografía y Seguridad de la Información, 279-283. 283.

[28] MUÑOZ I. , *Compartición de secretos y aplicaciones*, Trabajo fin de grado informática, 2021.

[29] CETINKAYA, O., *Cryptography in electronic voting systems*, International Conference on Government and Governance, 2009.

[30] CETINKAYA, O., *Analysis of security requirements for cryptography voting protocols (extended abstract)*, Third International Conference on Availability, Reliability and Security, 2008.

[31] CETINKAYA, O., *Cryptography in electronic voting systems*, International Conference on Government and Governance, 2009.

[32] CETINKAYA, O., *Analysis of security requirements for cryptography voting protocols (extended abstract)*, Third International Conference on Availability, Reliability and Security, 2008.

[33] DAMGÅRD, I. GROTH, J. AND ALOMONSEN, G., *The Theory and Implementation of an Electronic Voting System*, Kluwer Academic Publishers, 2002.

[34] SCHOENMAKERS, B., *A Simple Publicly Verifiable Secret Sharing Scheme and Its Application to Electronic Voting*, Department of Mathematics and Computing Science Eindhoven University of Technology 1999.